A Novel Threshold Conference-Key Agreement Protocol Based on Generalized Chinese Remainder Theorem

Cheng Guo[1] and Chin-Chen Chang[2,3]
(Corresponding author: Chin-Chen Chang)

School of Software Technology, Dalian University of Technology, China[1]
No. 8 Road, Jinzhou District, Dalian, 116620 China
Department of Information Engineering and Computer Science, Feng Chia University[2]
No. 100, Wenhwa Road, Seatwen, Taichung, Taiwan 40724, R.O.C.
Department of Computer Science and Information Engineering, Asia University[3]
500, Lioufeng Road, Wufeng, Taichung, Taiwan 41354, R.O.C.
(Email: alan3c@gmail.com)
(Received May 24, 2014; revised and accepted Nov. 25, 2014)

Abstract

The conference-key agreement protocol is a mechanism for generating a common session key among the authorized conference members. The common session key is used to encrypt communication messages transmitted over an open network. Inspired by traditional key agreement protocols and threshold cryptosystems, we have proposed a novel threshold conference-key agreement protocol in this paper. In the proposed protocol, we used a secret sharing scheme based on the generalized Chinese remainder theorem (GCRT) to achieve the threshold characteristic, and we can alter the shared data by adjusting an additional parameter k of the GCRT. If the number of conference members involved in generating the conference key exceeds a certain number, the members can cooperate to generate a valid common session key that also can be verified and used by other authorized conference members. Compared with traditional key agreement protocols, the proposed protocol has some unique characteristics that are beneficial in real applications.

Keywords: Conference-key agreement, generalized Chinese remainder theorem, threshold cryptosystem

1 Introduction

With the rapid development of Internet technology and its growing popularity, group-oriented applications and protocols have become increasingly important. A group of people can use the Internet to communicate at anytime and from their various locations instead of having to assemble in one location, thereby saving a lot of time and money. Because of the convenience of the network, web conferencing has become the trend of future development. A web conference can be held by connecting conference members located in different areas, even different continents, via the Internet. However, the network is an open environment, which means that it is vulnerable to a variety of attacks, such as masquerade attacks, replay attacks, and the modification of messages. Regardless of the underlying environments, communication privacy and integrity in the group are essential. A general and effective method for ensuring the confidentiality of the messages transmitted among participants is to establish a common session key for encrypting their communications over an insecure channel.

Conference-key establishment protocols can be classified roughly into two categories i.e., key agreement protocols [3, 7, 9, 10, 11, 12, 16, 20, 24, 25, 26, 27] and key distribution protocols [5, 6, 8, 13, 14, 21]. In a conference-key distribution protocol, a trusted third party, called a key generation center, is responsible for generating and distributing the conference key to authenticated conference members. Conference-key agreement is a mechanism in which a group of conference members computes a function $K = f(k_1, k_2, \cdots, k_n)$ securely, where k_i is a conference member's private input. There are some potential secure problems. First, conference member's private input must be transmitted in an open channel. That could allow some non-authorized attackers to obtain the conference key and listen to or observe the content of the communication. Second, attackers also can try to impersonate authenticated members. The basic technique for solving the above security issues is the utilization of public key cryptology, such as RSA, ElGamal, or ECC, to confirm the conference members' identities and to guarantee the

confidentiality of the function $K = f(k_1, k_2, \cdots, k_n)$.

It is well known that Diffie and Hellman (DH) [7] proposed a protocol that can establish a common key between two parties. Most group key-management protocols, including those developed by Ingemarsson et al. [9]; Burmester and Desmedt [3]; Steiner et al. [24]; and Just and Vaudenay [11], have attempted to extend the elegance and simplicity of Diffie-Hellman's two-party key exchange to the group setting. Just and Vaudenay [11] proposed an authenticated, multi-party, key-agreement protocol in which the group members could interact via an exchange of messages to obtain a common session key by using DH and public-key techniques without requiring a trusted third party. There has been intensive research on conference-key agreement protocols [10, 12, 16, 20, 25, 26, 27], such as the study of key agreement protocols in dynamic peer groups [16, 25].

In 1979, the first (t, n) threshold schemes, based on Lagrange interpolation and liner project geometry, were proposed by Shamir [23] and Blakley [2], respectively. Other well-known secret sharing schemes include Mignotte's scheme [18] and the Asmuth-Bloom scheme [1], which were based on the Chinese remainder theorem (CRT). With the emergence of different kinds of applications, threshold cryptosystems have been studied extensively, and threshold cryptosystems and their many variations form an important research direction.

In traditional conference-key agreement protocols, conference members u_i, for $i = 1, 2, \cdots, n$ who want to establish a secure channel for transferring confidential information must cooperate to compute a common conference key K, using each conference member's input k_i. No information about K can be obtained from a protocol run without knowledge of at least one of the k_i. Most existing conference-key agreement protocols have followed this pattern. However, in reality, there are many situations in which the ability to hold the conference is determined by a certain number of people. Once it has been decided that the conference will take place, all conference members must participate in the conference. Consider the following scenario: In a board meeting, the condition for holding the conference is that a certain number of members of the board of directors must agree that the meeting should proceed. Then, once it has been decided that the board meeting will occur, all members of the board of directors must participate in the meeting. However, to the best of our knowledge, traditional conference-key agreement protocols cannot meet this requirement. Existing conference-key agreement protocols do not have threshold characteristics, and other members cannot obtain the conference key or verify the validity of the conference key if they were not involved in the generation of the conference key. For the above reasons, we have focused on the threshold conference-key agreement in this paper.

We propose a novel threshold conference-key agreement protocol that has the following characteristics:

1) The conference key can be obtained if and only if at least t or more conference members cooperate to generate the conference key;

2) The other authorized members who do not participate in the generation of the conference key also can obtain the conference key;

3) The other authorized members can verify the validity of the conference key and communicate with other conference members using this common session key.

The remainder of the paper is organized as follows: In the next section, we describe our main objective. In Section 3, we introduce some preliminaries. In Section 4, we propose our threshold conference-key agreement protocol based on the generalized Chinese remainder theorem. The security and performance analysis are presented in Section 5. Finally, we present our conclusions in Section 6.

2 Objective

In this section, we describe the design principles of our threshold conference-key agreement protocol and then present the security requirements for the proposed protocol.

2.1 Design Principles

In the proposed scheme, same as for the existing conference-key agreement protocols, a common key can be reached among the conference members. However, there are some distinct characteristics in the proposed threshold conference-key agreement protocol. The idea of our design is to allow a certain number of conference members to decide whether the conference should be held and what the conference key will be. In the proposed scheme, one authorized conference member can initiate a request for a conference. Then, if a sufficient number of conference members agree and are willing to cooperate to generate the common conference key, the common key can be computed. But if the number of conference members who agree is less than the threshold value, they cannot construct a valid conference key.

Our protocol uses threshold Mignotte secret sharing and the generalized Chinese remainder theorem (GCRT) as building blocks. First, we utilize the Mignotte secret sharing scheme based on CRT to achieve the threshold access structure that can be used to obtain the conference key among the conference members. It is well known that GCRT is a variation of CRT, and in a similar manner, the threshold access structure based on CRT also can be achieved by using GCRT. In the Mignotte secret sharing scheme, secret data can be represented by a corresponding congruence system. However, if we want to modify the shared data, we have to readjust the corresponding congruence system. In conference-key agreement protocols, key independence is an important property. That

is, the previous conference keys are irrelevant to the subsequent conference keys. The conference key can be obtained by evaluating a function $f(k_1, k_2, \cdots, k_n)$, where $k_1, k_2, \cdots, k_n$ are the n conference members' private inputs. Therefore, the Mignotte secret sharing scheme based on the traditional CRT is inappropriate for this situation. In GCRT, an additional modulus k is provided to strengthen and enlarge the CRT's applications. We can change the shared data by adjusting the parameter k, which can be computed by collecting a certain number of conference members' inputs, k_i. In Section 3, we provide a brief introduction to the Mignotte secret sharing scheme and the generalized Chinese remainder theorem.

First, each conference member must register at the key generation center (KGC). During the registration process, the KGC shares the necessary information for key agreement with each authorized conference member. A conference may be requested by any authorized member. For example, if conference member U_i wants to convene a conference, he or she must participate in the key-generation process and send some messages in an authenticated broadcast channel. If enough conference members participate in the key-generation phase, the conference key can be computed. Meanwhile, all authorized members can obtain the conference key and verify its validity. Because we use GCRT, a new conference key can be generated by modifying an additional parameter k, which does not require any change in the information shared with the KGC by each member.

The proposed protocol also supports changes in the conference members, such as when members join or leave the group.

2.2 Design Principles

In this section, we summarize the desired security goals for our threshold conference-key agreement protocol. Referring to the protocol developed by Kim et al. and Harn et al. [8, 12], we define the following security properties:

1) Key authentication guarantees that the conference key can be generated by authorized conference members, but not by fraudulent attackers;

2) Key confidentiality guarantees that the conference key only can be obtained by authorized conference members;

3) Forward secrecy is that a compromise of the previous conference key cannot disclose the current key;

4) Backward secrecy is that an earlier key cannot be obtained if the current conference key is compromised.

In our protocol, we utilize the RSA cryptosystem [22] to protect the authenticity and confidentiality of the key as proposed by Zhao et al.'s protocol [26]. Concerning forward and backward secrecy, the characteristic of the GCRT can ensure the conference key's freshness and independence. That is, old keys and the current key cannot be derived from each other. We will give the details in Section 5.

3 Preliminaries

In this section, we briefly introduce the threshold Mignotte secret sharing scheme [18] and the generalized Chinese remainder theorem [4, 15], which are the major building blocks of our protocol.

3.1 Threshold Mignotte Secret Sharing Scheme

In 1982, Mignotte [18] proposed a threshold secret sharing scheme that uses some special sequences of integers, referred to as the Mignotte sequences.

Let n be a positive integer and $2 \leq t \leq n$. A (t, n)-Mignotte sequence is a sequence of positive integers $p_1 < p_2 < ... < p_n$, such that $\prod_{i=0}^{t-2} p_{n-i} < \prod_{i=1}^{t} p_i$, where $p_1, p_2, \cdots, p_n$ are co-prime in pairs.

Given a public (t, n)-Mignotte sequence, the scheme works as follows:

1) The secret y can be chosen as a random integer such that $b < y < a$, where $a = \prod_{i=1}^{t} p_i$ and $b = \prod_{i=0}^{t-2} p_{n-i}$;

2) The shadows y_i are computed such that $y_i = y \bmod p_i$, for all $1 \leq i \leq n$;

3) Given t distinct shadows $y_1, y_2, \cdots, y_t$ and t corresponding modulo $p_1, p_2, \cdots, p_t$, using the Chinese remainder theorem, the secret y can be recovered in the following congruence system:

$$\begin{cases} y \equiv y_1 \bmod p_1 \\ y \equiv y_2 \bmod p_2 \\ \vdots \\ y \equiv y_t \bmod p_t. \end{cases}$$

3.2 Generalized Chinese Remainder Theorem

GCRT [4, 15] is a variation of CRT. Similar to CRT, n positive co-prime integers $m_1, m_2, \cdots, m_n$ and $\{x_1, x_2, \cdots, x_n\}$ are needed to construct a system of simultaneous congruencies. In GCRT, an additional modulus k is required during the computations, where $\text{Max}\{x_i\}_{1 \leq i \leq n} < k < \text{Min}\{m_j\}_{1 \leq j \leq n}$. A number X can be represented by using $\{x_1, x_2, \cdots, x_n\}$, where $x_i = \lfloor X/m_i \rfloor \bmod k$,for $i = 1, 2, \cdots, n$. According to the

GCRT, the number X can be computed as follows:

$$\begin{aligned} X &= \sum_{i=1}^{n} M_i * M'_i * N_i (\mathrm{mod}(k * \prod_{i=1}^{n} m_i)), \quad \text{where} \\ M_i &= k * \prod_{j=1, j\neq i}^{n} m_j, \\ M_i * M'_i &= k(\mathrm{mod} k * m_i) \\ N_i &= \lceil x_i * m_i / k \rceil . \end{aligned}$$

4 The Proposed Protocol

In this section, we develop an extension to the existing conference-key agreement protocols. In the proposed protocol, the conference key can be computed if and only if the number of involved conference members who want to cooperate to generate a conference key exceeds a certain threshold. Our protocol consists of four phases: registration, sub-key distribution and commitment, sub-key recovery, conference-key derivation and verification. Table 1 is used notations throughout the remainder of this paper.

Table 1: The notations

n	number of conference members
t	threshold
U_i	i-th conference member, $i \in [1, n]$
ID_i	identity of the conference member U_i
SN_ℓ	unique serial number for the ℓth conference-key
k_i	sub-key generated by U_i
K	conference-key
$h()$	collision-free one-way hash function

4.1 Registration Phase

As described in Sections 3.1 and 3.2, we can construct a new secret-sharing scheme based on GCRT instead of CRT. First of all, KGC generates n positive coprime numbers $m_1, m_2, \cdots, m_n$ that satisfy the Mignotte sequence characteristics and selects n positive integers $x_1, x_2, \cdots, x_n$. KGC randomly selects a number k that satisfies $\mathrm{Max}\{x_i\}_{1\leq i\leq n} < k < \mathrm{Min}\{m_j\}_{1\leq j\leq n}$. The initial value of the conference key K can be represented by n-tuple $\{x_1, x_2, \cdots, x_n\}$ and $k\prod_{i=0}^{t-2} m_{n-i} < K < k\prod_{i=1}^{t} m_i$.

We assume that there are n conference members and that each conference member is required to register at KGC before the conference-key agreement. Learning from Zhao et al.'s group key-agreement protocol [26], we also utilize the RSA public-key cryptosystem [22] to guarantee the authentication, confidentiality, and integrity of the sub-key.

Upon receiving a registration request from any conference member U_i, KGC will perform the following steps:

1) According to the properties of the RSA cryptosystem, KGC computes $N_i = p_i \times q_i$, where p_i and q_i are two large prime numbers selected randomly, and factoring N_i is hard;
2) KGC generates the corresponding private key d_i and public key e_i that satisfy $d_i \times e_i \equiv \varphi(N_i)$, where $\varphi(N_i)$ is the Euler phi-function;
3) KGC distributes the secret value pair (x_i, m_i) and private key d_i to conference member U_i over a secure channel;
4) KGC publishes the public key $\{e_i, N_i\}$.

4.2 Sub-key Distribution and Commitment Phase

The protocol starts with an initiator initiating a conference request for a set $U = \{U_1, U_2, \cdots, U_n\}$ of conference members and the initiator broadcasts a unique serial number SN_ℓ for the conference. If any member U_i wants to accept the conference request, he will perform the following steps:

1) U_i randomly selects its sub-key k_i, $\mathrm{Max}\{x_i\}_{1\leq i\leq n} < k_i < \mathrm{Min}\{m_j\}_{1\leq j\leq n}$;
2) Compute $\mu_i = (x_i||m_i||k_i||SN_\ell)^{d_i} \bmod N_i$;
3) Compute $\sigma_{ij} = (\mu_i||ID_i)^{e_j} \bmod N_j$ for $U_j (i \neq j)$ and $h_i = h(x_i||m_i||k_i||SN_\ell)$;
4) Publish $\omega_i = \{h_i, \sigma_{ij}, \mathrm{for} j = 1, 2, ..., n, j \neq i\}$.

4.3 Sub-key Recovery Phase

Without losing generality, all conference members can receive ω_i from U_i for $i = 1, 2, \cdots, t$. Then, they can compute the corresponding sub-key k_i and check the validity of the sub-key. The details are as follows:

1) U_j computes $\mu'_i||ID_i = (\sigma_{ij})^{d_j} \bmod N_j$ using her or his private key d_j and reads μ'_i and ID_i. Then, he or she uses the corresponding public key e_i to compute $(x_i||m_i||k_i||SN_\ell)' = (\mu'_i)^{e_i} \bmod N_i$;
2) U_j checks whether or not SN'_ℓ is the current conference serial number SN_ℓ and computes $h'_i = h(x'_i||m'_i||k'_i||SN'_\ell)$ and verifies the equation $h'_i = h_i$;
3) If the equation holds, then U_j can retrieve the corresponding $\{x_i, m_i, k_i\}$.

4.4 Conference-key Derivation and Verification Phase

If a sufficient number of conference members respond to the conference request, the conference key can be derived from sufficient sub-key information $\{x_i, m_i, k_i\}$ for $i = 1, 2, \cdots, t$. The procedure consists of two phases: (1) the conference-key derivation phase and (2) the conference-key verification phase.

(1) Conference-key derivation phase

1) Once U_j obtains t verified sub-key information ω_i, U_j can first compute $k = \left\lfloor \frac{\sum_{i=1}^{t} k_i}{t} \right\rfloor$ without losing generality;
2) According to the GCRT and Mignotte secret sharing, the conference key K can be computed by each conference member U_j as follows:

$$K = \sum_{i=1}^{t} M_i * M'_i * N_i (\mathrm{mod}(k * \prod_{i=1}^{t} m_i)), \quad (1)$$

where

$$\begin{aligned} M_i &= k * \prod_{j=1, j\neq i}^{t} m_j, \\ M_i * M'_i &= k(\mathrm{mod} k * m_i) \\ N_i &= \lceil x_i * m_i / k \rceil . \end{aligned}$$

(2) Conference-key verification phase

As mentioned above, all authorized conference members can compute the conference key, even though they do not participate in the generation of the conference key. Meanwhile, they also have the capacity to verify the validity of the conference key. Each authorized member U_j can use her or his (x_j, m_j) to check the following equation:

$$x_j \stackrel{?}{\equiv} \left\lfloor K/m_j \right\rfloor \bmod k \quad (2)$$

If Equation (2) holds, the conference-key K is validated. This shows that at least t authorized conference members agree to convene the network conference. That is, all conference members must participate in this conference, and all conference members can share a common secret conference key used to encrypt the conference message.

4.5 Conference Members Join and Leave

Concerning a conference group, new members may join such a group, thereby increasing the number of members; also, some members may leave the group, thereby decreasing its number of members. Therefore, the proposed scheme must support these potential occurrences.

If a new member U_j wants to join the conference group, he or she must register at the KGC. The KGC selects a corresponding m_j that meets the characteristic of the Mignotte sequence. According to the current K and k, KGC can compute $x_j = \lfloor K/m_j \rfloor \bmod k$ and send (x_j, m_j) and the private key d_j to U_j over a secure channel. Then, KGC publishes the corresponding public key $\{e_j, N_j\}$.

If a conference group member U_j leaves the group, KGC will broadcast the information about U_j's leaving the conference group. Then, in the sub-key distribution and commitment phase, the conference members will not use U_j's public key to encrypt the sub-key information.

4.6 A New Conference-key is Computed

If a new conference-key is required, we need to change the shared the secret number aiming at modifying the shared conference-key. If we utilize Shamir's secret sharing based on a Lagrange interpolation polynomial or Mignotte's secret sharing based on CRT to construct a threshold protocol, then, if we want to change the shared number, we have to modify the corresponding interpolation polynomial or the corresponding congruence system. That is, we have to update the data stored by each authorized conference member. That means that the registration phase has to be replayed and all conference members have to obtain a new secret shadow. However, using the GCRT, we can simply solve the problem instead of changing the interpolation polynomial or the whole congruence system. We can change the shared number by adjusting the parameter k. Therefore, the conference members can execute the rest of the protocol except for the registration phase to compute a new conference-key.

5 Discussions

In this section, we present the security and performance analysis of our protocol.

5.1 Security Analysis

As discussed in Section 2.2, the main security requirements of a threshold conference-key agreement are: conference-key authentication, conference-key confidentiality, and forward/backward secrecy. In this section, we show that the proposed protocol satisfies those four security requirements and can resist some types of attacks.

5.1.1 Conference-key Authentication

In the process of the generating the conference key, each conference member U_i computes her or his sub-key information and encrypts the information using her or his private key d_i. Meanwhile U_i computes $h_i = h(x_i||m_i||k_i||SN_\ell)$. The other members can decrypt this message using U_i's public key and derive the corresponding sub-key information $(x_i||m_i||k_i||SN_\ell)'$. So, the other members can confirm that this message was sent by U_i. And, the other members can compute h'_i and check whether $h'_i = h_i$ in order to verify the integrity of the sub-key information.

5.1.2 Conference-key Confidentiality

Before considering conference-key confidentiality, we briefly illustrate key freshness. The conference key K can

be computed as follows:

$$K = \sum_{i=1}^{t} M_i * M'_i * N_i (\mathrm{mod}(k * \prod_{i=1}^{t} m_i)), \quad \text{where}$$
$$k = \left\lfloor \frac{\sum_{i=1}^{t} k_i}{t} \right\rfloor.$$

Every conference key is fresh since each conference member selects a new random k_i for the generation of each conference key. In our protocol, we use the GCRT to achieve the threshold access structure instead of Shamir's secret sharing. As we shall see, our protocol is somewhat sensitive to selected parameters [17, 19]. However, it is information theoretically secure if the parameters for GCRT are selected appropriately.

As discussed in Sections 4.2 and 4.3, each conference member U_i encrypts the sub-key information using the other conference members' public keys e_j for $j = 1, 2, \cdots, n, i \neq j$. So, unauthorized members cannot compute the corresponding sub-key information, so they cannot obtain the conference key.

5.1.3 Forward/Backward Secrecy

First, we consider forward secrecy, which states that 1) if an attacker obtains the previous conference keys, he or she cannot derive the current conference key, and 2) if a member leaves the conference group, he or she also cannot compute the subsequent conference keys. During the computation of the conference key K, an additional modulus k is required. The parameter k is computed as follows:
$k = \left\lfloor \frac{\sum_{i=1}^{t} k_i}{t} \right\rfloor$, where k_i is conference member U_i's private input.

For each conference key, conference member U_i must randomly select a number k_i that satisfies $\mathrm{Max}\{x_i\}_{1 \leq i \leq n} < k_i < \mathrm{Min}\{m_j\}_{1 \leq j \leq n}$. And, each k_i is independent. Therefore, the previous conference keys and the current conference key are irrelevant. Even though an adversary may know a series of previous conference keys, he or she cannot compute the current conference key.

Then, we consider backward secrecy, meaning that if an attacker obtains the current conference key, he or she cannot derive the previous conference keys, and, if a member joins the conference group, he or she also cannot derive any previous conference keys. As mentioned in the previous paragraph concerning the generation of each conference key, conference member U_i randomly re-selects a new k_i. Then, all conference keys K are independent of each other. So, an adversary who knows the current conference key cannot derive the previous conference keys.

5.1.4 Resistance to Some Types of Attacks

In this section, we show that our protocol is secure against the following attacks:

1) Impersonation attack

The main active attack to a conference-key agreement protocol is the impersonation attack, in which an unauthorized adversary impersonates a legal conference member to take part in a conference. In the registration phase, KGC generates the corresponding private key d_i and public key e_i published to all conference members for each conference member U_i. And, in the sub-key distribution and commitment phase, U_i must use her or his private key d_i to encrypt the sub-key information. Therefore, other members can utilize the corresponding public key e_i to check whether the sub-key information is from the alleged member.

2) Forgery attack

We divide forgery attacks into two categories. First, a malicious member can not forge other members' sub-key information since he or she does not have the other members' private keys. Meanwhile, conference members also can verify whether the sub-key information received has been tampered with by checking the equation $h'_i = h_i$, where $h'_i = h(x'_i||m'_i||k'_i||SN'_\ell)$ and $h_i = h(x_i||m_i||k_i||SN_\ell)$. On the other hand, as mentioned in Section 2.1, it takes a sufficient number of conference members to cooperate to generate a valid conference key. We assume that a certain number of compromised members want to forge the conference key. According to threshold secret sharing, if the number of compromised members involved is less than the threshold, the members cannot generate a valid conference key. And, the other authorized members can use their secret value pair (x_i, m_i) to verify the validity of the conference key. Therefore, the conference key cannot be forged easily unless the number of compromised members exceeds the threshold.

3) Replay attack

A group of conference members run a threshold conference-key protocol over an open network. Even though their communications are encrypted with the corresponding public key cryptosystem, an inside attacker or an outside attacker also can intercept these data and retransmit them. But, in the proposed protocol, the initiator first broadcasts a unique serial number SN_ℓ, and the other conference members involved embed into the sub-key information. The conference members that receive the sub-key information $\omega_i = \{h_i, \sigma_{ij}, \mathrm{for} j = 1, 2, ..., n, j \neq i\}$ can check to determine whether the serial number SN_ℓ is the current serial number. So, the adversary cannot resend the previous key-agreement message.

5.2 Performance Analysis

In this section, we analyzed the computation costs for the proposed protocol. In our protocol, if the number of conference members involved in the generation of a

Table 2: Comparison of computational costs

	Tzeng's protocol	**Hung et al's protocol**	**Zhao et al's protocol**	**Our protocol**
Registration	nT_e	nT_e	0	0
Sub-key distribution and commitment phase	$(n+2)\,T_e + T_{inv} + 3T_{mul} + T_h$	$(n+2)\,T_e + T_h + (n+2)\,T_{mul}$	$nT_e + T_h$	$nT_e + T_h$
Sub-key recovery phase	$4nT_e + nT_{mul}$	$4nT_e + nT_{mul} + nT_h$	$2\,(n-1)\,T_e + (n-1)\,T_h$	$2\,(t-1)\,T_e + (t-1)\,T_h$
Conference-key derivation and verification phase	0	0	0	$tT_{inv} + 2\,(t-1)\,T_{mul}$

conference key exceeds a certain number, a common session key can be achieved. And, all authorized conference member can obtain the common session key and can verify its validity. We assume that the threshold is t. During the sub-key distribution and commitment phase, each member U_i must utilize public-key cryptosystems to confirm sub-key's authentication, confidentiality, and integrity. For facilitating the performance evaluation of our scheme, we first denote the following notations:

1) T_h:the time for performing a hash function h;

2) T_{inv}:the time for performing a modular inverse computation;

3) T_{mul}:the time for performing a modular multiplication computation;

4) T_e:the time for performing a modular exponentiation computation.

Table 2 compares the computational costs of our protocol with those of Tzeng [20], Hung et al. [10], and Zhao et al. [26].

As can be seen in Table 2, since we utilize the RSA cryptosystem applied in Zhao et al.'s protocol to confirm the authentication, confidentiality, and integrity of the sub-key, the computational costs of our protocol is the same as Zhao et al.'s protocol in the sub-key distribution and commitment phase. In the sub-key recovery phase, our protocol requires fewer modular exponentiations than Zhao et al.'s protocol since our protocol is a threshold scheme. However, in the conference-key derivation and verification phase, our protocol requires additional modular inverse computation and modular multiplication computation to reconstruct the conference key. The result is that the total computational costs of our protocol are less than those associated with Tzeng's protocol and Huang et al.'s protocol, and they are about the same as those associated with Zhao et al.'s protocol. In addition, we need to point out that the conference-key can be recovered by collecting t value pairs and that the conference-key recovery by the usual Lagrange interpolation method requires $O(t\log^2 t)$ operations, while the GCRT method of the proposed protocol requires only $O(t)$ operations.

6 Conclusions

In this paper, we propose a novel, threshold conference-key agreement protocol based on the generalized Chinese remainder theorem. In our protocol, the conference key can be generated only if the number of conference members involved exceeds a certain threshold, and other authorized conference members, who were not involved in the generation of the conference key, also can compute the conference key and check its validity. The unique threshold characteristic of our conference-key agreement protocol can fill the gaps in many applications.

The security analysis shows that our protocol is resistant to impersonation attack, forgery attack, and replay attack and that it also has some important security features, such as conference-key authentication, conference-key confidentiality, and forward/backward secrecy. The performance analysis shows that the computational cost of our protocol is satisfactory.

Acknowledgments

This paper is supported by the National Science Foundation of China under grant No. 61272173, 61100194, 61401060 and The general program of Liaoning Provincial Department of Education Science Research under grants L2014017.

References

[1] C. Asmuth and J. Bloom, "A modular approach to key safeguarding," *IEEE Transactions on Information Theory, IT-29 (2)*, pp. 208–210, 1983.

[2] G. Blakley, "Safeguarding cryptographic keys," in *The National Computer Conference*, pp. 313–317, Montvale:NCC, 1979.

[3] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system," in *EUROCRYPT 1994, LNCS 0950*, pp. 275– 286, Springer, Berlin, 1995.
[4] C.C. Chang and H.C. Lee, "A new generalized group-oriented cryptoscheme without trusted centers," *IEEE Journal on Selected Areas in Communications 11 (5)*, pp. 725–729, 1993.
[5] C.C. Chang, T.C. Wu, and C.P. Chen, "The design of a conference key distribution system," in *AUSCRYPT 1992, LNCS 0718*, pp. 459–466, Springer, Berlin, 1993.
[6] V. Daza, J. Herranz, and G. Sáez, "On the computational security of a distributed key distribution scheme," *IEEE Transactions on Computers 57 (8)*, pp. 1087–1097, 2008.
[7] W. Diffie and M.E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory IT-22 (6)*, pp. 644–654, 1976.
[8] L. Harn and C.L. Lin, "Authenticated group key transfer protocol based on secret sharing," *IEEE Transactions on Computers 59 (6)*, pp. 842–846, 2010.
[9] I. Ingemarsson, D.T. Tang, and C.K. Wong, "A conference key distribution system," *A conference key distribution system, IEEE Transactions on Information Theory IT-28 (5)*, pp. 714–720, 1982.
[10] Q. Jiang, J.F. Ma, G.S. Li, and et al., "Robust two-factor authentication and key agreement preserving user privacy," *International Journal of Network Security*, vol. 16, no. 3, pp. 229–240, 2014.
[11] M. Just and S. Vaudenay, "Authenticated multi-party key agreement," in *ASIACRYPT 1996, LNCS 1163*, pp. 36–49, Springer, Berlin, 1996.
[12] Y. Kim, A. Perrig, and G. Tsudik, "Group key agreement efficient in communications," *IEEE Transactions on Computers 53 (7)*, pp. 905–921, 2004.
[13] K. Koyama, "Secure conference key distribution schemes for conspiracy attack," in *EUROCRYPT 1992, LNCS 0658*, pp. 449–453, Springer, Berlin, 1993.
[14] S.S. Kulkarni and B. Bruhadeshwar, "Key-update distribution in secure group communication," *Computer Communications 33 (6)*, pp. 689–705, 2010.
[15] Y.P. Lai and C.C. Chang, "Parallel computational algorithms for generalized chinese remainder theorem," *Computers and Electrical Engineering 29 (8)*, pp. 801– 811, 2003.
[16] P.P.C. Lee, J.C.S. Lui, and D.K.Y. Yau, "Distributed collaborative key agreement and authentication protocols for dynamic peer groups," *IEEE/ACM Transactions on Networking 14 (2)*, pp. 263–276, 2006.
[17] Quisquater M, Preneel B, and Vandewalle J, "On the security of the threshold scheme based on the chinese remainder theorem," in *Proc. PKC 2002, LNCS 2274*, pp. 199–210, 2002.
[18] M. Mignotte, "How to share a secret," in *EUROCRYPT 1982, LNCS 149*, pp. 371–375, Springer, Berlin, 1982.
[19] Goldreich O, Ron D, and Sudan M, "Chinese remaindering with errors," *IEEE Transactions on Information Theory IT-46*, pp. 1330–1338, 2000.
[20] Y.K. Peker, "A new key agreement scheme based on the triple decomposition problem," *International Journal of Network Security*, vol. 16, no. 6, pp. 426–436, 2014.
[21] A. Perrig, D. Song, J.D. Tygar, and Elk, "A new protocol for efficient large group key distribution," in *IEEE Symposium Security and Privacy*, pp. 247–262, 2001.
[22] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM 21 (2)*, pp. 120– 126, 1978.
[23] A. Shamir, "How to share a secret," *Communications of the ACM 22 (11)*, pp. 612–613, 1979.
[24] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-hellman key distribution extended to group communication," in *Third ACM Conference Computer and Communications Security*, pp. 31–37, New Delhi, India, 1996.
[25] M. Steiner, G. Tsudik, and M. Waidner, "Key agreement in dynamic peer groups," *IEEE Transactions on Parallel and Distributed Systems 11 (8)*, pp. 769–780, 2000.
[26] J.J. Zhao, D.W. Gu, and Y.L. Li, "An efficient fault-tolerant group key agreement protocol," *Computer Communications 33 (7)*, pp. 890–895, 2010.
[27] B.Q. Zhou, S.J. Li, J.X. Wang, and et al., "A pairwise key establishment scheme for multiple deployment sensor networks," *International Journal of Network Security*, vol. 16, no. 3, pp. 221–228, 2014.

Cheng Guo received the B.S. degree in computer science from Xian University of Architecture and Technology in 2002. He received the M.S. degree in 2006 and his Ph.D in computer application and technology, in 2009, both from the Dalian University of Technology, Dalian, China. From July 2010 to July 2012, he was a post doc in the Department of Computer Science at the National TsingHua University, Hsinchu, Taiwan. Since 2013, he has been an associate professor in the School of Software at the Dalian University of Technology. His current research interests include information security and cryptology

Chin-Chen Chang received his B.S. degree in applied mathematics in 1977 and the M.S. degree in computer and decision sciences in 1979, both from the National Tsing Hua University, Hsinchu, Taiwan. He received his Ph.D in computer engineering in 1982 from the National Chiao Tung University, Hsinchu, Taiwan. From 1983-1989, he was on the faculty of the Institute of Applied Mathematics, National Chung Hsing University, Taichung, Taiwan. From August 1989 to July 1992, he was the head of, and a professor in, the Institute of Computer Science and Information Engineering at the National Chung Cheng University, Chiayi, Taiwan.

From August 1992 to July 1995, he was the dean of the college of Engineering at the same university. From August 1995 to October 1997, he was the provost at the National Chung Cheng University. From September 1996 to October 1997, Dr. Chang was the Acting President at the National Chung Cheng University. From July 1998 to June 2000, he was the director of Advisory Office of the Ministry of Education of the R.O.C. From 2002 to 2005, he was a Chair Professor of National Chung Cheng University. Since February 2005, he has been a Chair Professor of Feng Chia University. In addition, he has served as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, image compression and data structures. He is a fellow of the IEEE.