INVITED PAPER Special Section on Information and Communication System Security

# Secure Key Transfer Protocol Based on Secret Sharing for Group Communications

**Chia-Yin LEE**[†], **Zhi-Hui WANG**[††], **Lein HARN**[†††], ***and*** **Chin-Chen CHANG**[††††a)], ***Nonmembers***

**SUMMARY** Group key establishment is an important mechanism to construct a common session key for group communications. Conventional group key establishment protocols use an on-line trusted key generation center (KGC) to transfer the group key for each participant in each session. However, this approach requires that a trusted server be set up, and it incurs communication overhead costs. In this article, we address some security problems and drawbacks associated with existing group key establishment protocols. Besides, we use the concept of secret sharing scheme to propose a secure key transfer protocol to exclude impersonators from accessing the group communication. Our protocol can resist potential attacks and also reduce the overhead of system implementation. In addition, comparisons of the security analysis and functionality of our proposed protocol with some recent protocols are included in this article.

***key words:*** *key transfer protocol, group key, Diffie-Hellman key agreement, secret sharing*

## 1. Introduction

With the development of computer and network technologies, network communications have become a part for many people's daily lives. It is well known that data confidentiality is one of the most important issues for secure communications. To prevent an adversary from gaining access to the sensitive content of communications, a session key can be used for encryption/decryption. Therefore, before exchanging communication messages, a key establishment protocol must be used to construct the session keys for legitimate participants in the communication. As we know, the Diffie-Hellman (DH) key agreement protocol [1] is the most commonly used protocol for constructing a common session key between two parties. Since the public-key itself does not provide the property of authentication, a digital signature [2] or certificate [3] can be attached to the public-key to ensure authentication. However, the DH key agreement protocol is not suitable for a group communication, such as an e-conference, e-learning, and multi-user games, which has more than two participants. Therefore, a group key establishment protocol is needed for group communications. In general, group key establishment protocols can be classified into two major types. In the first type, a trusted third party, such as a group key generation center (KGC), generates the common session key and assigns the key to all group members. In the second type, the common session key is generated by group members directly without any third party joining. In addition, the protocols of the second type can be subdivided into group key transfer protocols and group key agreement protocols. The group key transfer protocol is that an *initiator* (a chairperson) demands to organize a group communication, and then he/she selects a group key and distributes the key to the other participants. On the other hand, the group key agreement protocol is that all participants together compute a common session key for the group communication. Even though group key agreement protocols have more flexibility to generate the group key, these protocols usually have heavy communication cost to construct the group key.

In 1995, Klein et al. [4] first proposed group key agreement protocol with fault-tolerance. The goal of fault-tolerance is to exclude malicious participants from the group. In 2002, Tzeng [5] pointed out that Klein et al.'s protocol is quite inefficient and its security is not rigidly proven. Thus, Tzeng proposed a secure fault-tolerant group key agreement protocol to overcome those drawbacks. In 2009, Huang et al. [6] proposed an enhanced group key agreement protocol based on Discrete Logarithm Problem (DLP) and they claimed that their protocol is more efficient in terms of computation and communication. In 2010, Zhao et al. [7] proposed a group key agreement protocol based on RSA cryptosystem [8] to improve the performance of Huang et al.'s protocol.

On the other hand, secret sharing has been used to design group key distribution protocols in recent years. There are two different methods to implement secret sharing scheme, i.e., the first method assumes that an off-line trusted server is involved only at initialization [9], [10], and the second method assumes that an on-line trusted server, such as KGC, is involved in all processes [11]. In 1991, Berkovits [12] employed the same concept of [11] to propose scheme to distribute group messages. This scheme can be adopted to transfer the session key for group members. However, Harn and Lin [13] indicated that this kind of method might suffer from the insider attack. Thus, they used modulus $N$, a composite integer, to prevent this kind

Manuscript received May 2, 2011.

[†]The author is with the Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi 62102, Taiwan.

[††]The author is with the School of Software, Dalian University of Technology, Dalian, Liaoning 116024, China.

[†††]The author is with the Department of Computer Science Electrical Engineering, University of Missouri-Kansas City, MO 64110, USA.

[††††]The author is with the Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan.

a) E-mail: ccc@cs.ccu.edu.tw

DOI: 10.1587/transinf.E94.D.2069

of attack. Unfortunately, the on-line KGC is required in distributing the group key; therefore it increases the overhead of the system.

In this article, we adopt the advantages of the DH key agreement and the secret sharing to design an efficient key transfer protocol that is secure. The rest of this article is organized as follows. In Sect. 2, we review some group key establishment protocols and then discuss some drawbacks. In Sect. 3, we illustrate the design concept of the improved scheme and a practical design example is proposed. Some security issues and required functionalities regarding group key establishment protocols are discussed in Sect. 4. Finally, some conclusions are summarized in Sect. 5.

## 2. Survey of Group Key Establishment Protocols

Since conventional group key establishment protocols use an on-line trusted third party, such as the KGC, to transfer the session key for each participant, it may increase the overhead of the system and lose the flexibility of the group key. Some group key establishment protocols without on-line KGC have been proposed to overcome those drawbacks in recent years.

In 1996, Steiner et al. [14] proposed a key agreement protocol based on the natural extension of the DH key agreement protocol for the group communication. In 2001, Steiner's protocol has been enhanced with the property of authentication by Bresson et al. [15]. In 2006, Bohli [16] proposed a framework for robust group key agreement that provides security against malicious insiders and active adversaries in public point-to-point network. However, most DH based group key agreement protocols do not scale well and, in particular, require $O(n)$ rounds. In 2007, Katz and Yung [17] proposed the first constant-round and fully scalable group key exchange protocol to reduce the message transmission overhead.

There are other group key establishment protocols based on non-DH key agreement approach. In 2002, Tzeng proposed a secure group key agreement protocol with fault-tolerant. With this ability, the protocol can detect malicious participants and prevent them from joining the group communications. However, each participant needs to maintain $n$ $n$-degree polynomials, where the parameter $n$ depends on the number of participants. In fact, this is a serious problem to system overhead. In 2007, Tseng [18] demonstrated that Tzeng's protocol does not provide forward secrecy and then proposed a secure group key agreement protocol based on the decisional Diffie-Hellman (DDH) problem. In 2009, Huang et al. proposed a group key protocol based on DLP to enhance the performance of Tzeng's scheme. In 2010, Zhao et al. proposed a similar protocol based on RSA cryptosystem and also claim that their scheme can achieve the ability of fault-tolerant. In this article, we demonstrate that Zhao et al.'s protocol cannot exclude adversaries from the group completely. Also, Zhao et al.'s protocol depends on unicasting to distribute the sub-key for each group member, this means that heavy communication costs for messages transmission.

Secret sharing schemes have been used to establish the group key in recent years. Unfortunately, most group key establishment protocols based on secret sharing may suffer from the insider attack. In 2010, Harn and Lin proposed a secure group key transfer protocol based on Shamir's $(t, n)$ secret sharing [19], denoted as $(t, n)$-SS. They also modified Shamir's $(t, n)$-SS as modulus $N$, a composite integer, to withstand the insider attack.

In this section, we first summarize the characteristics of some existing group key establishment schemes in Table 1. Next, we review three recent schemes, i.e., Huang et al.'s scheme, Zhao et al.'s scheme, and Harn and Lin's scheme, respectively, and then indicate some potential drawbacks.

### 2.1 Huang et al.'s Scheme

Huang et al.'s scheme is composed of five phases, i.e., parameter generation, secret distribution and commitment, sub-key computation and verification, fault detection, and session key computation. The detailed processes are illustrated as follows.

• *Parameter generation*

All the group members must register with the trusted server to obtain their public-key and private-key. For each registered member $U_i$, for $i = 1$ to $n$, where $n$ is the total numbers of the group members, the server performs the following processes:

1) Selects a large prime $p$ comprised of $2q + 1$, where $q$ is

**Table 1** Characteristics of some group key establishment schemes.

| Schemes | Steiner et al.'s (1996) [14] | Bresson et al.'s (2001) [15] | Tzeng's (2002) [5] | Bohli's (2006) [16] | Katz and Yung's (2007) [17] | Tseng's (2007) [18] | Huang et al.'s (2009) [6] | Zhao et al.'s (2009) [7] | Harn and Lin's (2010) [13] |
|---|---|---|---|---|---|---|---|---|---|
| Security principle | CDH assumption | DDH + CDH assumptions | DLP + Polynomial interpolation | Signature + Session identifiers | DDH assumption | DDH assumption | DLP | Factorization problem | Shamir's $(t, n)$-SS + Factorization problem |
| Advantages | Natural extensions of DH key agreement for $n$-party | Authenticated key exchange / Mutual authentication | Without an on-line KGC / Fault-tolerance | Prevent insider and outsider attacks | Constant-round and fully scalable | Without an on-line KGC / Faul-tolerance | Without an on-line KGC / Faul-tolerance | Without an on-line KGC | Efficient |
| Drawbacks | High computation and communication costs | High communication cost | High system's overhead / Do not provide forward secrecy | Session identifiers must be pre-shared | High computation cost | High computation and communication costs | High communication cost | High communication cost / Cannot exclude adversaries completely | The on-line KGC is required / Secrets must be pre-shared |

also a large prime.

2) Selects a generator $g$ of order $q$ over $GF(p)$.

3) Selects the private-key $x_i$ as $x_i \in \mathbb{Z}_q^*$ and then compute the corresponding public-key $y_i$ as $y_i = g^{x_i} \bmod p$.

4) Delivers $(x_i, y_i)$ to $U_i$ through a secure channel and publishes the values $(p, q, g, h(\cdot))$, where $h(\cdot)$ is a one-way hash function.

• *Secret Distribution and Commitment*

Each participant $U_i$ executes the following processes to distribute his/her temporary sub-key to the other participants:

1) Selects a random integer $a_i \in \mathbb{Z}_q^*$ and then computes $k_{ij} = y_j^{a_i} \bmod p \bmod q$, for $1 \le j \le n$.

2) Selects a line $L(x)$ randomly such that $L(x) = c_i x + CK_i \bmod q$, where $c_i = g^{a_i} \bmod p$.

3) Computes two parameters $d_{ij}$ and $d'_{ij}$ as below:

$$d_{ij} = L(k_{ij}) \bmod q, \text{ for } 1 \le j \le n \,,$$
$$d'_{ij} = k_{ij} \oplus d_{ij}, \text{ for } 1 \le j \le n \,.$$

4) Randomly selects an integer $r_i \in \mathbb{Z}_q^*$ and generates the individual digital signature $(R_i, S_i)$ on $CK_i$ as following terms:

$$R_i = g^{r_i} \bmod p \,,$$
$$S_i = x_i h(CK_i \| T) + r_i R_i \bmod q,$$

where $T$ is the current timestamp.

5) Broadcasts the message

$$M_i = (T, R_i, S_i, c_i, d'_{i1}, d'_{i2}, \cdots, d'_{i(i-1)}, d'_{i(i+1)}, \cdots, d'_{in}).$$

• *Sub-key Computation and Verification*

After receiving $M_i$ from $U_i$, each participant $U_j$ $(j \ne i)$ performs the following processes:

1) Checks whether the timestamp $T$ is valid. If $T$ is valid, $U_j$ continues the next process. Otherwise, terminate the sub-key computation and verification phase.

2) Computes the common session key $k_{ij}$ with the other participants $U_i$ by computing

$$k_{ij} = c_i^{x_j} \bmod p \bmod q, \text{ for } 1 \le i \le n.$$

3) Computes $d_{ij} = d'_{ij} \oplus k_{ij}$, for $1 \le i \le n$, and recovers the sub-key $CK_i$ by computing

$$CK_i = d_{ij} - c_i k_{ij} \bmod q, \text{ for } 1 \le i \le n.$$

4) Checks whether the signature of $CK_i$ is correct by using the following verification equation:

$$g^{S_i} \bmod p \stackrel{?}{=} y_i^{h(CK_i \| T)} R_i^{R_i} \bmod p, \text{ for } 1 \le i \le n.$$

5) If the above equation is satisfied, broadcasts $v_{ji} = success$ or else, broadcasts $v_{ji} = failure$ .

• *Fault Detection*

Each participant can detect faults by using the following processes:

1) When receiving $v_{ji} = failure$ for $U_j$, $U_j$ claims that $U_i$ is faulty. $U_i$ secretly exposes the self-retained value $a_i$ and the sub-key $CK_i$ to all other participants.

2) When receiving $v_{jm} = failure$, $U_j$ claims that $U_m$ $(m \ne i)$ is faulty, and following procedure is executed:

   (a) Wait for the fault detection messages $a_m$ and $CK_m$ from $U_m$.

   (b) If no fault detection messages are received from $U_m$, the $U_m$ must be the malicious member.

   (c) On receiving $a_m$ and $CK_m$, check whether $R_m$, $S_m$, $c_m$ , and $d'_{mj}$ are correct:

      (i) Check $c_m \stackrel{?}{=} g^{a_m} \bmod p$.

      (ii) Use $a_m$ and $CK_m$ to verify the term $d'_{mj}$.

      (iii) Verify whether the signature $(R_m, S_m)$ of $CK_m$ is valid. If the signature is valid, set $U_j$ as a malicious participant, otherwise, set $U_m$ as the malicious one.

3) The malicious participant is removed from the group by the other honest participants and the protocol is restarted.

• *Conference Key Computation*

When the previous phase is executed until no more faults are detected, each honest member of the set $U' = \{U'_1, U'_2, \cdots, U'_m\}$ can compute the conference key $CK$ as $CK = (CK'_1 + CK'_2 + \cdots + CK'_m) \bmod q$.

However, each participant $U_i$ must publish $M_i = (T, R_i, S_i, c_i, d'_{i1}, d'_{i2}, \cdots, d'_{i(i-1)}, d'_{i(i+1)}, \cdots, d'_{in})$ in the secret distribution and commitment phase. Actually, unicasting must be used for publishing the term $d'_{ij}$, for $j = 1, \cdots, (i-1), (i+1), \cdots, n$, to the corresponding participant. In other words, $n \times (n-1)$ messages must be sent to all group members in order to publish $d'_{ij}$ $(j \ne i)$. Thus, the communication cost would be increased.

### 2.2 Zhao et al.'s Scheme

Similar to Huang et al.'s scheme, Zhao et al.'s scheme is also composed of five phases. The detailed processes are described below.

• *Registration*

All the group members must register with the trusted server to obtain their public-key and private-key. For each registered member $U_i$, for $i = 1$ to $n$, where $n$ is the total numbers of the group members, the server performs the following processes:

1) Selects two large primes, $p_i$ and $q_i$, and computes $N_i = p_i \cdot q_i$.

2) Assigns an integer $s_i \in [2, N_i]$ as the private-key for each $U_i$, and computes $f_i$ such that $f_i \cdot s_i \equiv 1 \bmod \phi(N_i)$, where $\phi(N_i) = (p_i - 1)(q_i - 1)$. Note that the server should ensure that $s_i$ is unique for each registered member.

3) Delivers $s_i$ to $U_i$ via a secure channel and publishes the values $(f_i, N_i)$.

• *Sub-key Distribution and Commitment*

In order to distribute his/her temporary sub-key $K_i$ to the other participants securely, each $U_i$ executes the following processes:

1) Computes $R_i = (K_i)^{s_i} \bmod N_i$.

2) Computes $I_{ij} = (R_i \| ID_i)^{f_j} \bmod N_j$ for each participant $U_j$ $(j \neq i)$, where $ID_i$ is the public identity of $U_i$. Then, computes $h_i = h(K_i \| T_i)$, where $T_i$ is the current timestamp.

3) Publishes $M_i = (T_i, h_i, I_{i1}, I_{i2}, \cdots, I_{i(i-1)}, I_{i(i+1)}, \cdots, I_{in})$.

• *Sub-key Recovery and Verification*

After receiving $M_i$ from $U_i$, each participant $U_j$ $(j \neq i)$ performs the following procedures:

1) Checks whether the timestamp $T_i$ is valid. If the result is valid, $U_j$ continues the next process. Otherwise, $U_j$ claims that $U_i$ is fraudulent.

2) Recovers $R'_i$ and $ID_i$ by computing $(R'_i \| ID_i) = (I_{ij})^{s_j} \bmod N_j$, and then uses the corresponding $f_i$ to obtain $K'_i$ by computing $K'_i = (R'_i)^{f_i} \bmod N_i$.

3) Computes $h'_i = h(K'_i \| T_i)$ and then checks $h'_i \stackrel{?}{=} h_i$. If the result is equivalent, broadcasts $v_{ji} = success$ or else, broadcasts $v_{ji} = failure$.

After the above processes are completed, there are three possible cases:

Case 1: $v_{ji} = success$, this means that each member is legitimate, and, later, all the group members directly execute the session key computation phase.

Case 2: $v_{ji} = failure$ and $U_i$ is the malicious member.

Case 3: $v_{ji} = failure$ and $U_j$ is the malicious member. In this case, $U_j$ might cheat others who are honest members and exclude the honest member $U_i$.

• *Fault Detection*

This phase is executed when $v_{ji} = failure$. It can detect the real adversary by using the following processes:

1) After receiving $v_{ji} = failure$, each member waits for the fault detection message $\{R_i, K_i\}$ from $U_i$. Actually, there are two probable situations:

   (a) If no one receives the message from $U_i$ in a valid period, mark $U_i$ as the malicious member.

   (b) When receiving parameters $R_i$ and $K_i$ from $U_i$, each participant $U_m$ $(m \neq i)$ executes the following processes to detect fault:

      (i) Check $(R_i \| ID_i)^{f_m} \bmod N_m \stackrel{?}{=} I_{im} \bmod N_m$. If the result is wrong, $U_i$ must be the malicious member.

      (ii) Compute $h''_i = h(K_i \| T_i)$ and then check $h''_i \stackrel{?}{=} h_i$.
      If both results provided by (i) and (ii) are correct, $U_j$ is identified as a malicious member. Otherwise, it means that $U_i$ is the malicious member.

2) Remove all the malicious members from the group and restart the protocol.

• *Session Key Computation*

When malicious members are excluded from the group, each honest member of the set $U' = \{U'_1, U'_2, \cdots, U'_n\}$ can compute the group key as $k = K'_1 + K'_2 + \cdots + K'_n$. After establishing the group key $k$, each member will destroy the temporary sub-key.

Although Zhao et al. claimed that their scheme can detect and exclude the malicious participant in the group communication, we demonstrate below a possible case in which Zhao et al.'s scheme cannot detect the malicious participant. Assuming that an adversary wants to masquerade as an honest member $U_i$ to join the group communication, he or she might perform the following processes:

1) Randomly select $R_i \in \mathbb{Z}^*_{N_i}$ and then compute the temporary sub-key as $K_i = (R_i)^{f_i} \bmod N_i$.

2) Compute $I_{ij} = (R_i \| ID_i)^{f_i} \bmod N_j$ for each participant $U_j$ for $j = 1$ to $(n-1)$, $j \neq i$.

3) Compute $h_i = h(K_i \| T_i)$, where $T_i$ is the current timestamp, and then publish

$$M_i = (T_i, h_i, I_{i1}, I_{i2}, \cdots, I_{i(i-1)}, I_{i(i+1)}, \cdots, I_{in}).$$

In the sub-key recovery and verification phase, we can prove that the equivalence of $h'_i \stackrel{?}{=} h_i$ is true according to the following derivation:

We have $K'_i = (R'_i)^{f_i} \bmod N_i = (R_i)^{f_i} \bmod N_i = K_i$, hence $h_i = h(K_i \| T_i) = h(K'_i \| T_i) = h'_i$.

Thus, each $U_j$ will broadcast $v_{ji} = success$, meaning that the adversary can pass through the detection mechanism successfully, and the malicious participant is not excluded from the group communication.

From the above analysis, we show that Zhao et al.'s scheme cannot achieve the property of fault-tolerance, i.e., completely detect and exclude the adversary. On the other hand, each participant $U_i$ must publish $M_i = (T_i, h_i, I_{i1}, I_{i2}, \cdots, I_{i(i-1)}, I_{i(i+1)}, \cdots, I_{in})$ in the sub-key distribution and commitment phase. As the previous analysis of Huang et al.'s scheme, unicasting must be used for publishing the term $I_{ij}$ $(j \neq i)$, therefore, the communication cost would be increased. To overcome these drawbacks, we proposed a secure group key transfer protocol based on secret sharing. We illustrate the detailed processes of our scheme in the next session.

### 2.3 Harn and Lin's Scheme

In 2010, Harn and Lin proposed an efficient, authenticated group key transfer protocol based on Shamir's $(t, n)$-SS. As we know, Shamir's $(t,n)$-SS scheme satisfies two basic security requirements, as follows: 1) with knowledge of any $t$ or more than $t$ shares, it can reconstruct the secret $s$ easily and 2) with knowledge of fewer than $t$ shares, it cannot recover the secret $s$. In other words, the security of Shamir's $(t,n)$-SS scheme is *information-theoretically secure* since the scheme satisfies the above two requirements without making any computational assumptions. In Shamir's $(t,n)$-SS scheme, the secret of each shareholder is simply the $y$-coordinate of $f(x)$. However, Harn and Lin's scheme requires both the $x$-coordinate and the $y$-coordinate for each shareholder's secret. Moreover, in Shamir's $(t,n)$-SS scheme, the modulus $p$ used for all computations is a large prime number. To prevent the insider attack, Harn and Lin used modulus $N$, a composite integer, to replace modulus $p$. In addition, Harn and Lin's scheme consists of three processes, i.e., 1) initialization of KGC, 2) user registration, and 3) group key generation and distribution. Each of these processes is described below.

• *Initialization of KGC*

The KGC randomly generates two large, safe prime numbers $p$ and $q$ (i.e., prime numbers such that $p' = \frac{p-1}{2}$ and $q' = \frac{q-1}{2}$ are also prime numbers) and computes $N = p \cdot q$, where $N$ is publicly known.

• *User Registration*

Each user is required to register with the KGC for subscribing to the group key distribution service. The KGC shares a secret, $(x_i, y_i)$, with each user $U_i$, where $x_i, y_i \in \mathbb{Z}_N^*$.

• *Group Key Generation and Distribution*

Upon receiving a group key generation request from any user, the KGC selects a group key randomly and distributes this group key to all group members in a secure manner. Assume that a group consists of $t$ members, $\{U_1, U_2, \cdots, U_t\}$ and the shared secrets are $(x_i, y_i)$ for $i = 1, 2, \cdots, t$. The key generation and distribution process contains the following five steps:

Step 1. The *initiator* sends a key generation request to the KGC with a list of group members, i.e., $\{U_1, U_2, \cdots, U_t\}$.

Step 2. The KGC broadcasts the list of all participating member, $\{U_1, U_2, \cdots, U_t\}$, as the response.

Step 3. Each participating member selects a random number $R_i \in \mathbb{Z}_N^*$ and then sends $R_i$ to the KGC.

Step 4. The KGC randomly generates a group key $k$ and then generates an interpolated polynomial $f(x)$ with degree $t$ to pass through $(t+1)$ points, $(0, k)$ and $x_i, y_i \oplus R_i$, for $i = 1, 2, \cdots, t$. The KGC also computes $t$ additional points, $P_t$, for $i = 1, 2, \cdots, t$, on $f(x)$ and $Auth = h(k\|U_1\|U_2\| \cdots \|U_t\|R_1\|R_2\| \cdots \|R_t\|P_1\|P_2\| \cdots \|P_t)$, where $h(\cdot)$ is a collision-free, one-way hash function and $\|$ is the concatenation operator that combines the two values into one. The KGC broadcasts $\{Auth, P_i\}$ for $i = 1, 2, \cdots, t$ to all group members. All computations are performed in $\mathbb{Z}_N^*$.

Step 5. Each group member, $U_i$, who knows the shared secret, $(x_i, y_i \oplus R_i)$, and $t$ additional public points, $P_i$, for $i = 1, 2, \cdots, t$, on $f(x)$, can reconstruct the polynomial $f(x)$ and recover the group key $k = f(0)$. Afterwards, each group member $U_i$ computes $h(k\|U_1\|U_2\| \cdots \|U_t\|R_1\|R_2\| \cdots \|R_t\|P_1\|P_2\| \cdots \|P_t)$, and then determines whether this hash value is identical to *Auth*. If the result is correct, the authenticated key $k$ is established among all group members.

Although Harn and Lin's scheme can provide high security and distribute the group key efficiently, it requires an on-line KGC to construct and transfer the group key, which increases the overhead required to implement the system and reduces its flexibility. In addition, Harn and Lin did not propose a practical method to share the secrets $(x_i, y_i)$ between the KGC and users $U_i$ for real-life applications.

## 3. Improvement of Group Key Establishment Protocol

We propose a secure group key transfer protocol which overcomes the drawbacks in the previous schemes. In addition, the proposed scheme is more efficient in term of communication overhead. We first describe the concept of our design in Sect. 3.1. Next, a practical design example is presented in Sect. 3.2.

### 3.1 The Concept of Our Design

When the confidentiality of group communications must be assured, a one-time session key (group key) should be shared among communication members. The well-known Shamir's $(t,n)$-SS scheme can be employed to establish the common session key for all the group members. However, the conventional group key transfer schemes based on secret sharing (SS) require an on-line, trusted KGC as the dealer to issue the *shares* (shadows) for each member. In addition, the KGC must generate a secret key as the group key and then use the SS scheme to transmit the group key to all members. Actually, this approach can result in loss of flexibility and cause an increase in the overhead associated with the implementation of the system. To overcome these drawbacks, an *initiator*, one of the group members, is endowed with the authority to select a secret key as the group key and to originate the group communication. In addition, the *initiator* must share secrets with the other members by using an efficient method.

It is well know that the interactive key agreement protocol can construct a one-time secret between two parities in public environments. In our design, the concept of DH key agreement protocol is used to share secrets between the *initiator* and the others members of the group. Further, the *initiator* can construct an interpolated polynomial $f(x)$ passing through these shares and the selected session key by using Lagrangian interpolation, where the degree of $f(x)$ is equal to the number of group members minus one, and the session key is the term $f(0)$. Afterwards, the initiator publishes some additional points on $f(x)$, where the number of those public points is equal to the number of group members minus one. On the other hand, each group member except the initiator is able to use his/her secret with those public points to reconstruct the polynomial $f(x)$ and derive the session key as $f(0)$ by using the Lagrangian interpolation. Finally, all group members share a common session key for group communications. A practical design example is illustrated in the next section.

### 3.2 The Design Example

The proposed protocol consists of two phases, i.e., 1) the secret establishment phase and 2) the session key transfer phase. Suppose that a set of $t$ participants, $U = \{U_1, U_2, \cdots, U_t\}$, wants to set up a secure communication. Each participant must maintain a public/private key pair $(puk, prk)$, such that $puk = g^{prk} \bmod p$, where $g \in \mathbb{Z}_p^*$, $p$ is a large, safe prime number. Note that the long-term pair $(puk, prk)$ is authenticated by a trusted authority with the corresponding certificate. An *initiator*, one of the group members, is endowed with the authority to select a secret key as the group key and to originate the group communication. The secret establishment phase contains the following processes.

1) The *initiator* broadcasts a request containing a random number $r_i \in \mathbb{Z}_p^*$, his/her long-term public-key $puk_i$, and a list of members, $U = U_1, U_2, \cdots, U_t$, to announce the group communication.

2) Upon receiving the announcement from the *initiator*, each participating group member $U_j$ ($j \neq$ *initiator*), for $j = 1, 2, \cdots, (t-1)$, selects a random number $r_j \in \mathbb{Z}_p^*$ and uses his/her private-key $prk_j$ to compute the secret as $s_j = puk_i^{prk_j \cdot r_i \cdot r_j} \bmod p$. Afterwards, $U_j$ computes $Auth_j = h(s_j \| r_i)$, and sends $\{r_j, puk_j, Auth_j\}$ to the *initiator* as a response.

3) After receiving the message from each $U_j$, the initiator computes $s_j^* = puk_j^{prk_i \cdot r_i \cdot r_j} \bmod p$ and then checks $Auth_j \stackrel{?}{=} h(s_j^* \| r_i)$. If the result is valid, the *initiator* believes that the secret $s_j = g^{prk_i \cdot prk_j \cdot r_i \cdot r_j} \bmod p$ is shared with corresponding $U_j$. Otherwise, the *initiator* claims that $U_j$ is fraudulent and then restarts the protocol.

In the session key transfer phase, the *initiator* and the other participating members $U_j$ execute the following processes:

1) The *initiator* separates each shared secret $s_j$ into two parts to derive the point $(x_j, y_j)$, where $(x_j \| y_j) = s_j$, and randomly generates a session key $k$. Then, the *initiator* constructs an interpolated polynomial $f(x)$ of degree $(t-1)$ to pass through $t$ points, $(0, k)$ and $(x_j, y_j)$, for $j = 1$ to $(t-1)$, by using Lagrangian interpolation. Afterwards, the *initiator* also computes $(t-1)$ additional points $P_i$ on $f(x)$, where $P_i = (x_i, y_i)$, for $i = 1$ to $(t-1)$. Finally, the *initiator* computes $Auth = h(k \| r_i \| U_1 \| U_2 \| \cdots \| U_t \| P_1 \| P_2 \| \cdots \| P_{t-1})$ and broadcasts the message $\{Auth, P_i\}$, for $i = 1$ to $(t-1)$, to $U_j$.

2) For each participating member $U_j$, knowing $s_j$ and $(t-1)$ additional points $P_i$, for $i = 1$ to $(t-1)$, is able to reconstruct the polynomial $f(x)$ and derive the group key $k = f(0)$ by using Lagrangian interpolation. Afterward, $U_j$ computes $Auth^* = h(k \| r_i \| U_1 \| U_2 \| \cdots \| U_t \| P_1 \| P_2 \| \cdots \| P_{t-1})$ and then checks the hash value $Auth^* \stackrel{?}{=} Auth$. If the result is correct, the group key $k$ is authenticated.

After the above processes have been executed successfully, the session key $k$ is established among all group members. Later, the key $k$ can be used for secure group communications.

## 4. Discussion

In this section, we focus on two kinds of possible attacks, i.e., the *insider attack* and the *outsider attack*, for analyzing the security of our protocol. In addition, we also discuss some security requirements, such as *group key security* and *forward secrecy*. Finally, we compare the required functionalities of our scheme with three other related works.

### 4.1 Withstand Possible Attacks

We discuss some possible attacks and perform the heuristic security analyses for these attacks. First, the basic assumption is given as follows:

**Assumption 1 (The Computational Diffie-Hellman (CDH) Assumption).** Let $G = \langle g \rangle$ be a multiplicative cyclic group of order $q$, and two random integers $a$, $b$ are chosen in $\mathbb{Z}_q^*$. Given $g$, $g^a$, and $g^b$, the adversary has a negligible success probability $\varepsilon$ for obtaining an element $z \in G$, such that $z = g^{ab}$ within polynomial time.

Based on the CDH assumption, we consider two scenarios of attacks, i.e., the adversaries are outsiders and the adversaries are insiders of the group. In the first type, the outside adversary might try to masquerade as a group member and to obtain the secret group key. We will show that the outside attacker cannot recover the group key since the attacker cannot obtain the one-time shared secret. In the second type, the attackers are insiders of a group who are authorized to know the secret group key. The attacker wants to retrieve the previous secrets between the other members and the *initiator*. Thus, we need to ensure the one-time secret used in each session cannot be determined by the inside attacker.

**Proposition 1** (Withstand Outsider Attacks):
Assume that an adversary wants to masquerade as a group member to join the group communication; then, the adversary can neither obtain the group key nor share a group key with any group member.

**Proof** Although the adversary can intercept the messages between the *initiator* and the participating members $U_j$, the adversary cannot share the one-time secret $s_j$, i.e., $s_j = puk_i^{prk_j \cdot r_i \cdot r_j} \bmod p$, with the *initiator* successfully, due to the fact that the long-term private-key $prk_j$ of any member $U_j$ is unknown. In addition, the group key $k$, which is constructed by using secret sharing schemes, can only be recovered by any honest member who has the correct corresponding shared secret $s_j$. Therefore, the adversary cannot masquerade as any group member to obtain the group key $k$ by intercepting messages. On the other hand, since the adversary does not have the private-key $prk_i$ of the *initiator*, thus, the adversary cannot masquerade as the *initiator* successfully to share the secret $s_j$ with the other members. In other words, the adversary cannot share the key $k$ with any group member by masquerading as the *initiator*. □

**Proposition 2** (Withstand Insider Attacks):
Assume that the protocol has run successfully many times; then, the one-time secret $(x_j, y_j)$, where $(x_j \| y_j) = s_j$, of each $U_j$ shared with the *initiator* still cannot be traced by other group members.

**Proof** In order to transfer the group key $k$, the *initiator* generates a polynomial $f(x)$ of degree $(t-1)$ to pass through $t$ points, $(0, k)$ and $(x_j, y_j)$, for $j = 1$ to $(t-1)$. Each honest group member $U_j$ can obtain the one-time secret $(x_j, y_j)$ shared with the *initiator* by using a interactive key agreement protocol. Later, with knowledge of the one-time secret $(x_j, y_j)$ and $(t-1)$ public information, i.e., knowing $t$ points of $f(x)$, any honest group member can reconstruct the polynomial $f(x)$. However, the secret $(x_j, y_j)$ of each group member shared with the *initiator* is still untraceable by insiders, due to the fact that the one-time secret $(x_j, y_j)$ depended on random nonces $(R_i, R_j)$ and long-term private-keys $(prk_i, prk_j)$. □

### 4.2 Security of Group Key

In our protocol, we focus on protecting group key information transferred from the *initiator*. Since the group key $k$ is the constant term $f(0)$ of the polynomial $f(x)$ by employing Shamir $(t, n)$-SS, a participant who has $t$ shares or more than $t$ shares can reconstruct the polynomial and recover the secret key $k = f(0)$. In other words, Shamir $(t, n)$-SS scheme is information-theoretically secure, so the group key transfer procedure (i.e., the second phase) of the proposed scheme is also information-theoretically secure.

Moreover, the one-time secret $s_j$ is generated by a interactive key agreement protocol with random nonces, and then the shared secret $s_j$ is used to construct the interpolated polynomial $f(x)$. Even though the current group key is compromised, it does not reveal any information regarding the previous group keys. Therefore, our protocol achieves forward secrecy.

**Remark:** Most key transfer schemes based on Shamir's $(t, n)$-SS are claimed information-theoretically secure. However, these schemes must pre-share secrets (shadows) between the dealer and each participant. In other words, the secrets must be shared via a secure channel. Actually, it is a strong assumption to suppose that a secure channel is existed in public networks. That is, most existing schemes do not propose any practical method to share secrets in public networks. In this article, we first proposed a method based on the CDH assumption to share the secrets between the *initiator* and another participants. Next, we proposed a group key transfer protocol based on Shamir's $(t, n)$-SS. Since the concept of Shamir's $(t, n)$-SS is adopted to transfer the group key, so we say that the group key transfer procedure of our scheme is also information-theoretically secure.

**Table 2** Comparison of our protocol with three recent protocols.

| Protocols | Huang et al.'s (2009) [6] | Zhao et al.'s (2010) [7] | Harn and Lin's (2010) [13] | Ours |
|---|---|---|---|---|
| Without registration with an trusted server | No | No | No | Yes |
| Without an on-line KGC | Yes | Yes | No | Yes |
| Group key generated by users | Yes | Yes | No | Yes |
| Excludes malicious participants from the group | Yes | No | Yes | Yes |
| No additional time-synchronized mechanism required [20], [21] | No | No | Yes | Yes |
| Low communication cost | No | No | Yes | Yes |

### 4.3 Functionality Comparison

We compared the major security requirements and the cost of communications of our protocol with three recent protocols, and the results are summarized in Table 2. The results show that our protocol is the only one that is capable of achieving all desired functionalities.

## 5. Conclusions

In this article, we discussed some potential drawbacks of existing group key establishment protocols and proposed a practical key transfer protocol based on secret sharing for group communications. The group members can construct and share the common session key efficiently without an on-line KGC. Besides, malicious participants can be excluded completely from the group. Moreover, our scheme uses random nonces to withstand replay attacks; therefore, it does not require additional time-synchronized mechanisms [20], [21] in implementation.

## References


[1] W. Diffie and M.E. Hellman, "New directions in cryptography," IEEE Trans. Inf. Theory, vol.IT-22, no.6, pp.644–654, Nov. 1976.
[2] C.S. Laih and W.C. Kuo, "New signature schemes based on factoring and discrete logarithms," IEICE Trans. Fundamentals, vol.E80-A, no.1, pp.46–53, Jan. 1997.
[3] K. Oishi, M. Mambo, and E. Okamoto, "Anonymous public key certificates and their applications," IEICE Trans. Fundamentals, vol.E81-A, no.1, pp.56–64, Jan. 1998.
[4] B. Klein, M. Otten, and T. Beth, "Conference key distribution protocols in distributed systems," Proc. Codes and Ciphers: Cryptography and Coding IV, pp.225–242, 1995.
[5] W.G. Tzeng, "A secure fault-tolerant conference key agreement protocol," IEEE Trans. Comput., vol.51, no.4, pp.373–379, April 2002.
[6] K.H. Huang, Y.F. Chung, H.H. Lee, F. Lai, and T.S. Chen, "A conference key agreement protocol with fault-tolerant capability," Computer Standards and Interfaces, vol.31, no.2, pp.401–405, Jan. 2009.
[7] J. Zhao, D. Gu, and Y. Li, "An efficient fault-tolerant group key agreement protocol," Comput. Commun., vol.33, no.7, pp.890–895, May 2010.
[8] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," Commun. ACM, vol.21, no.2, pp.120–126, Feb. 1978.
[9] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly secure key distribution for dynamic conferences," Inf. Comput., vol.146, no.1, pp.1–23, Oct. 1998.
[10] G. Saze, "Generation of key predistribution schemes using secret sharing schemes," Discrete Appl. Math., vol.128, no.1, pp.239–249, May 2003.
[11] C.S. Laih, J.Y. Lee, and L. Harn, "A new threshold scheme and its application in designing the conference key distribution cryptosystem," Inf. Process. Lett., vol.32, no.3, pp.95–99, Aug. 1989.
[12] S. Berkovits, "How to broadcast a secret," Proc. Advances in Cryptology-EUROCRYPT '91: Workshop on the Theory and Application of Cryptographic Techniques, pp.535–541, Brighton, UK, April 1991.
[13] L. Harn and C. Lin, "Authenticated group key transfer protocol based on secret sharing," IEEE Trans. Comput., vol.59, no.6, pp.842–846, June 2010.
[14] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," Proc. 3rd ACM Conference on Computer and Communications Security (CCS '96), pp.31–37, March 1996.
[15] E. Bresson, O. Chevassut, D. Pointcheval, and J.J. Quisquater, "Provably authenticated group Diffie-Hellman key exchange," Proc. 8th ACM Conference on Computer and Communications Security (CCS '01), pp.255–264, Philadelphia, USA, Nov. 2001.
[16] J.M. Bohli, "A framework for robust group key agreement," Proc. International Conference on Computational Science and Applications (ICCSA '06), pp.355–364, Glasgow, UK, May 2006.
[17] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," J. Cryptology, vol.20, no.1, pp.85–113, Jan. 2007.
[18] Y.M. Tseng, "A communication-efficient and fault-tolerant conference-key agreement protocol with forward secrecy," J. Syst. Softw., vol.80, no.7, pp.1091–1101, July 2007.
[19] A. Shamir, "How to share a secret," Commun. ACM, vol.22, no.11, pp.612–613, Nov. 1979.
[20] D.L. Mills, "Precision synchronization of computer network clocks," ACM SIGCOMM Computer Communication Review, vol.24, no.2, pp.28–43, April 1994.
[21] D.L. Mills, "Adaptive hybrid clock discipline algorithm for the network time protocol," IEEE/ACM Trans. Networking, vol.6, no.5, pp.505–514, Oct. 1998.


**Chia-Yin Lee** received the B.S. and the M.S. degrees in computer science from Tunghai University, Taichung, Taiwan, in 2001 and 2004, respectively, and the Ph.D. degree in computer science and information engineering from National Chung Cheng University, Chiayi, Taiwan, in 2010. His current research interests include information security, wireless networking, and image processing.

**Zhi-Hui Wang** received her B.S. degree in software engineering from the North Eastern University, Shenyang, China, in 2004. She received her M.S. degree in software engineering and her Ph.D. degree in computer software and theory from Dalian University of Technology, Dalian, China, in 2007 and 2010, respectively. Her research interests include information hiding and image processing.

**Lein Harn** received the Ph.D. degree in electrical engineering from the University of Minnesota, USA, in 1984. Since 1995, he has been a full professor at Department of Computer and Electrical Engineering, University of Missouri-Kansas City, USA. His research interests include cryptography, network security, and digital signal processing.

**Chin-Chen Chang** received the Ph.D. degree in computer engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1982. Since 2005, he has been a chair professor at Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan. He has also served as a consultant to several research institutes and government departments. His specialties include database design, computer cryptography, image compression, and data structures. Dr. Chang is currently a Fellow of IEEE.