Round-Efficient Conference Key Agreement Protocols with Provable Security*

Wen-Guey Tzeng and Zhi-Jia Tzeng

Department of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan 30050
{tzeng,zjtzeng}@cis.nctu.edu.tw

Abstract. A conference key protocol allows a group of participants to establish a secret communication (conference) key so that all their communications thereafter are protected by the key. In this paper we consider the distributed conference key (conference key agreement) protocol. We present two round-efficient conference key agreement protocols, which achieve the optimum in terms of the number of rounds. Our protocols are secure against both passive and active adversaries under the random oracle model. They release no useful information to passive adversaries and achieve fault tolerance against *any coalition* of malicious participants. We achieve the optimal round by transferring an interactive proof system to a non-interactive version, while preserving its security capability.

1 Introduction

A conference key protocol allows a group of participants to establish a secret communication (conference) key so that all their communications thereafter are protected by the key. In this paper we consider the distributed conference key (conference key agreement) protocol under the broadcast channel model in which sent messages are guaranteed to be received intact. Nevertheless, the attacker can inject false messages.

For security, we consider both active and passive adversaries. A passive adversary, eavesdropper, tries to learn information by listening to the communication of the participants. There are two types of active adversaries: impersonators and malicious participants. An impersonator tries to impersonate as a legal participant. A malicious participant tries to disrupt conference key establishment among honest participants.

Our protocols focus on round efficiency. We would like to have a conference key agreement protocol by which the participants exchange messages with as few rounds as possible even when active adversaries are present. In this paper we present two round-efficient conference key agreement protocols that achieve

* Research supported in part by the National Science Council grant NSC-89-2213-E-009-180 and by the Ministry of Education grant 89-E-FA04-1-4, Taiwan, ROC.

T. Okamoto (Ed.): ASIACRYPT 2000, LNCS 1976, pp. 614–627, 2000.

the optimum in terms of the number of rounds, that is, they use only one round even in the worst scenario. After each participant sends messages to and receives messages from other participants, they go on to compute the conference key no matter whether the attack of active adversaries occurs. Our protocols are secure against both passive and active adversaries under the random oracle model. They release no useful information to passive adversaries and achieve fault tolerance against *any coalition* of malicious participants. We achieve the optimal round by transferring an interactive proof system to a non-interactive version, while preserving its security capability.

1.1 Related Work

Computing a conference key among a set of participants is a special case of secure multiparty computation in which a group of people, who each possesses a private input k_i, computes a function $f(k_1, k_2, \cdots)$ securely [2]. Therefore, it is possible to have a secure conference key agreement protocol by the generic construction for secure multiparty computation. Nevertheless, it is an overkill. Furthermore, there are some distinct features for the conference key agreement protocol. First, a cheater's goal in conference key agreement is to disrupt conference key establishment among the set of honest participants, which is quite different from that in secure multiparty computation. Second, since a cheater's secret is not a necessity in conference key agreement, the cheater can be simply excluded when detected. On the other hand, in secure multiparty computation when a cheater is found, the cheater's secret x_i, which is shared into others, is recovered by honest participants so that evaluation can proceed.

There have been intensive research on conference key protocols. Conference key distribution protocols (with a chairman) have been studied in [3,9,10,19]. Pre-distributed conference key protocols have been studied in [4,5,22]. And conference key agreement protocols have been studied in [17,19,20,27,29,28]. Information-theoretically secure conference key protocols have been studied in [5,12]. Most proposed protocols except [18,28] do not have the capability of fault-tolerance so that a malicious participant can easily mislead other participants to compute different conference keys so that the honest participants cannot confer correctly.

Burmester and Desmedt [7] proposed a round-efficient (two-round) protocol (Protocol 3) with $f(k_1, k_2, \ldots, k_n) = g^{k_1k_2+k_2k_3+\cdots+k_nk_1} \bmod p$. In the modified Protocol 7 (authenticated key distribution), they used an interactive proof for authenticating sent messages to show that the protocol is secure against impersonators. However, both protocols cannot withstand the attack of malicious participants. The fault-tolerant conference key agreement protocol of Klein et al. [18] is quite inefficient and its security is not rigidly proved. In [28], when malicious participants are detected, the protocol restarts for the remained participants. It can may be that a participant behaves maliciously in a new round and thus the protocol has to restart again. So, the protocols have to run $O(m)$ times for m malicious participants in the worst case. This may be inefficient since the number of rounds may entail main communication cost,

2 Preliminaries

A user in a conference key system is a probabilistic polynomial-time Turing machine. Each user U_i has a secret key x_i and a corresponding public key y_i. The system has a public directory of recording the system's public parameters and each user's public key that can be accessed by every one. All users are connected by a broadcast network such that the messages sent on the network cannot be altered, blocked or delayed. For simplicity, we assume that the network is synchronous, that is, for a given phase of a round, all users send their messages to other recipients (or receive messages from others senders) simultaneously. No private channel exists between users. A group of users who wants to establish a conference key is called the *set of participants*.

We consider three types of adversaries. They are all probabilistic polynomial-time Turing machines. An *eavesdropper*, who is not a participant, listens to the broadcast channel and tries to learn the conference key established by the honest participants. An *impersonator*, who is an outsider, tries to impersonate as a legal participant. A *malicious participant*, who is a participant, tries to disrupt establishment of a common conference key among the honest participants. A malicious participant mainly sends "malicious" messages to fool an honest participant to believe that he has computed the same conference key as that of other honest participants, while he does not indeed. We do not care about the possibility that two or more cheating participants collaborate and result in one of them or other malicious participants not being able to compute the key. For example, a malicious participant U_i sends "malicious" messages, but all honest participants compute the same key. Another malicious participant U_j, though receiving an incorrect key, still claims that he has had received the correct key. We tolerate this case since this type of collaboration between malicious U_i and U_j do no harm to the honest participants. We do not restrict the number of malicious participants in a conference.

A conference key agreement protocol should meet the following requirements:

- Authentication: an outsider cannot impersonate as a legal participant.
- Correctness: the set of honest participants who follow the protocol computes a common conference key.
- Fairness: the conference key should be determined unbiasedly by all honest participants together.
- Fault tolerance: no coalition of malicious participants can spoil the conference by making honest participants compute different conference keys.
- Privacy: an eavesdropper can not get any information about the conference key established by the honest participants.

We consider two types of communication cost:

- Message efficiency: the total number of messages sent by the participants for completing the protocol. This includes the extra messages for dealing with malicious participants.

– Round efficiency: the total number of rounds executed by the participant for completing the protocol. This includes the extra rounds for dealing with the malicious participants.

In security analysis we use the random oracle model [1], which assumes that a cryptographically strong (collision-free) hash function is a random function. Although this is only a security argument [8], it is a suitable paradigm for analyzing our first protocol.

3 Basic Techniques

We use the following setting for the system and users throughout the rest of the paper. The system has public parameters:

– p: a large prime number that is $2q + 1$, where q is a large prime also.
– g: a generator for the subgroup G_q of all quadratic residues in Z_p^*.

Each user U_i has two parameters:

– Private parameter x_i: a number in $Z_q^* - \{1\}$.
– Public parameter $y_i = g^{x_i} \bmod p$. Since q is a prime number, y_i is a generator for G_q.

Let $x \in_R S$ denote that x is chosen from the set S uniformly and independently and $[a..b]$ denote the set of numbers in between a and b, where $a \leq b$. In order to simplify presentation, we omit the or complexity measure n from the related parameters, unless necessary. For example, when we say a probability ϵ is negligible, we mean that for any positive constant c, $\epsilon = \epsilon(n) < 1/n^c$ for large enough n. A probability δ is overwhelming if $\delta = 1 - \epsilon$ for some negligible probability ϵ.

The discrete logarithm (DL) problem is to compute $x \equiv \log_g y \pmod p$ from given (y, g, p), where $p = 2q + 1$, g is a generator of G_q and $y \in_R G_q$. The decisional Diffie-Hellman (DDH) problem is to distinguish the distributions

$$(g_1, g_2, g_1^r \bmod p, g_2^r \bmod p) \text{ and } (g_1, g_2, u_1, u_2)$$

with a non-negligible probability, where g_1 and g_2 are generators of G_q, $r \in_R Z_q$ and $u_1, u_2 \in_R G_q$. We assume that the DL and DDH problems are computationally infeasible. They are called the DL assumption (DLA) and the DDH assumption (DDHA). In particular, any probabilistic polynomial-time algorithm cannot solve even a non-negligible fraction of input of the DL problem.

The main building block of our conference key agreement protocols is a protocol of sending a (random) secret to the other participants such that any one can verify that all participants receive the same secret. We call this as the protocol for a publicly verifiable secret (PVS). Let t be the security parameter of the system. If participant U_i wants to send the secret (subkey) $g^{k_i} \bmod p$ to all other participants in a publicly verifiable way, it broadcasts

$$u_{i,j} = y_j^{k_i} \bmod p, 1 \leq j \leq n,$$

where $k_i \in_R Z_q$. Another participant U_j can obtain the shared secret $g^{k_i} \bmod p$ with U_i by computing $(u_{i,j})^{x_j^{-1}} \bmod p$. The PVS proof system shows that

- $\log_{y_1} u_{i,1} \equiv \log_{y_2} u_{i,2} \equiv \cdots \equiv \log_{y_n} u_{i,n} \pmod p$, and
- U_i knows that the exponent $k_i = \log_{y_j} u_{i,j} \bmod p$, $1 \le j \le n$.

with error probability $1/2^t$. We can make the error probability inverse exponentially small by repeating the system for a polynomial number of times. The PVS proof system is:

1. $P \to V$: $b_j = y_j^r \bmod p$, $1 \le j \le n$, where $r \in_R Z_q$;
2. $V \to P$: $c \in_R [0..2^t - 1]$;
3. $P \to V$: $w = r - ck_i \bmod q$;
4. V checks whether $b_j = y_j^w \cdot u_{i,j}^c \bmod p$, $1 \le j \le n$.

Theorem 1. *Assume the DLA. The PVS proof system above is complete, sound and zero-knowledge.*

Proof. The completeness property can be verified easily. For soundness, if a probabilistic polynomial-time adversary A can impersonate P with a non-negligible probability ϵ, the verifier V and A together can solve the discrete logarithm problem with an overwhelming probability. Since the probability ϵ is non-negligible, one can use A to generate two responses $w_1 = r - c_1 k_i \bmod q$ and $w_2 = r - c_2 k_i \bmod q$ for the same commitment b_j's and two different challenges c_1 and c_2. One can compute U_i's secret key $k_i = (w_1 - w_2)(c_2 - c_1)^{-1} \bmod q$. Furthermore, if the prover does not know k_i, he can pass a challenge by the verifier V with the probability of $1/2^t$.

To simulate the view of a verifier V^*, the simulator S first selects $c \in_R [0..2^t - 1]$ and $w \in_R Z_q$ and computes $b_j = y_j^w \cdot u_{i,j}^c \bmod p, 1 \le j \le n$. S then simulates $V^*(b_1, b_2, \ldots, b_n)$ to get c'. If $c = c'$, then S outputs $(b_1, b_2, \ldots, b_n, c, w)$. Otherwise, S resets V^* to its original state before this round of simulation and starts the next round of simulation. The output of S and the view of V^* are statistically indistinguishable. □

We need the proof system to be non-interactive. By the standard technique [14], we replace V with a cryptographically strong (collision-resistant) hash function $\mathcal{H}$ for generating the challenge c. In the non-interactive paradigm, the interactive version of a proof system need only be zero-knowledge for the honest verifier. Our PVS proof system is honest-verifier zero-knowledge even when $c \in Z_q$. Therefore, we choose $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{\lceil \log q \rceil}$.

The message (c, w) sent by U_i for non-interactive PVS satisfies

$$c = \mathcal{H}(g||y_1|| \cdots ||y_n||u_{i,1}|| \cdots ||u_{i,n}||y_1^w u_{i,1}^c|| \cdots ||y_n^w u_{i,n}^c)$$

where $||$ is the concatenation operator of strings. U_i can compute (c, w) by choosing $r \in_R Z_q$, computing $c = \mathcal{H}(g||y_1|| \cdots ||y_n||u_{i,1}|| \cdots ||u_{i,n}||y_1^r|| \cdots ||y_n^r)$, and setting $w = r - ck_i$. We shall use NIPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$

to denote the non-interactive proof system described above. By verifying $\log_{y_1} u_{i,1} \equiv \log_{y_2} u_{i,2} \equiv \cdots \equiv \log_{y_n} u_{i,n}$, one can be assured that all participants receive the same secret value g^{k_i}. The proof system releases no useful information assuming the DLA and the random oracle model.

The integral APVS (PVS with authentication) proof system achieves public verification of a secret and authentication of an identity simultaneously. For APVS, the participant U_i broadcasts

$$u_{i,j} = y_j^{k_i} \bmod p, 1 \leq j \leq n,$$

to other participants, where $k_i \in_R Z_q$. Given the broadcast messages, the APVS proof system is to show that

- $\log_{y_1} u_{i,1} \equiv \log_{y_2} u_{i,2} \equiv \cdots \equiv \log_{y_n} u_{i,n} \pmod p$,
- U_i knows that the exponent $k_i = \log_{y_j} u_{i,j} \bmod p$, $1 \leq j \leq n$, and
- U_i knows that the secret $x_i = \log_g y_i \bmod p$.

The APVS proof system with input $(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$ is:

1. $P \rightarrow V$: $b_j = y_j^{r_1} g^{r_2} \bmod p, 1 \leq j \leq n$;
2. $V \rightarrow P$: $c \in [0..2^t - 1]$;
3. $P \rightarrow V$: $w_1 = r_1 - ck_i \bmod q, w_2 = r_2 - cx_i \bmod q$;
4. V checks $b_j = y_j^{w_1} g^{w_2} (y_i u_{i,j})^c \bmod p, 1 \leq j \leq n$.

Theorem 2. *Assume the DLA. The APVS proof system is complete, sound and zero-knowledge.*

Proof. The completeness and soundness properties are easily checked.

For zero-knowledge, the simulator S simulates P's interaction with any verifier V^*. S randomly selects $c \in_R [0..2^t - 1]$, and $w_1, w_2 \in_R Z_q$ and computes $b_j = y_j^{w_1} g^{w_2} (y_i u_{i,j})^c \bmod p, 1 \leq j \leq n$. S then simulates $V^*(b_1, b_2, \ldots, b_n)$ to get c'. If $c = c'$, then S outputs $(b_1, b_2, \ldots, b_n, c, w_1, w_2)$. Otherwise, S resets V^* to its original state before this round of simulation. We can see that the output of S and V^*'s view with P are statistically indistinguishable. □

Again, we can make the proof system non-interactive by using a cryptographically strong hash function $\mathcal{H}$ in place of V. Let

$$\text{NIAPVS}(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n}) = (w_1, w_2, c)$$

denote the non-interactive APVS proof system such that

$$c = \mathcal{H}(g||y_1|| \cdots ||y_n||u_{i,1}|| \cdots ||u_{i,n}||y_1^{w_1} g^{w_2} (y_i u_{i,1})^c|| \cdots ||y_n^{w_1} g^{w_2} (y_i u_{i,n})^c),$$

where $c, w_1, w_2 \in_R Z_q$.

We now present two proofs for the Diffie-Hellman and equality properties, respectively. The proof system DH for the Diffie-Hellman property is to show that an input has the form $(g, u, v, z) = (g, g^a, g^b, g^{ab})$ and the prover knows a and b. The proof system EQ for the equality property is to show that an input has the form $(g, u, y, v) = (g, g^a, y, y^a)$ and the prover knows a.

The DH proof system is as follows.

1. $P \rightarrow V$: $a_1 = g^{r_1}, a_2 = u^{r_1}, b_1 = g^{r_2}, b_2 = v^{r_2}$, where $r_1, r_2 \in_R Z_q$;
2. $V \rightarrow P$: $c \in_R [0..2^t - 1]$;
3. $P \rightarrow V$: $w_1 = r_1 + bc \bmod q, w_2 = r_2 + ac \bmod q$;
4. V checks $g^{w_1} = a_1 v^c, u^{w_1} = a_2 (z)^c, g^{w_2} = b_1 u^c, v^{w_2} = b_2 (z)^c$.

Theorem 3. *The DH proof system is complete, sound and zero-knowledge.*

Proof. The system's completeness follows easily. For soundness, if an adversary, who does not know a and b, can impersonate the prover with a non-negligible probability, it can answer two different challenges c and c' of the same commitment (a_1, a_2, b_1, b_2), corresponding to r_1 and r_2, from the verifier. Let the adversary give the answers (w_1, w_2) and (w_1', w_2'). We can compute $b = (w_1 - w_1')/(c - c')^{-1} \bmod q$ and $a = (w_2 - w_2')/(c - c')^{-1} \bmod q$.

For zero-knowledge, the simulator S simulates P's interaction with any verifier V^*. S randomly selects $c \in_R [0..2^t - 1]$, and $w_1, w_2 \in_R Z_q$ and computes $a_1 = g^{w_1}/v^c$, $a_2 = u^{w_2}/z^c$, $b_1 = g^{w_2}/u^c$, and $b_2 = v^{w_2}/z^c$. S then simulates $V^*(a_1, a_2, b_1, b_2)$ to get c'. If $c = c'$, S outputs $(a_1, a_2, b_1, b_2, c, w_1, w_2)$; otherwise, S resets V^* to its original state before this round of simulation. We can see that the output of S and V^*'s view with P are statistically indistinguishable. □

The EQ proof system is:

1. $P \rightarrow V$: $b_1 = g^r \bmod p, b_2 = y^r \bmod p$, where $r \in_R Z_q$;
2. $V \rightarrow P$: $c \in_R [0..2^t - 1]$;
3. $P \rightarrow V$: $w = r - ca \bmod q$;
4. V checks $b_1 = g^w u^c \bmod p$ and $b_2 = y^w v^c \bmod p$.

Theorem 4. *The EQ proof system is complete, sound and zero-knowledge [11].*

We use NIDH and NIEQ to denote the non-interactive versions of the DH and EQ proof systems, respectively.

4 Our Round-Efficient Protocols

We present two round-efficient conference key protocols and show their security. The first one uses the PVS protocol for verifying the sender's subkey and digital signature for sender's identity. The second protocol uses the integral APVS for both subkey verification and identity authentication.

Since our protocols are non-interactive, we need use a session token ST, which is new for each conference session, in the hash functions to prevent the replay attack. Thus, in our protocols each collision-free hash function $\mathcal{H}(\cdot)$ is computed as $\mathcal{H}(ST, \cdot)$.

4.1 Protocol CONF-1

The protocol starts with that an initiator calls for a conference for a set $\mathcal{U}$ of participants and sets the session token ST. Without loss of generality, let $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$ be the initial participant set. Each participant U_i, $1 \leq i \leq n$, knows $\mathcal{U}$. Let H be a collision-resistant hashing function, which is used in the modified ElGamal signature scheme. In the protocol, each participant U_i first selects a random number k_i and computes his subkey $g^{k_i} \bmod p$. This subkey is conveyed to the other participants by sending $u_{i,j} = y_j^{k_i} \bmod p, 1 \leq j \leq n$. U_i sends NIPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$ for convincing other participants that all other participants receive the same subkey. U_i also sends the signature (r_i, s_i) of his subkey for authentication. After receiving messages from other participants, U_i checks whether the participant $U_j, j \neq i$, sends the correct messages and authenticates U_j's identity. If not, U_i excludes U_j from the set of honest participants. Then, U_i computes the conference key according to the set of honest participants. Our protocol is as follows.

1. **Message sending:** each participant U_i does the following:
   (a) Randomly select $k_i, R_i \in Z_q$.
   (b) Compute and broadcast $u_{i,j} = y_j^{k_i} \bmod p$, $1 \leq j \leq n$, NIPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$, $r_i = g^{R_i} \bmod p$ and $s_i = R_i^{-1}(H(ST, r_i, g^{k_i}) - r_i x_i) \bmod q$.
2. **Conference key computing:** each participant U_i does the following:
   (a) Fault detection and exclusion: for each $j \neq i$,
      – Compute $z_j = (u_{j,i})^{x_i^{-1}} \bmod p$ and verify whether (r_j, s_j) is the signature of z_j.
      – Verify NIPVS$(g, y_1, y_2, \ldots, y_n, u_{j,1}, u_{j,2}, \ldots, u_{j,n})$.
      If both checkings are correct, add U_j to its honest participant set $\mathcal{U}_i$.
   (b) Compute the conference key: assume that U_i's honest participant set $\mathcal{U}_i$ is $\{U_{i_1}, U_{i_2}, \ldots, U_{i_m}\}$. U_i computes the conference key

$$\begin{aligned} K &= (u_{i_1,i} u_{i_2,i} \cdots u_{i_m,i})^{x_i^{-1}} \bmod p \\ &= g^{k_{i_1} + k_{i_2} + \cdots + k_{i_m}} \bmod p. \end{aligned}$$

Note that only legal participants can verify whether (r_i, s_i) is the signature of the subkey z_i. This property is crucial to the proof of releasing no useful information in the random oracle model.

4.2 Security Analysis of CONF-1

We now show security of protocol CONF-1 on authentication, correctness, fairness, fault tolerance against malicious participants, and releasing no useful information.

We first show that all honest participants who follow the protocol compute the same conference key. The conference key is determined by all honest participant unbiasedly.

Theorem 5 (Fault tolerance, correctness and fairness). *All honest participants who follow the protocol compute a common conference key with an overwhelming probability no matter how many participants are malicious. Furthermore, the common conference key is determined by the honest participants unbiasedly.*

Proof. For fault tolerance, we show two things. First, any malicious participant U_i who tries to cheat another participant U_j to accept a different subkey will be excluded by all honest participants. Second, any honest participant will not be excluded by any other honest participant.

Since we assume the broadcast channel, every participant receives the same messages. If a malicious participant U_i sends $(y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$ such that not all $\log_{y_j} u_{i,j}, 1 \leq j \leq n$, are equal, the probability that he can construct NIPVS$(y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$ is at most T/q, which is negligible, where T is U_i's runtime. Thus, all honest participants will exclude the malicious participant with an overwhelming probability. We can easily check that an honest participant who follows the protocol shall be accepted by other honest participants as "honest". Therefore, each honest participant computes the same honest participant set with an overwhelming probability.

For correctness, since each honest participant U_i computes the same participant set, U_i uses his private key x_i to compute the subkeys $z_j = g^{k_j} \bmod p$ of all honest participants. Thus, they compute the same conference key with an overwhelming probability.

For fairness, since the common conference key is $g^{k_1+k_2+\cdots+k_n} \bmod p$, it is unbiased if any of $k_i, 1 \leq i \leq n$, is selected over Z_q uniformly and independently. Therefore, no participants can bias the conference key as long as one of the honest participants behaves properly. □

In our protocol, we let each participant sign his broadcast subkey by the modified ElGamal signature scheme, which is existentially unforgeable against the chosen ciphertext attack under the random oracle model [23]. No outsider can impersonate as a legal participant with a non-negligible probability under the chosen-ciphertext attack in the random oracle model.

Theorem 6 (Authentication). *Assume the random oracle model. If an outsider A can impersonate as a legal participant U_i to V with a non-negligible probability, A and V together can extract U_i's secret x_i from A with an overwhelming probability.*

Proof. Since the modified ElGamal signature scheme is secure against existential forgery under the chosen ciphertext attack, successful impersonation in the interactive system with a non-negligible probability would lead to computing U_i's secret x_i with an overwhelming probability. The use of session token ST makes our non-interactive protocol secure against the replay attack. Note that without special care, the replay attack is inevitable in non-interactive systems. Therefore, our protocol is authenticated under the random oracle model. □

Since we replace the verifier's challenge with a cryptographically strong hash function, the protocol is not zero-knowledge. But, it releases no useful information that can be used in the protocol under the random oracle model.

Theorem 7 (No useful information leakage). *Assume the DLA and the random oracle model. Protocol* CONF-1 *releases no useful information that can be used in the protocol.*

Proof. Even though the PVS protocol is complete, sound and honest verifier zero-knowledge, we cannot claim that our protocol release no useful information directly since U_i sends a signature (r_i, s_i) in addition. The simulator S should handle this case. To simulate U_i's output, $1 \leq i \leq n$, S selects $k_i \in Z_q$ randomly and computes $u_{i,j} = y_j^{k_i} \bmod p$, $1 \leq j \leq n$, and NIPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$. S then computes a forged signature (r_i', s_i') of U_i for the hash value h, i.e., $r_i' = g^a y_i^b \bmod p$, $s_i' = -r_i' b^{-1} \bmod q$ and $h = -r_i a b^{-1} \bmod p$ for $a \in_R Z_q$ and $b \in_R Z_q^*$. Since H is assumed to be a random function under the random oracle model, we let $H(ST, r_i', g^{k_i}) = h$. Finally, S outputs $(u_{i,1}, u_{i,2}, \ldots, u_{i,n}$, NIPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$, r_i', $s_i')$, together with a partial description of the random oracle H, i.e., setting $H(ST, r_i', g^{k_i}) = h$.

We now compare the output distributions of U_i and S. For the output of U_i, since h is random, (r_i, s_i) is independent of $(u_{i,1}, u_{i,2}, \ldots, u_{i,n}$, NIPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n}))$ and uniformly distributed over $G_q \times Z_q$ that satisfies $g^h \equiv y^{r_i} r_i^{s_i} \pmod p$. For the output of S, the distribution of $(u_{i,1}, u_{i,2}, \ldots, u_{i,n},$ NIPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n}))$ is the same as that of U_i. The distribution of (r_i', s_i') is also uniformly distributed over $G_q \times Z_q$ that satisfies $g^h \equiv y^{r_i'} r_i'^{s_i'} \pmod p$ since a and b are randomly chosen to fit the equation. Thus, the output distribution of S is equal to that of U_i under the random oracle model. Therefore, our protocol releases no useful information under the random oracle model. □

4.3 Protocol CONF-2

In this protocol, identity authentication is achieved by NIAPVS. The protocol is as follows.

1. **Message sending:** each participant U_i does the following:
   (a) Randomly select $k_i \in Z_q$.
   (b) Compute and broadcast $u_{i,j} = y_j^{k_i} \bmod p$, $1 \leq j \leq n$, NIAPVS$(g, y_1, y_2, \ldots, y_n, u_{i,1}, u_{i,2}, \ldots, u_{i,n})$.
2. **Conference key computing:** each participant U_i does the following:
   (a) Fault detection and exclusion: for each $j \neq i$, Verify NIAPVS$(g, y_1, y_2, \ldots, y_n, u_{j,1}, u_{j,2}, \ldots, u_{j,n})$. If the verification holds, add U_j to its honest participant set $\mathcal{U}_i$.

(b) Compute the conference key: assume that U_i's honest participant set $\mathcal{U}_i$ is $\{U_{i_1}, U_{i_2}, \ldots, U_{i_m}\}$. U_i computes the conference key

$$\begin{aligned} K &= (u_{i_1,i} u_{i_2,i} \cdots u_{i_m,i})^{x_i^{-1}} \bmod p \\ &= g^{k_{i_1}+k_{i_2}+\cdots+k_{i_m}} \bmod p. \end{aligned}$$

4.4 Security Analysis of Conf-2

The security analysis of NIAPVS-based Conf-2 is similar to that of NIPVS-based Conf-1.

Theorem 8. *Assume the DLA and the random oracle model. Protocol* Conf-2 *is correct, fair, fault-tolerant, and authenticated, and releases no useful information that can be used in the protocol.*

Proof. The only difference between Conf-1 and Conf-2 is that Conf-1 uses digital signature to authenticate participants, while Conf-2 uses NIAPVS to authenticate participants. Since Theorem 2 shows that participant's identity is authenticated, Conf-2 meets the security requirements. □

5 A Message-Efficient Protocol

The main protocol in [7] is message-efficient. If all the participants are honest, they broadcast $O(n)$ messages totally. But, the protocol is not fault tolerant, that is, it cannot withstand the attack of malicious participants. We can apply the technique of publicly verifiable secrets to obtain a message-efficient, but not round-efficient, conference key agreement protocol that meets the security requirements. The protocols's message complexity is $O(n)$ in the best case and $O(n^2)$ in the worst case. It seems that there is no easy way to augment the protocol to be both message- and round-efficient. The modified protocol is as follows.

1. **Message sending:** each participant U_i does the following:
   (a) Randomly select $k_i \in Z_q$.
   (b) Compute and broadcast $z_i = g^{k_i x_i} \bmod p$, $t_i = g^{k_i} \bmod p$ and NIDH(g, y_i, t_i, z_i).
2. **Message sending and fault detection:** each participant U_i does the following:
   (a) For each $j, j \neq i$, check whether NIDH(g, y_j, t_j, z_j) is valid. If yes, add U_j to his honest participant set $\mathcal{U}_i$.
   (b) Let $\mathcal{U}_i$=$\{U_1, U_2, \ldots, U_m\}$ and Z_i=$z_{(i+1 \bmod m)}/z_{(i-1 \bmod m)} \bmod p$. Compute and broadcast $Y_i = {Z_i}^{k_i x_i} \bmod p$ and NIEQ(g, z_i, Z_i, Y_i).
3. **Conference key computing:** each participant U_i does the following:
   (a) Fault detection and exclusion: for each $j \neq i$, validate NIEQ(g, z_j, Z_j, Y_j). If the validation does not hold, remove U_j from its honest participant set $\mathcal{U}_i$ and *restart* the protocol with the new honest participant set.

(b) Compute the conference key: assume that U_i's honest participant set $\mathcal{U}_i$ is $\{U_1, U_2, \ldots, U_m\}$. U_i computes the conference key

$$\begin{aligned} K &= (z_{i-1})^{mk_i x_i} Y_i^{m-1} Y_{i+1}^{m-2} \cdots Y_m^{i-1} Y_1^{i-2} Y_2^{i-3} \cdots Y_{i-2} \bmod p \\ &= g^{x_1 x_2 k_1 k_2 + x_2 x_3 k_2 k_3 + \cdots + x_n x_1 k_n k_1} \bmod p. \end{aligned}$$

Theorem 9. *Assume the DDHA and the random oracle model. The protocol above is correct and secure with authentication, fault tolerance, and leaking no useful information.*

Proof. The correctness follows from [7], while the security follows from the previous two round-efficient conference key agreement protocols. □

6 Conclusion

We have presented two round-efficient conference key agreement protocols. The protocols meet the security requirements: authentication, correctness, fairness, fault tolerance (robustness) and privacy. Their message complexity is $O(n^2)$ for n participants. We also modified Burmester and Desmedt's protocol so that it can withstand the attack of malicious participants.

It would be interesting to find a round-efficient protocol that meets all security requirements and has $O(n)$ message complexity.

References

1. M. Bellare, P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols", Proceedings of the First ACM Conference on Computer and Communications Security, pp.62-73, 1993.
2. M. Ben-Or, S. Goldwasser, A. Wigderson, "Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation", Proceedings of the 20th ACM Symposium on the Theory of Computing, pp.1-10, 1988.
3. S. Berkovits, "How to Broadcast a Secret", Proceedings of Advances in Cryptology - Eurocrypt '91, Lecture Notes in Computer Science 547, Springer-Verlag, pp.535-541, 1991.
4. R. Blom, "An Optimal Class of Symmetric Key Generation Systems", Proceedings of Advances in Cryptology - Eurocrypt '84, Lecture Notes in Computer Science 196, Springer-Verlag, pp.335-338, 1984.
5. C. Blundo, A.D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, M. Yung, "Perfectly-Secure Key Distribution for Dynamic Conferences", Proceedings of Advances in Cryptology - Crypto '92, Lecture Notes in Computer Science 740, Springer-Verlag, pp.471-486, 1992.
6. D. Boneh, R. Venkatesan, "Hardness of Computing the Most Significant Bits of Secret Keys in Diffie-Hellman and Related Problems", Proceedings of Advances in Cryptology - Crypto '96, Lecture Notes in Computer Science 1109, Springer-Verlag, pp.129-142, 1996.

7. M. Burmester, Y. Desmedt, "A Secure and Efficient Conference Key Distribution System", Proceedings of Advances in Cryptology - Eurocrypt '94, Lecture Notes in Computer Science 950, Springer-Verlag, pp.275-286, 1994.
8. R. Canetti, O. Goldreich, S. Halevi, "The Random Oracle Methodology Revisited", Proceedings of the 30th STOC, pp.209-218, 1998.
9. C.C. Chang, C.H. Lin, "How to Converse Securely in a Conference", Proceedings of IEEE 30th Annual International Carnahan Conference, pp.42-45, 1996.
10. C.C. Chang, T.C. Wu, C.P. Chen, "The Design of a Conference Key Distribution System", Proceedings of Advances in Cryptology - Auscrypt '92, Lecture Notes in Computer Science 718, Springer-Verlag, pp.459-466, 1992.
11. D. Chaum, T.P. Pedersen, "Wallet DataBases with Observers", Proceedings of Advances in Cryptography - Crypto'92, pp.90-105, 1992.
12. Y. Desmedt, V. Viswandathan, "Unconditionally secure dynamic conference distribution", IEEE International Symposium on Information Theory 98, pp.383, 1998.
13. W. Diffie, P.C. van Oorschot, M.J. Weiner, "Authentication and Authenticated Key Exchanges", Design, Codes and Cryptography Vol. 2, pp.107-125, 1992.
14. U. Feige, A. Fiat, A. Shamir, "Zero-Knowledge Proof of Identity", Journal of Cryptology Vol. 1, pp.77-94, 1988.
15. O. Goldreich, H. Krawczyk, "On the Composition of Zero-Knowledge Proof Systems", ICALP 90, Lecture Notes in Computer Science 443, pp.268-282, Springer-Verlag, 1990.
16. T. Hwang, J.L. Chen, "Identity-Based Conference Key Broadcast Systems", Proceedings of IEE Computers and Digital Techniques, Vol. 141, No. 1, pp.57-60, 1994.
17. I. Ingemarsson, D.T. Tang, C.K. Wong, "A Conference Key Distribution System", IEEE Transactions on Information Theory, Vol. IT-28, No. 5, pp.714-720, 1982.
18. B. Klein, M. Otten, T. Beth, "Conference Key Distribution Protocols in Distributed Systems", Proceedings of Codes and Ciphers-Cryptography and Coding IV, IMA, pp.225-242, 1995.
19. K. Koyama, "Secure Conference Key Distribution Schemes for Conspiracy Attack", Proceedings of Advances in Cryptology - Eurocrypt '92, Lecture Notes in Computer Science 658, Springer-Verlag, pp.449-453, 1992.
20. K. Koyama, K. Ohta, "Identity-Based Conference Key Distribution Systems", Proceedings of Advances in Cryptology - Crypto '87, Lecture Notes in Computer Science 293, Springer-Verlag, pp.175-184, 1987.
21. K. Koyama, K. Ohta, "Security of Improved Identity-Based Conference Key Distributioin Systems", Proceedings of Advances in Cryptology - Eurocrypt '88, Lecture Notes in Computer Science 330, Springer-Verlag, pp.11-19, 1988.
22. T. Matsumoto, H. Imai, "On the Key Predistribution System: A Practical Solution to the Key Distribution Problem", Proceedings of Advances in Cryptology - '87, Lecture Notes in Computer Science 293, Springer-Verlag, pp.185-193, 1987.
23. D. Pointcheval, J. Stern. "Security proofs for signatue schemes", Proceedings of Advances in Cryptology - Eurocrypt '96, Lecture Notes in Computer Science 1070, Springer-Verlag, pp.387-398, 1996.
24. R.A. Rueppel, P.C. Van Oorschot, "Modern Key Agreement Techniques", Computer Communications, 1994.
25. A. Shimbo, S.I. Kawamura, "Cryptanalysis of Several Conference Key Distribution Schemes", Proceedings of Advances in Cryptology - Asiacrypt '91, Lecture Notes in Computer Science 739, Springer-Verlag, pp.265-276, 1991.

26. V. Shoup, "Lower Bounds for Discrete Logarithms and Related Problems", Proceedings of Advances in Cryptology - Eurocrypt '97, Lecture Notes in Computer Science 1233, Springer-Verlag, pp.256-266, 1997.
27. D.G. Steer, L. Strawczynski, W. Diffie, M. Wiener, "A Secure Audio Teleconference System", Proceedings of Advances in Cryptology - Crypto '88, Lecture Notes in Computer Science 409, Springer-Verlag, pp.520-528, 1988.
28. W.G. Tzeng, "A Practical and Secure Fault-tolerant Conference-key Agreement Protocol", Proceedings of Public Key Cryptography - PKC 2000, Lecture Notes in Computer Science 1751, Springer-Verlag, pp.1-13, 2000.
29. T.C. Wu, "Conference Key Distribution System with User Anonymity Based on Algebraic Approach", Proceedings of IEE Computers and Digital Techniques, Vol. 144, No 2, pp.145-148, 1997.
30. Y. Yacobi, "Attack on the Koyama-Ohta Identity Based Key Distribution Scheme", Proceedings of Advances in Cryptology - Crypto '87, Lecture Notes in Computer Science 293, Springer-Verlag, pp429-433, 1987.