Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 7703466, 10 pages
https://doi.org/10.1155/2021/7703466

Research Article

Practical KGC-Free Polynomial-Based Multiple Group Keys Agreement for IoT Health Care Systems

Qi Cheng,[1] Zhuo Zhao,[2] Chingfang Hsu,[2] Maoyuan Zhang,[2] Qing Yang,[2] and Di Wu[2]

[1]*Product Engineering Department, Wuhan Digital Engineering Institute, Wuhan 430074, China*
[2]*Computer School, Central China Normal University, Wuhan 430079, China*

Correspondence should be addressed to Chingfang Hsu; cherryjingfang@gmail.com

Received 8 August 2021; Revised 31 August 2021; Accepted 13 September 2021; Published 22 September 2021

Academic Editor: Chien Ming Chen

Although nowadays lots of group key agreement schemes have been presented, most of these protocols generate a secret key for a single group. However, in the IoT HCS, more and more communications are involved in multiple groups and users can join multiple groups to communicate at the same time. Therefore, applying the conventional public-key-based one-at-a-time group key establishment protocols has heavy computational cost or suffer from security vulnerabilities. At the same time, in an IoT HCS, a trusted KGC is usually not available and so more flexible self-organized multigroup keys generation will be desired by all group members. In order to address this issue, a practical scheme for efficient and flexible KGC-free polynomial-based multigroup key establishments for IoT HCS is proposed. The proposed protocol can generate multiple group keys for all group members at once, instead of generating one key each time for a single group; more importantly, there is no need for a trusted KGC in the process of group keys establishment and each user can join multiple groups at the same time using only one reserved share. Meanwhile, the security of the proposed protocol is discussed in detail. Finally, we compare this protocol with the latest related group key distribution protocols in performance analysis. The results show that this efficient and flexible KGC-free polynomial-based multiple group keys establishment protocol is more suitable for practical group key agreement in IoT HCS.

1. Introduction

The widespread application of the Internet of Things (IoT) brings great opportunities to the health care system (HCS). The IoT-based HCS provides enormous convenience for group communication among doctors, patients, paramedics, ambulances, and hospitals. The healthcare system can transmit the medical information collected by the internal equipment to multiple members in the group. Since medical information involves the life safety of patients, ensuring the safety of personal health information is crucial [1–6]. HCS is mainly responsible for collecting patient's health information and transmitting this information to group members in the system through the access point. When information is transmitted on the network, it is vulnerable to malicious attacks such as eavesdropping, tampering, and replay. It is possible that the adversary performs malicious attacks and manipulates the information transmitted on the network, which will threaten the lives of patients. The sensitivity of medical data brings many privacy and security issues to the IoT-based HCS. For example, an adversary may eavesdrop on medical information transmitted on the network [7–10]; an adversary may destroy the key used to encrypt data. Hence, it is essential to protect the security of medical data. Only when safety is guaranteed can the hospital provide better services to patients. Then, it is necessary to provide security services for the IoT-based HCS to resist various attacks. For data security, the source node and target node need to share a key before communicating. This process is called key distribution or establishment in IoT-based HCS. It is worth noting that these nodes have small memory space, slow operation speed, and limited battery power. Therefore, a lightweight key distribution protocol needs to be designed for the IoT-based HCS.

The asymmetric cryptographic schemes (e.g., RSA [11]) are impractically used in IoT HCS due to node's inherit

characteristics such as limited memory, power, and CPU [12, 13]. There are many methods that can be used to design a secure key distribution protocol for the IoT-based HCS. One way is to use a master key to preload all nodes, which has the advantages of low memory consumption and no communication/computing overhead. Unfortunately, this method cannot resist node capture attacks because all nodes are preloaded with the same key. Once one node is captured, the entire network may be in danger. Another method is to use the paired key shared between the two nodes to preload each node, where each node needs to store the paired keys shared with other nodes. Obviously, this method is able to withstand the node capture attack, but the storage space of the node will increase linearly with the increase of the network size. Therefore, it is impractical to apply this method to large networks.

In recent years, memory consumption, computing and communication efficiency, connectivity, and robustness to node capture attacks have been focused on by key distribution protocols. At the same time, as an important part of group-based services, in IoT HCS, it is important to ensure secure communication between all group members through the group key. This goal can be achieved through the key establishment protocol. Secret sharing (SS) is computational complexity based on polynomials and unconditional security. Due to its special advantage, it becomes a very popular tool to design group key establishment protocols and then many types of SS are proposed, such as dynamic threshold SS [14]. A linear secret sharing scheme is designed by Hsu et al. [15] using Vandermonde Matrix to effectively generate the group key. Recently, by using an asymmetric bivariate polynomial, [16] proposed a lightweight construction, which realizes both the verification of membership and the establishment of group key. At the same time, lots of group key establishment protocols based on public key cryptography methods have been presented. However, most of these protocols [17–21] are the same as the above two protocols [15, 16], only one key can be generated at a time for a single group.

Group-oriented communication services have attracted widespread attention, and it is increasingly being used in HCS based on the IoT, enabling users to join multiple groups at the same time to facilitate communication. At present, the application of multigroup communication in HCS based on the IoT faces two important issues of security and privacy. In order to meet the challenge, we study how to establish efficient and secure multigroup keys for many-to-many group communication in HCS based on the IoT. The traditional group key agreement protocol can be used in a straightforward way to establish multiple group keys. But in this protocol, if users frequently leave or join group communication, the system needs to constantly regenerate new group keys, which greatly increases the overhead. Hence, the traditional group key agreement protocol has the disadvantage of higher computational cost, and it is not suitable for IoT HCS. More importantly, in an IoT HCS, a trusted KGC is usually not available, then self-organized KGC-free multiple group keys generation will be more desired by all group members.

To solve this problem, an efficient and flexible polynomial-based self-organized one-time multiple group keys establishment scheme for IoT HCS is presented in this paper. This scheme does not need to distribute a separate group key for each group once at a time and can generate multiple group keys for all group members at one time. In addition, there is no need for a trusted KGC in the process of group keys establishment. Each user uses only one share reserved to join multiple groups at the same time. We define it as self-organize one-time multiple group keys establishment method. Meanwhile, the security properties of our scheme are analyzed in detail. Finally, comparing the performance of our protocol with the latest public-key-based group key establishment protocol, the results show that in IoT HCS our scheme has the advantages of high efficiency and practicality.

Our main contributions are summarized as follows:

(a) We design a polynomial-based self-organize multiple group keys establishment protocol for IoT HCS, in which there is no need for a trusted KGC in the process of group keys establishment, and multigroup keys generation will be performed by all involved group members.

(b) Our method is very efficient since in this protocol each user can join multiple groups at the same time using only one share reserved. There has no rekeying overhead.

(c) One unique feature of our design is that in this protocol the multigroup keys generation is performed by all group members. There is no need to set up a trusted server. It is very flexible for IoT HCS. Moreover, the polynomial is much more efficient than public key calculations. It is truly low computation.

The rest of this paper is arranged as follows. Some related work of the key agreement schemes is discussed in Section 2. Section 3 introduces some essential preliminaries. The model of the proposed protocol is briefly introduced in Section 4. Section 5 presents our polynomial-based self-organize multiple group keys establishment protocol. The correctness and the security are proven in Section 6. Section 7 evaluates the performance of our scheme and makes comparisons between our protocol and latest protocol. At last, we make a summary for this paper in Section 8.

2. Related Work

For HCS based on the IoT, more and more key establishment protocols are proposed [22–30]. Most schemes are implemented in a flat structure and establish a separate key for each group once a time. The following methods are more used in group key agreement schemes, namely random-key predistribution [31], polynomial-based predistribution [32], and grid-based predistribution [33]. The first random key protocol was designed by Eschenauer and Gligor [31]. This scheme first randomly selects a key set from the key space, namely the key pool. Before being deployed, each sensor

node randomly selects a subset from the key pool, called a key ring, and stores it in its own memory. Sensor nodes must look for public keys in their respective key rings before communicating with other nodes. If there is a public key, it will be used as a shared key for both parties to communicate. On the contrary, it is necessary to find a neighbor node that has a public key with both parties in the communication. The random key scheme is a probabilistic scheme. In other words, sensor nodes can only establish a shared key with a certain probability, and it cannot ensure that there is a shared key between all nodes. This requires increasing the size of the key ring of the sensor node to increase the probability of establishing a shared key between nodes. But it will also increase the success rate of node capture attacks. Hence, it is necessary to weigh the advantages and disadvantages between network connections and node capture attacks. The key distribution scheme using polynomials is deterministic, which means that there is a shared key between any two nodes. Suppose that the proposed scheme uses a $t-1$ degree polynomial to establish the shared key for the node, if the number of nodes captured by the adversary is t or more than t, it will pose a threat to the entire network. For the purpose of improving the security of the scheme, the degree of the polynomial needs to be increased, but this makes the storage and calculation overhead of the nodes larger. Therefore, our intention is to design a multigroup key distribution protocol, which has the advantages of high efficiency and high security simultaneously.

A new key management protocol was proposed by Park et al. [34], which is aimed at the coexistence of multiple multicast groups in the same network. In this scheme, three different services are provided by the service provider for the IEEE 802.16 network [35]. The service provider is responsible for managing each user group. When a user exits or joins the user group, the service provider needs to update the broadcast key using asymmetric encryption. However, due to the limited resources of IoT devices, asymmetric encryption increases the computational cost of the key generation process. In the group key agreement scheme proposed in [36], the group key used for encryption is negotiated by members of the group, and then each group member is assigned a key for decryption. Only members of the group can decrypt the ciphertext encrypted by their shared key. Like the above scheme, in [34], this scheme also uses asymmetric encryption to establish multigroup keys. Recently, the authors of [37] proposed a multiparty key agreement based on elliptic curve cryptography (ECC) encryption, which is more computational efficient than RSA, but this protocol also needs a group controller (GC) and rekeying overhead. Hsu et al. [38] proposed an efficient user-oriented multigroup key agreement scheme based on secret sharing, which relies on the trusted key generation center (KGC) to negotiate keys. We observe that in an IoT HCS, if there is no trusted KGC, self-organized one-time multiple group keys generation will be desired by all group members. This observation motivates us to come up with a solution to meet this requirement.

3. Preliminaries

We briefly described the knowledge related to secret sharing in this section. In the secret sharing scheme, the trusted *dealer* splits the secret s into multiple smaller *shares* and transmits them to the participants in the group to realize the sharing of the secret in the same group. Authorized participants in the same group can recover their secrets, while other unauthorized participants cannot recover their secrets. If a scheme can make it impossible for any unauthorized participant to recover the key and obtain any secret-related information, it is regarded as a perfect scheme.

Suppose $P = \{1, \ldots, n\}$ represents a collection of participants. Based on the Shannon's entropy function, [39] proposed that *secret sharing protocol* should meet the following conditions:

(a) *Correctness*. The secrets s can be recovered by authorized participants. In other words, it has $H(S|A) = 0$ for any $A \in \Gamma$. Γ refers to *access structure* that is the collection of authorized participants.

(b) *Security*. It is impossible for the secret s to be recovered by an unauthorized participant. In other words, it has $0 < H(S|A) \leq H(S)$ for any $A \notin \Gamma$. What we are concerned about is $H(S|A) = H(S)$. In this case, any information related to the secret s cannot be obtained by the participants in A. So, the security of this protocol is *perfect*.

If the participant's share is in the same domain as the secret (this is the minimum size of the shares as demonstrated in [40]), a perfect secret sharing protocol is *ideal*.

3.1. Secret Sharing Scheme Based on Polynomials. In Shamir's (t, n) secret sharing scheme [41] based on linear polynomial, the trusted dealer chooses a $t-1$ degree polynomial $f(x)$, where $f(0) = s$. The dealer uses $f(x)$ to split s into smaller shares, $f(x_i)$, $i = 1, 2, \ldots, n$, and distribute them to each participant, where x_i is a public identifier for each participant U_i. This secret sharing scheme meets the above two security features. That are (a) the secret that can be recovered only if the number of shares is not less than t and (b) the number of shares is less than t, it is impossible to recover the secret. Hence, Shamir's (t, n) secret sharing scheme is unconditionally secure, and it contains the following two phases.

3.1.1. Share Generation. Suppose there are n participants, $U = \{U_1, U_2, \ldots, U_n\}$. Dealer D randomly selects a $t-1$ degree polynomial $f(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1} \bmod p$, where the secret $s = f(0) = a_0$, and all coefficients, a_i, $i = 0, 1, \ldots, t-1$, belong to the finite field GF_p with $p > s$. D calculates n shares, $y_i = f(x_i)$, $i = 1, 2, \ldots, n$, where x_i is a public identifier for each participant U_i. Then, each share y_i is secretly distributed to the corresponding participant U_i.

3.1.2. Secret Reconstruction. Assume that there are t participants, $\{U_1, U_2, \ldots, U_t\}$, reconstructing the secret s.

Participants release their shares and recover the secret by using the Lagrange interpolating formula, $s = f(0) = \sum_{i=1}^{t} f(x_i) \prod_{r=1, r \neq i}^{t} (-x_r/(x_i - x_r)) \bmod p$.

4. Model of Our Multigroup Key Agreement Scheme

The model of the proposed multigroup key agreement scheme for IoT HCS is presented in this section, which contains system model and security model.

4.1. System Model. There is a KGC in our proposed protocol for IoT HCS, and it is assumed that there are n users $\{U_1, U_2, \ldots, U_n\}$ participating in multigroup communication. The system model of our proposed protocol is illustrated in Figure 1. These users can be doctors, patient, caretaker, ambulance, and hospital. KGC is responsible for user registration and managing all registered users, including adding users and deleting users. In the IoT HCS, if there is no trusted KGC, all members of the group participating in the communication will negotiate to generate multiple group keys before communication in order to exchange information securely. Generally, self-organized multiple group keys generation should be performed by all group members. Hence, group session keys can only be generated by members in the same group.

During the registration phase of the proposed protocol, each user U_i is secretly assigned a long-term secret generated by KGC. Next, self-organized multigroup keys generation will be performed by all group members. In other words, when accepting the key agreement request initiated by one of the users to multiple groups, each user select one value for each group he joins and transfer each value secretly to the corresponding group members. Then, each user uses the values received from other group members, who belong to the same group, to recover the polynomial and the corresponding group key and further authenticates that this group key is the same with other group members. Later, members in the group use the generated self-organizing multigroup key for secure communication.

Public key calculation uses a large modulus, such as at least 1024 bits in RSA. In comparison, polynomial encryption uses a small modulus, only 160 bits. Therefore, our protocol based on polynomial encryption is more efficient and computationally faster. In addition, conventional group key agreement protocols need a mutually trusted KGC generate all group keys for multigroups. This method relies on trusted servers and will incur communication and storage overhead in IoT HCS. The problem with the trusted server is that if it is attacked, the network will be completely insecure. In order to address the problem, self-organized multigroup keys generation is performed by all group members. This makes our protocol very effective and practical.

4.2. Security Model. We briefly describe the security model to evaluate the security of the proposed scheme.

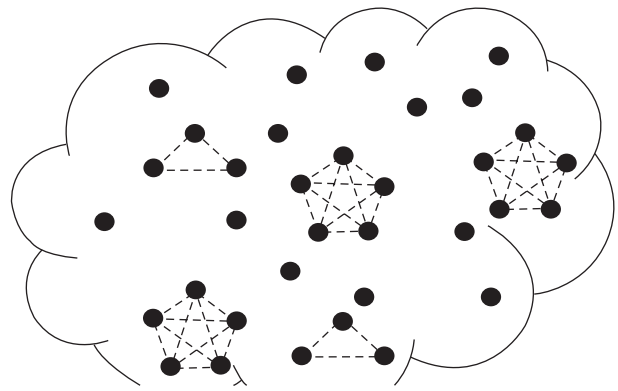

Figure 1: Multiple-group applications in IoT HSC.

4.2.1. Type of Adversaries. Our protocol mainly analyzes two types of adversaries, internal and external. An internal attacker refers to a legitimate member of the group, so the group key is known to him. The internal attacker may attempt to obtain the long-term secret keys of other members, which allows him to impersonate other members for secure communication. In addition, internal attackers may also obtain other group keys without authorizing him to know and leak them out. On the other hand, group keys that are not allowed to be known by outsiders may be maliciously obtained by an external attacker. The confidentiality of the group key affects the success rate of this attack. We will explain in detail that our scheme can resist these attacks in the following security analysis.

4.2.2. Security Features. The following security features need to be satisfied:

(a) Key confidentiality: it is computationally infeasible for external attackers to obtain any group key.

(b) Key authentication: the generated group key is required to be authenticated by group members, which is the same with the corresponding group members.

(c) Key independence: unauthorized users are not able to obtain other group keys $K'' = K - K'$ based on the known subset of group keys $K' \subset K$.

(d) Forward secrecy: ensure that members who have left cannot know the new communication message.

(e) Backward secrecy: ensure that new members cannot know the historical communication message.

5. The Proposed Protocol for Multigroup Communications

Suppose that there are a total of n users participating in multigroup communication $\{U_1, U_2, \ldots, U_n\}$. Before receiving system services, users need to register with KGC. KGC is responsible for user registration and managing all registered users, including adding users and deleting users. Before group members communicate, the session key of each group is distributed to the corresponding members of the same group in a secure manner, which ensures the security

of communication. Generally, the session key of each group is determined by all corresponding members of the group according to the membership to which they belong. Hence, group session keys can only be generated by members in the same group.

Table 1 illustrates the symbols used in this paper is shown in Table 1. There are three stages in the proposed multigroup communication protocol, namely the initialization phase of KGC, the registration phase, and the multigroup key agreement phase. Users participating in multigroup communication are recorded as $\{U_1, U_2, \ldots, U_n\}$, and these groups are recorded as $G_1, G_2, \ldots, G_m$. A multigroup table is determined in Table 2, where if U_i $(1 \le i \le n)$ joins the group G_k $(1 \le k \le m)$, then the corresponding unit $a_{ik} = 1$, else $a_{ik} = 0$. Here we define the rank of a user, $|U_i|$, as the number of nonzero elements in $(a_{i1}, a_{i2}, \ldots, a_{im})$ and define the rank of a group, $|G_k|$, as the number of nonzero elements in $(a_{1k}, a_{2k}, \ldots, a_{nk})$.

The detailed multigroup keys establishment is as follows:

Initialization of KGC. First, the KGC selects a large prime p, a generator g of GF_p, and constructs a secure one-way hash function $h(\cdot)$ based on the domain GF_p. These parameters p, g, and $h(\cdot)$ are published by the KGC.

User Registration. Every user who needs multigroup key agreement service must first register with KGC. KGC is responsible for managing all registered users and updating the number of users in real time. After receiving the user's registration request, KGC generates a long-term secret, $x_i \in \mathrm{GF}_p$, for user U_i and distributes it to U_i secretly and publishes g^{x_i}, where $x_i \neq x_j$, and $i, j \in \{1, \ldots, n\}$. Later in real-time operation, multigroup keys will be calculated by the members of the group participating in the communication using their long-term secrets and used for secure communication between group members.

Multigroup Keys Establishment. In the IoT HCS, if there is no trusted KGC, then self-organized multigroup keys generation should be performed by all group members. Upon receiving multigroup keys agreement request for these groups $G_1, G_2, \ldots, G_m$ from any group member, all involved group members will establish the m corresponding group keys $K_1, K_2, \ldots, K_m$ as the following steps:

Step 1. The initiator broadcasts a multigroup keys establishment request for these groups $G_1, G_2, \ldots, G_m$, where each group $G_k = \left\{U_{k_1}, U_{k_2}, \ldots, U_{k_{|G_k|}}\right\}$, $k \in \{1, \ldots, m\}$ and $k_1, k_2, \ldots, k_{|G_k|} \in \{1, \ldots, n\}$.

Step 2. Each participating group member U_i $(1 \le i \le n)$ responds by broadcasting the list of his involved groups, $G_{i_1}, G_{i_2}, \ldots, G_{i_{|U_i|}}$, $i_1, i_2, \ldots, i_{|U_i|} \in \{1, \ldots, m\}$.

Step 3. Each member U_i $(1 \le i \le n)$ selects and broadcasts a random challenge, $r_i \in \mathrm{GF}_p$.

Step 4. According to the multigroup table, if $a_{ik} = 1$ for $1 \le k \le m$, then each member U_i $(1 \le i \le n)$, needs to randomly select a corresponding value $R_{ik} \in \mathrm{GF}_p$, which is used to compute the group key K_k. Altogether U_i should select $|U_i|$ such values.

Step 5. Each member U_i uses his secret share x_i, his challenge r_i, and the public value $g^{x_j + r_j}$ to calculate the pairwise shared secret keys between U_i and U_j, $k_{i,j} = (g^{x_j + r_j})^{x_i + r_i}$, where $i \neq j$ for $i, j \in \{1, 2, \ldots, n\}$. Then, if $a_{ik} = 1$ and $a_{jk} = 1$ for $1 \le k \le m$, U_i sends the corresponding R_{ik} secretly to U_j as $c_{i,j} = E_{k_{i,j}}(R_{ik})$, $1 \le i \le n$, where $E_{k_{i,j}}(R_{ik})$ represents the encryption of R_{ik} using the key $k_{i,j}$.

Step 6. After receiving ciphertext, $c_{j,i}$ $(i \neq j$ for $i, j \in \{1, 2, \ldots, n\})$ from each member U_j, U_i computes $R_{jk} = D_{k_{j,i}}(c_{j,i})$, where $D_{k_{j,i}}(c_{j,i})$ refers to decrypt $c_{j,i}$ using the key $k_{j,i} = k_{i,j} = (g^{x_j + r_j})^{x_i + r_i}$. Then, for each group G_k $(k \in \{1, 2, \ldots, m\})$ that U_i joins, U_i will altogether obtain $|G_k|$ points (l, R_{lk}), where $a_{lk} = 1$ for $1 \le l \le n$. According to these $|G_k|$ points, user U_i generates a $(|G_k| - 1)$ degree polynomial $f_k(x)$ for each group G_k and select the constant term of $f_k(x)$ as the group key K_k. Then, U_i broadcasts $|U_i|$ such values $h(K_k)$ to all group members, where $a_{lk} = 1$ for $1 \le k \le m$.

Step 7. Each member U_i checks whether these broadcasting $h(K_k)$ for $1 \le k \le m$ are identical, respectively. If they are identical, U_i, for $1 \le i \le n$, authenticates that these m group keys $K_1, K_2, \ldots, K_m$ are valid. If some of these group keys are not identical, the corresponding group members will replay this protocol again. All computations are performed in GF_p.

After successfully completing the above steps, m group keys $K_1, K_2, \ldots, K_m$ associated with $G_1, G_2, \ldots, G_m$, respectively, are self-established among all group members.

Table 1: Notation table.

Notation	Description
U_i	User i
KGC	Key generation center
p	A safe large prime with $p > n$
$f(x)$	A univariate polynomial
g	A generator of GF_p
G_i	Group i
K_i	Secret group communication key i
$h(\cdot)$	One-way hash function
$\|U_i\|$	The rank of user U_i
$\|G_k\|$	The rank of group G_k

Table 2: Multigroup description.

Users	Groups					
	G_1	G_2	...	G_k	...	G_m
U_1	a_{11}	a_{12}	...	a_{1k}	...	a_{1m}
U_2	a_{21}	a_{22}	...	a_{2k}	...	a_{2m}
...	...	...	...	...	...	...
U_i	a_{i1}	a_{i2}	...	a_{ik}	...	a_{im}
...	...	...	...	...	...	...
U_n	a_{n1}	a_{n2}	...	a_{nk}	...	a_{nm}

Then, group members can use these group keys $K_1, K_2, \ldots, K_m$ for secure multigroup communication.

6. Security Analysis

6.1. Correctness. In Step 6, U_i will altogether obtain $|G_k|$ points (l, R_{lk}) from the group G_k he joined, where $a_{lk} = 1$ for $1 \le l \le n$, and $k \in \{1, \ldots, m\}$. According to these points $|G_k|$, user U_i can calculate a $(|G_k| - 1)$ degree polynomial $f_k(x)$ for each group G_k by using the Lagrange interpolation formula and select the constant term of $f_k(x)$ as the group key K_k. Then U_i broadcasts $|U_i|$ such values $h(K_k)$ to all group members, where $a_{ik} = 1$ for $1 \le k \le m$. In Step 7, each member U_i checks whether these broadcasting $h(K_k)$ for $1 \le k \le m$ are identical, respectively. If they are identical, U_i $(1 \le i \le n)$ authenticates that these m group keys $K_1, K_2, \ldots, K_m$ are valid.

6.2. Security. The security of the proposed protocol is discussed by analyzing the following security features:

(1) The proposed scheme can guarantee the freshness, confidentiality, and independence of the key and provide verification for the key.

(2) This scheme is able to withstand attacks that occur on synchronous and asynchronous networks.

(3) The forward and backward safety are guaranteed in this scheme. Forward security refers to the protection of new keys from being obtained by leaving members. Backward security means that new members who join the group cannot obtain the previous key.

(4) Internal attacks and external attacks cannot be achieved in this scheme. The internal attacker does not know other group keys except the key of the group he belongs to. All group keys are not obtained by external attackers.

Theorem 1. *The proposed protocol can ensure the freshness, confidentiality and independence of the key, and provide verification for the key.*

Proof. Key freshness is satisfied since for each request to generate multigroup key, there are m new group keys $K_1, K_2, \ldots, K_m$ associated with $G_1, G_2, \ldots, G_m$, where each group's session key K_k $(k \in \{1, 2, \ldots, m\})$ is decided by all corresponding group members according to the membership to which they belong. Hence, the group session key can only be negotiated by members belonging to the group. In addition, each group member, U_i, uses $|G_k|$ points (l, R_{lk}), where $a_{lk} = 1$ for $1 \le l \le n$ and R_{lk} is randomly selected by U_l, to generate a $(|G_k| - 1)$ degree polynomial $f_k(x)$ and the constant term of $f_k(x)$ is the group key K_k for group G_k.

Key confidentiality is guaranteed by secret sharing protocol. The secret key of each group is decided by all members participating in the communication in the group according to the memberships that they belong to. These group members will interact with each other by fresh pairwise keys, which are computed using their long-term secrets x_i and random challenges r_i. Hence, the group session key can only be negotiated by members belonging to the group.

Key authentication is provided by the value $h(K_1), h(K_2), \ldots, h(K_m)$, which is generated by one-way hash function in Step 6, with the group keys $K_1, K_2, \ldots, K_m$ as input. The secret group key is determined by all members participating in the communication in the group. Besides, the group key cannot be forged by an internal attacker because it is decided by all corresponding group members according to the memberships that they belong to.

Key independence is provided. It means that the group member cannot obtain any other group key information that he has not authorized from the corresponding group key that he has obtained. This is because each group key K_k $(k \in \{1, 2, \ldots, m\})$ is computed by $|G_k|$ points (l, R_{lk}), where $a_{lk} = 1$ for $1 \le l \le n$ and R_{lk} is randomly selected by U_l. The proof process is given in detail in Theorem 5. □

Theorem 2. *The proposed protocol is able to withstand attacks in synchronous and asynchronous networks.*

Proof. Group members will interact with each other by fresh pairwise keys, which are computed using their long-term secrets x_i and random challenges r_i. Each group key K_k $(k \in \{1, 2, \ldots, m\})$ reconstruction is based on $|G_k|$ points (l, R_{lk}), where $a_{lk} = 1$ for $1 \le l \le n$ and R_{lk} is randomly selected by U_l. There is only a list of groups $G_1, G_2, \ldots, G_m$, the parameters p, g, $h(\cdot)$, and g^{x_i}, and random challenges $r_i \in \mathrm{GF}_p$ available. In real-time operation, multigroup keys generation is performed by all involved group members. It is impossible for an attacker to get information related to the key from the asynchronously released values. The proof process is given in detail in Theorems 4 and 5. □

Theorem 3. *The forward and backward secrecy are guaranteed in the proposed scheme, which means that the leaving members are unable to obtain the new group key, and the newly joined member does not know the past key.*

Proof. When the group members change, such as a member leaving the group or a new member joining, in step 1, the list of groups $G_1, G_2, \ldots, G_m$ will be updated in real time. Group key in multigroup session is decided by all corresponding group members according to the memberships that they belong to. Members in the group can only get the session key of the group they are currently in. In other words, new keys will not be obtained by the leaving members. And the previous key cannot be obtained by newly joined group members. Therefore, the proposed protocol guarantees both the forward and backward security of multigroup keys.

Our proposed scheme divides adversaries into two types. One type of adversary is external attacker, which refers to members outside the group. An external attacker may attempt to obtain a private group key that is not allowed to be known by user outside the group. The confidentiality of the key guarantees that external attackers cannot achieve this kind of attack. In addition, our scheme allows any user to

send a request to KGC to obtain the service of multigroup key establishment. Then an external attacker may pretend to be other legitimate members of the group to request the service of key establishment. However, the information related to the group key cannot be obtained by an external attacker through this attack. Because the proposed scheme guarantees that members who are not authorized cannot obtain the group key. The other type of adversary refers to internal attackers. They are authorized to access the group key of their group, but they try to obtain the secrets shared by other members with KGC. Therefore, it is necessary to protect the secrets shared by other members with KGC from inside attackers. □

Theorem 4. *(outsider attack). Suppose there is an adversary impersonating a member of a group. Our scheme guarantees that the attacker cannot acquire the corresponding group key and share the key with other members of the group.*

Proof. In our scheme, any attacker is able to impersonate another member to request services from KGC and get a response message. However, it is guaranteed that only legitimate members of the group can obtain the secret key of the group. In our proposed scheme, group members will interact with each other by fresh pairwise keys, which are computed using their long-term secrets x_i and random challenges r_i. Each group key K_k $(k \in \{1, 2, \ldots, m\})$ reconstruction is based on $|G_k|$ points (l, R_{lk}), where $a_{lk} = 1$ for $1 \le l \le n$ and R_{lk} is randomly selected by U_l. There is only a list of groups $G_1, G_2, \ldots, G_m$, the parameters p, g, $h(\cdot)$, and g^{x_i}, and random challenges $r_i \in \mathrm{GF}_p$ available. The polynomial-based secret sharing scheme, the difficulty of the discrete logarithm problem, and the one-way nature of the hash function protect the secret group key from being acquired by an attacker.

Group members cannot obtain information about other keys that are not allowed to know based on the recovered secret group key. This is because each group key can only be calculated by the long-term secret calculation of the corresponding member in the group. Hence, *key independence* is guaranteed in our protocol.

The possibility of an attacker successfully negotiating the leaked group key with other members by replaying the eavesdropped communication message is negligible. This is because the fresh pairwise keys are computed using their random challenges r_i and each group key K_k $(k \in \{1, 2, \ldots, m\})$ reconstruction is based on $|G_k|$ points (l, R_{lk}), where R_{lk} is randomly selected by U_l. The parameters r_i and R_{lk} are different in each round of communication. Thus, our protocol is able to withstand the *replay attack*. □

Theorem 5. *(insider attack). Suppose the proposed scheme has been performed many times. The secret $x \in \mathcal{K}$ shared by group members and KGC is not known to all other members.*

Proof. The group key in our protocol will be generated by members of the group participating in the communication. Each group's session key is decided by all corresponding group members according to the memberships that they belong to. These group members will interact with each other by fresh pairwise keys, which are computed using their long-term secrets. However, the secret $x \in \mathcal{K}$ shared by group members and KGC is not known by outsiders.

Our scheme does not authenticate the user who sent the service request. Internal attackers can request services from KGC and pretend to be a member of the group to initiate a challenge. Suppose that there is an adversary U_i, he sends a group key agreement service request to the group including himself and member U_{target} and forges the group member's challenge r_{target}. Although the adversary U_i can obtain the group key, the value x_{target} is not known to him, since x_{target} is protected in $k_{i,\text{target}} = (g^{x_{\text{target}}+r_{\text{target}}})^{x_i+r_i}$ due to the difficulty of discrete logarithm problem. Thus, the internal attacker can only obtain the secret group key of the group and cannot know the long-term secrets of other members in the group. Therefore, the insider attack cannot be implemented in the proposed protocol. □

7. Performance Evaluation

By comparing with the recently proposed multigroup key agreement scheme [34, 37] based on public key encryption, the performance of our scheme is evaluated in this section. Then we show the comparison between our protocol and the latest multigroup key establishment protocols.

Compared with the public-key-based multigroup key establishment schemes [34, 37], our protocol has the following advantages:

(a) Flexible and convenient network structures do not require a central server, such as peer-to-peer network. In P2P network, 'peers' are the nodes or computer system that are connected to each other. Files or resources can be shared directly between the system on the network, without the need of any central server. Conventional group key agreement schemes require a central server, namely trusted KGC, to generate all group keys for multigroups. This method needs to set up a trusted server, so it will incur the overhead costs in communications and storages in sensor networks. In addition, if the trusted server is compromised, the network will be insecure. To overcome this drawback, in our protocol, KGC-free multigroup keys generation is performed by all group members.

(b) In the public key broadcast-based scheme [34, 37], the broadcast key needs to be updated when the user changes, which increases the cost of the scheme. In comparison, KGC is responsible for managing member changes in our secret sharing scheme. If a new user joins the group, he only needs to register with KGC and obtain the long-term secret distributed by KGC in a secure way. This process will not affect the long-term secrets of other existing members. In addition, the member's departure only requires KGC to delete the user without regenerating the key.

Table 3: Comparison with latest multigroup key establishment protocols.

	Park et al. [34]	Hsu et al. [38]	Naresh et al. [37]	Ours
Multiple group keys establishment at one time	Yes	Yes	Yes	Yes
Trusted server needed	Yes	Yes	Yes	No
Computations for encryption and decryption	RSA-based	Polynomial-based	Elliptic curve cryptography- (ECC-) based	Polynomial-based
Rekeying overhead needed	Yes	No	Yes	No

(c) It is well known that symmetric key encryption is a way that each pair of users shares a symmetric key, but this way only provides confidentiality. Further, key distribution and management is a bottleneck in symmetric key cryptography, which produces huge communication and storage cost. Hence, public key encryption appeared, which can provide confidentiality, authenticity, and nonrepudiation but with high computation cost due to very large modulus and modular exponentiation operations. Compared with public key operations producing high computation cost, bivariate polynomial-based approach can provide not only authentication and information-theoretic security but also with lower computation cost. At the same time, compared with symmetric key distribution process that needs huge communication cost, bivariate polynomial-based approach is really efficient while providing authentication. In our protocol, the polynomial calculation uses a small modulus, only 160 bits. In comparison, public key calculations not only require a larger modulus (for example, at least 1024 bits in RSA) but also use modular exponentiation, pairing, and scalar multiplication operations (such as ECC-based schemes). Therefore, the calculation efficiency of polynomials is higher than that of public key calculations.

Meanwhile, Table 3 compares our proposed scheme with the latest multigroup key agreement schemes, which demonstrate that our protocol has the optimal performance.

8. Conclusions

We present an efficient and flexible KGC-free polynomial-based multiple group keys establishment protocol for IoT HCS in this paper. The proposed protocol can generate multiple group keys for all group members at one time. In addition, there is no need for a trusted KGC in the process of group keys establishment, and each user can join multiple groups at the same time using only one share reserved. Meanwhile, the security of the proposed protocol is strictly analyzed. Finally, we compare this protocol with the latest multigroup key establishment protocols in performance analysis, which indicates that our KGC-free polynomial-based multiple group keys establishment protocol is fairly attractive for efficient and flexible IoT HCS.

Data Availability

The data used to support the findings of this study are included within the article.

Consent

Informed consent was obtained from all individual participants included in the study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grants nos. 61772224 and 62072133), the Fundamental Research Funds for the Central Universities (No. CCNU19TS019), the Research Planning Project of National Language Committee (No. YB135-40), and the Key projects of Guangxi Natural Science Foundation (No. 2018GXNSFDA281040).

References

[1] S. Kavitha, P. J. A. Alphonse, and Y. V. Reddy, "An improved authentication and security on efficient generalized group key agreement using hyper elliptic curve based public key cryptography for IoT health care system," *Journal of Medical Systems*, vol. 43, no. 8, pp. 1–6, 2019.

[2] K.-H. Yeh, "A secure IoT-based healthcare system with body sensor networks," *IEEE Access*, vol. 4, pp. 10288–10299, 2016.

[3] P. Wang, R. Yu, N. Gao, C. Lin, and Y. Liu, "Task-driven data offloading for fog-enabled urban IoT services," *IEEE Internet of Things Journal*, vol. 8, no. 9, pp. 7562–7574, 2020.

[4] C. Wang, D. Wang, Y. Tu, G. Xu, and H. Wang, "Understanding node capture attacks in user authentication schemes for wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2020.

[5] P. Wang, Z. Yu, C. Lin, L. Yang, Y. Hou, and Q. Zhang, "D2D-Enabled reliable data collection for mobile crowd sensing," in *Proceedings of the 2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 180–187, Hong Kong, China, December 2020.

[6] L. Chen, J. Li, and Y. Zhang, "Anonymous certificate-based broadcast encryption with personalized messages," *IEEE Transactions on Broadcasting*, vol. 66, no. 4, pp. 867–881, 2020.

[7] L. Chen, J. Li, Y. Lu, and Y. Zhang, "Adaptively secure certificate-based broadcast encryption and its application to cloud storage service," *Information Sciences*, vol. 538, pp. 273–289, 2020.

[8] C. Esposito, M. Ficco, and B. B. Gupta, "Blockchain-based authentication and authorization for smart city applications," *Information Processing & Management*, vol. 58, no. 2, Article ID 102468, 2021.

[9] C. L. Stergiou, K. E. Psannis, and B. B. Gupta, "IoT-based big data secure management in the fog over a 6G wireless network," *IEEE Internet of Things Journal*, vol. 8, 2020.

[10] B. B. Gupta and M. Quamara, "An overview of Internet of Things (IoT): architectural aspects, challenges, and protocols," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 21, Article ID e4946, 2020.

[11] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[12] M. Alshammari and K. Elleithy, "Secure and efficient key management protocol (SEKMP) for wireless sensor networks," in *Proceedings of the Tenth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, pp. 253-254, New York, NY, USA, October 2014.

[13] N. Saqib and U. Iqbal, "Security in wireless sensor networks using ECC," in *Proceedings of the 2016 IEEE International Conference on Advances in Computer Applications (ICACA)*, pp. 270–274, Coimbatore, India, October 2016.

[14] C. Hsu, L. Harn, Z. Xia, and M. Zhang, "Non-interactive dealer-free dynamic threshold secret sharing based on standard shamir's SS for 5G networks," *IEEE Access*, vol. 8, pp. 203965–203971, 2020.

[15] C.-F. Hsu, L. Harn, Y. Mu, M. Zhang, and X. Zhu, "Computation-efficient key establishment in wireless group communications," *Wireless Networks*, vol. 23, no. 1, pp. 289–297, 2017.

[16] Q. Cheng, C. Hsu, and L. Harn, "Lightweight noninteractive membership authentication and group key establishment for WSNs," *Mathematical Problems in Engineering*, vol. 2020, Article ID 1452546, 2020.

[17] C.-F. Hsu, L. Harn, T. He, and M. Zhang, "Efficient group key transfer protocol for WSNs," *IEEE Sensors Journal*, vol. 16, no. 11, pp. 4515–4520, 2016.

[18] S. Gupta, B. L. Parne, and N. S. Chaudhari, "DGBES: dynamic group based efficient and secure authentication and key agreement protocol for MTC in LTE/LTE-A networks," *Wireless Personal Communications*, vol. 98, no. 3, pp. 2867–2899, 2018.

[19] J. Zhou, L. Sun, and J. Song, "Efficient group key management for non-reliable link networks," *Wireless Personal Communications*, vol. 98, no. 2, pp. 1955–1973, 2018.

[20] P. Ahlawat and M. Dave, "Deployment based attack resistant key distribution with non overlapping key pools in WSN," *Wireless Personal Communications*, vol. 99, no. 4, pp. 1541–1568, 2018.

[21] N. Ayub, M. Raja, S. Saleh, and M. U. Ilyas, "MuGKeG: secure multi-channel group key generation algorithm for wireless networks," *Wireless Personal Communications*, vol. 96, no. 3, pp. 4799–4818, 2017.

[22] M. P. Đurišić, Z. Tafa, G. Dimić, and V. Milutinović, "A survey of military applications of wireless sensor networks," in *Proceedings of the 2012 Mediterranean Conference on Embedded Computing (MECO)*, pp. 196–199, Budva, Montenegro, June 2012.

[23] T. Azzabi, H. Farhat, and N. Sahli, "A survey on wireless sensor networks security issues and military specificities," in *Proceedings of the 2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET)*, pp. 66–72, Hammamet. Tunisia, January 2017.

[24] I. R. Sinclair, *Sensors and Transducers*Newnes, London, UK, 2001.

[25] R. Blom, *Non-Public Key Distribution*, Springer, Boston, MA, USA, 1983.

[26] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Proceedings of the Annual International Cryptology Conference*, Berlin, Heidelberg, Germany, August 1992.

[27] E. Khan, E. Gabidulin, B. Honary, and H. Ahmed, "Matrix-based memory efficient symmetric key generation and predistribution scheme for wireless sensor networks," *IET Wireless Sensor Systems*, vol. 2, no. 2, pp. 108–114, 2012.

[28] L. Harn and C.-F. Hsu, "Predistribution scheme for establishing group keys in wireless sensor networks," *IEEE Sensors Journal*, vol. 15, no. 9, pp. 5103–5108, 2015.

[29] S. Murthy, R. J. D'Souza, and G. Varaprasad, "Digital signature-based secure node disjoint multipath routing protocol for wireless sensor networks," *IEEE Sensors Journal*, vol. 12, no. 10, pp. 2941–2949, 2012.

[30] R. Azarderakhsh, K. U. Järvinen, and M. Mozaffari-Kermani, "Efficient algorithm and architecture for elliptic curve cryptography for extremely constrained secure applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 4, pp. 1144–1155, 2014.

[31] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the Proceedings of the 9th ACM Conference on Computer and Communications Security*, pp. 41–47, Washington, NJ, USA, November 2002.

[32] L. Harn, C. F. Hsu, O. Ruan, and M. Y. Zhang, "Novel design of secure end-to-end routing protocol in wireless sensor networks," *IEEE Sensors Journal*, vol. 16, no. 6, pp. 1779–1785, 2015.

[33] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 52–61, Washington, NJ, USA, October 2003.

[34] M. H. Park, Y. H. Park, H. Y. Jeong, and S. W. Seo, "Key management for multiple multicast groups in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 12, no. 9, pp. 1712–1723, 2012.

[35] T. Cooklev, Air Interface for Fixed Broadband Wireless Access Systems, 2004.

[36] Q. Wu, B. Qin, L. Zhang, J. Domingo-Ferrer, and O. Farràs, "Bridging broadcast encryption and group key agreement," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 143–160, Gold Coast, Australia, December 2011.

[37] V. S. Naresh, V. D. Allavarpu, and S. Reddi, "Blockchain privacy-preserving smart contract centric multiple multiparty key agreement over large WANETs," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 2, p. e4165, 2021.

[38] C. F. Hsu, L. Harn, and B. Zeng, "UMKESS: user-oriented multi-group key establishments using secret sharing," *Wireless Networks*, vol. 23, pp. 1–10, 2018.

[39] C.-F. Hsu, Q. Cheng, X. Tang, and B. Zeng, "An ideal multi-secret sharing scheme based on MSP," *Information Sciences*, vol. 181, no. 7, pp. 1403–1409, 2011.
[40] E. Karnin, J. Greene, and M. Hellman, "On secret sharing systems," *IEEE Transactions on Information Theory*, vol. 29, no. 1, pp. 35–41, 1983.
[41] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.