A Provable Authenticated Certificateless Group Key Agreement with Constant Rounds

Jikai Teng and Chuankun Wu

Abstract: **Group key agreement protocols allow a group of users, communicating over a public network, to establish a shared secret key to achieve a cryptographic goal. Protocols based on certificateless public key cryptography (CL-PKC) are preferred since CL-PKC does not need certificates to guarantee the authenticity of public keys and does not suffer from key escrow of identity-based cryptography. Most previous certificateless group key agreement protocols deploy signature schemes to achieve authentication and do not have constant rounds. No security model has been presented for group key agreement protocols based on CL-PKC. This paper presents a security model for a certificateless group key agreement protocol and proposes a constant-round group key agreement protocol based on CL-PKC. The proposed protocol does not involve any signature scheme, which increases the efficiency of the protocol. It is formally proven that the proposed protocol provides strong AKE-security and tolerates up to $n-2$ malicious insiders for weak MA-security. The protocol also resists key control attack under a weak corruption model.**

Index Terms: **Admissible pairing, certificateless public key cryptography (CL-PKC), group key agreement, insider attack, provable security.**

I. INTRODUCTION

In many group-oriented scenarios such as video conferences and collaborative applications, secure group communication is of great importance since most communication occurs over insecure networks. A group key agreement protocol enables a group of users to establish a shared secret key to achieve message confidentiality and integrity; thus, it plays an important role in achieving secure group communications. Among existing group key agreement protocols, asymmetric cryptographic technologies such as public key infrastructure (PKI), identity-based public key cryptography, and certificateless public key cryptography are commonly adopted. In typical PKI, there is a need for a certificate to provide an assurance relationship between a public key and the identity that holds the corresponding private key. Thus there are many issues associated with certificates, such as storage, revocation, distribution, and cost of validation. In 1984, Shamir introduced identity-based public key cryptography [1] to address the drawbacks of PKI. In an identity (ID)-based cryptosystem, users just have to know the public identity of other users such as e-mail address and telephone number, and everyone may use a public key directly without verifying it, which dramatically simplifies the public key management procedure as compared to a typical PKI. Unfortunately, an ID-based cryptosystem requires a trusted key generation center (KGC) to generate private keys for users. Thus, key escrow property is inherent. In 2003, Al-Riyami and Paterson [2] introduced certificateless public key cryptography that avoids the inherent key escrow of ID-based cryptography and does not require certification to guarantee the authenticity of public keys.

Related works: Since the first 2-party key agreement protocol [3] based on asymmetric cryptographic technique was proposed by Diffie and Hellman in 1976, there have been many works to generalize the protocol to a multiparty setting. Ingemarsson proposed the first group key agreement protocol [4] in 1982. However, there is no formal security analysis and authentication mechanism in [4]. Bresson $et\ al.$ proposed authenticated group key agreement protocols in [5]–[7] with formal provable security. However, protocols in [5]–[7] require $O(n)$ rounds to establish group session keys. TGDH [8] and Dutta [9] make use of key trees to generate session keys and need $O(lgn)$ rounds. In 2003, Katz and Yung [10] proposed a scalable compiler to transform any group key agreement protocol to an authenticated one and applied their compiler to protocol [11] proposed by Burmester and Desmedt, which presented a provably secure authenticated group key agreement with three rounds. In [12] and [13], two-round protocols are proposed with a formal security analysis. Protocols [14] and [15] also need constant rounds that take insider attack into consideration and thus provide stronger security guarantee. These protocols are under the deployment of PKI. There are also a lot of ID-based group key agreement protocols proposed. Barua [16] attempted to extend Joux's tripartite protocol to a multiparty key agreement. Reddy proposed an ID-based n-party key agreement protocol in [17] using one way hash function trees. Protocols proposed in [16] and [17] require $O(lgn)$ rounds. Protocol in [18] is efficient and requires constant rounds with formal security analysis. Since certificateless public key cryptography (CL-PKC) was introduced, there have been a lot of two-party key agreement schemes like [19] and [20] under the deployment of CL-PKC. However, group key agreement protocol based on CL-PKC is seldom studied. In 2007, S. Heo *et al.* [21] proposed the first certificateless group key agreement protocol. In 2008, E.-J. Lee *et al.* pointed out that their protocol did not provide forward security and proposed a certificateless group key agreement protocol [22] with forward security. Protocols in [21] and [22] need $O(lgn)$ rounds, and their security is analyzed in a

Manuscript received December 11, 2010; approved for publication by Victor Leung, Division III Editor, June 9, 2011.

This work is supported by the National Science Foundation of China (grants No. 60903210).

J. K. Teng is with the State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China and The Graduate University of Chinese Academy of Sciences, Beijing, 100049, China, email: tjikai@yahoo.com.cn.

C. K. Wu is with the State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China, email: chuankun.wu@gmail.com.

1229-2370/12/$10.00

heuristic way. Protocol [23] has constant rounds, and its security is formally analyzed. However, its security model is limited because the ability of an adversary in a certificateless group key agreement is not modeled in their security model. In [24], protocol [23] is found not to provide forward security, but the security of the protocol proposed in [24] is not analyzed formally. No security model has been constructed for a certificateless group key agreement protocol so far.

In a round, all messages may be sent simultaneously during the protocol execution and one needs not wait for the messages from other parties before sending out his/her own messages. Thus, it is desirable to minimize the number of rounds. Most previous certificateless group key agreement protocols do not have constant rounds. It is a trivial way of designing a constant-round certificateless group key agreement protocol by applying a compiler of Katz-Yung [10] using some certificateless signature scheme to an unauthenticated constant-round group key exchange; however, the signature and verification process is costly. To achieve insider security using signaturse, each user needs to verify the signatures of all other users, which considerably degrades the performance of the protocol.

Our contributions In this paper, we present a certificateless group key agreement with constant rounds. It does not involve a signature scheme to achieve mutual authentication, which improves the efficiency of the protocol. In order to analyze the security of the proposed protocol, we present a security model for certificateless group key agreement protocol. The protocol is formally proved to achieve strong AKE-security against active adversaries and tolerate up to $n-2$ malicious insiders for weak MA-security. The proposed protocol also resists key control attack in a weak corruption model.

II. PRELIMINARIES

This section briefly describes some notions related to the proposed protocol and introduces the notion of certificateless public key cryptosystem.

A. Related Notions

Admissible Pairing

Let $\mathbb{G}_1$ be a cyclic additive group of prime order q and $\mathbb{G}_2$ be a cyclic multiplicative group of the same order. Assume that the discrete logarithm problems in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are intractable.

Admissible pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ satisfying the following properties:

(1) **Bilinear**: $e(aP, bQ) = e(P, Q)^{ab}$ for any $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$.

(2) **Non-degenerate**: There exists $P \in \mathbb{G}_1$ such that $e(P, P)$ is of order q.

(3) **Computable**: There exits a polynomial time algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

Modified weil pairing [25] and tate pairing [26] are examples of admissible pairings.

Then, we consider the following assumptions.

BDH assumption

For $a, b, c \in \mathbb{Z}_q^*$, $P \in \mathbb{G}_1$, given (P, aP, bP, cP), computing $e(P, P)^{abc}$ is intractable.

m-BCAA1 assumption

For an integer m, and $x \in \mathbb{Z}_q^*$, $P \in \mathbb{G}_1$, given $(P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \cdots, (h_m, \frac{1}{h_m+x}P))$, where $h_i \in \mathbb{Z}_q^*$ are different from each other for $0 \le i \le m$, computing $e(P, P)^{\frac{1}{h_0+x}}$ is intractable.

Theorem 1. Given $P, aP, e(P, P)^b$ with $a, b \in \mathbb{Z}_q^*$, $P \in \mathbb{G}_1$, it is intractable to compute $e(P, P)^{ab}$.

Proof: Assume that given $P, aP, e(P, P)^b$, there is a polynomial time algorithm to compute $e(P, P)^{ab}$.
Given (P, aP, bP, cP) with $P \in \mathbb{G}_1, a, b, c \in \mathbb{Z}_q^*$, we can get $e(P, P)^{ab} = e(aP, bP)$. According to the assumption, $e(P, P)^{abc}$ can be computed with $e(P, P)^{ab}$ and cP, which is contrary to the bilinear Diffie-Hellman (BDH) assumption.

B. Certificateless Public Key Cryptosystem

In a certificateless cryptosystem, the operation to generate system parameters and long-term keys of users consists of the following algorithms:

SystemSetup: Takes as input a security parameter k, returns **params** (system parameters) and **msk** (master secret key). This algorithm is run by KGC.

Partial private key extract: Takes as input **params** and an arbitrary $ID \in \{0, 1\}^*$, returns a partial private key D_{ID}. This algorithm is run by KGC. KGC communicates the partial private key D_{ID} secretly to the user with identification ID.

Set secret value: Given **params** and a user's identification ID outputs a secret value x_{ID} for that identity. This algorithm is run once by the user.

Set private key: Takes as input **params**, a user's partial private key D_{ID}, and his secret value x_{ID}, returns a private key S_{ID} to that user. This algorithm is run once by the user.

Set public key: Given **params** and a user's secret value x_{ID} returns a public key pk_{ID} for that user. This algorithm is run once by the user and the resulting public key is widely and freely distributed.

III. SECURITY MODEL

In this section, we will describe the security model under which we prove the security of the proposed protocol. We extend the formal security model proposed in [27] that considers insider attack. In our model, there exists an adversary $\mathcal{A}$ who is assumed to control the network completely. The adversary may delay, replay, modify, interleave, delete, or redirect messages. Let $\mathbb{U} = \{u_1, \cdots, u_n\}$ be a set of users. At any point of time, any subset of $\mathbb{U}$ may decide to execute the protocol. A user may execute many protocol instances in parallel. The ith instance (also called oracle) of user u is denoted by Π_u^i. An instance Π_u^i enters the accepted state if it accepts a session key sk_u^i. Note that an instance may terminate without entering the accepted state. Every instance Π_u^i maintains internal state information denoted by state_u^i consisting of ephemeral secret values used during the protocol execution. A protocol is correct if all users have accepted a common session key with correctly formatted messages. The following notions will be required.

Session ID: Session ID for instance Π_u^i denoted by sid_u^i is a non-secret session identifier that servers as the identifier for instance i of user u.

Partner ID: Partner ID for instance Π_u^i is a set of users with whom user u intends to establish a session key in his i-th instance, including user u himself, denoted by pid_u^i.

Instances Π_u^i and Π_v^j are partnered if and only if $sid_u^i = sid_v^j$ and pid_u^i=pid_v^j.

At any time, the adversary can make the following queries in any order:

Send (Π_u^i, M): This query is used to send a message M to Π_u^i. It returns to the adversary what Π_u^i would have generated in processing M. If M is empty, this query initiates an execution of the protocol.

Reveal key (Π_u^i): This query is available to the adversary if oracle Π_u^i has accepted. The session key is output to the adversary.

Reveal partial private key (u): This query returns the partial private key of user u to the adversary.

Reveal secret value (u): This query returns the secret value of user u to the adversary.

Request public key (u): This oracle generates the public key of user u and returns it to the adversary.

Replace public key (u, pk): This query replaces the public key pk_u of user u with any value pk in the public key space. The new public key will be used for future communication and computation; however, user u will use his original private key for future computation.

Reveal state (Π_u^i): This query returns internal secret information state_u^i to the adversary.

Test (Π_u^i): A random bit b is generated. If b = 1, the session key is returned. Otherwise, a random value in the session key space is returned. **Test** query can be performed only once against an oracle that is fresh (see below).

After **Test** query, the adversary may continue to issue queries with the restriction that the test session remains fresh. At any point, the adversary may output its guess b'. We say that event **Succ** occurs if $b' = b$. The advantage of an adversary $\mathcal{A}$ in attacking a protocol is defined as $Adv_{\mathcal{A}}(k) = \mid \Pr[\textbf{Succ}] - \frac{1}{2} \mid$.

The security model distinguishes two types of adversaries: Type I adversary and type II adversary. Type I adversary is not allowed to have access to the master secret key. Type II adversary is equipped with the master secret key and can compute the partial private keys of all users. Neither type I adversary nor type II adversary is allowed to reveal the secret value of any identity if he has replaced the public key of that identity.

A user u is said to be corrupted if both **Reveal partial private key** (u) query and **Reveal secret value** (u) query or both **Reveal partial private key** (u) query and **Replace public key**(u, pk) query have been asked in the presence of type I adversary. In the presence of type II adversary, if **Reveal secret value** (u) query or **Replace public key** (u) query is asked, user u is corrupted, because type II adversary knows the partial private keys of all users. A user who is not corrupted is called an honest user.

Definition 1 (strong/weak corruption model). If an adversary is given access to **Send**, **Reveal key**, **Reveal secret value**, **Request public key**, **Reveal partial private key**, **Replace public key**, and **Test** queries, we say that the adversary operates in a weak corruption model. If he is additionally given access to the **Reveal state** query, he is said to operate in a strong corruption model.

In order to define a meaningful notion of security, we will first define *freshness*.

Definition 2. Instance Π_u^i is fresh if the following conditions are satisfied:

(1) Π_u^i has accepted a session key.
(2) Neither Π_u^i nor any of its partners has been asked the **Reveal key** query or **Reveal state** query.
(3) No user in pid_u^i has been corrupted.

Definition 3 (strong/weak AKE-security). A correct certificateless group key agreement protocol is AKE-secure if, for any probabilistic polynomial time (PPT) adversary $\mathcal{A}^{\mathrm{AKE}}$ (including type I adversary and type II adversary), $\mathrm{Adv}_{\mathcal{A}^{\mathrm{AKE}}}(k)$ is negligible in k. If the adversary is not allowed to issue the **Reveal state** query, the protocol provides weak AKE-security. Otherwise, it provides strong AKE-security.

Definition 4 (strong/weak MA-security). An adversary $\mathcal{A}^{\mathrm{MA}}$ (including type I adversary and type II adversary) violates the MA-security of a correct certificateless group key agreement protocol if at some point there exists an honest user u_i, whose oracle Π_i^s has accepted sk_i^s and another honest user $u_j \in pid_i^s$, such that

1. There is no instance Π_j^t with $(pid_i^s, sid_i^s) = (pid_j^t, sid_j^t)$ or
2. There is an instance Π_j^t with $(pid_i^s, sid_i^s) = (pid_j^t, sid_j^t)$ that has accepted with $sk_j^t \neq sk_i^s$.

The probability that adversary $\mathcal{A}^{\mathrm{MA}}$ successfully violates the MA-security of a protocol is denoted by $\mathrm{Succ}_{\mathcal{A}^{\mathrm{MA}}}(k)$. A correct certificateless group key agreement protocol is *MA-secure*, if for any PPT adversary $\mathcal{A}^{\mathrm{MA}}$, $\mathrm{Succ}_{\mathcal{A}^{\mathrm{MA}}}(k)$ is negligible in k. If the adversary is allowed to issue the **Reveal state** query, the protocol achieves strong MA-security; otherwise, it achieves weak MA-security.

IV. THE PROPOSED PROTOCOL

In this section, we will present a constant-round certificateless group key agreement protocol. The protocol consists of an initialization phase in which system parameters and long-term keys of users are generated and a key agreement phase in which the group session key is established. Detailed descriptions are given below.

Initialization:

System Setup: Given a security parameter k, KGC chooses two groups- $\mathbb{G}_1$ and $\mathbb{G}_2$- of prime order q, a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ as specified in section II-A, a random generator P of $\mathbb{G}_1$ and, computes $g = e(P, P) \in \mathbb{G}_2$. In addition, it chooses hash functions $H_1 : \{0,1\}^* \longrightarrow \mathbb{Z}_q^*, H_2 : \mathbb{G}_2 \times \{0,1\}^* \longrightarrow \{0,1\}^k, H_3 : \{0,1\}^* \longrightarrow \{0,1\}^l$, with $l > |q|$, where $|q|$ denotes the binary length of q. Then, KGC chooses $s \in \mathbb{Z}_q^*$ as the master secret key and sets $P_{\mathrm{pub}} = sP$ as its public key. KGC publishes system parameters **params** as $\{\mathbb{G}_1, \mathbb{G}_2, e, P, P_{\mathrm{pub}}, g, H_1, H_2, H_3\}$.

Partial private key extract: Takes as input **params** and $ID_i \in \{0,1\}^*$, with ID_i being the identification of user u_i, KGC computes the partial private key of user u_i as $D_i = (q_i + s)^{-1}P$, where $q_i = H_1(ID_i)$. Then, KGC secretly communicates D_i secretly to u_i.

Set secret value: Takes **params** and $ID_i \in \{0,1\}^*$ as input; returns a random value $x_{ID_i} \in \mathbb{Z}_q^*$.

Set private key: Takes **params**, D_i, and x_{ID_i} as input; returns $S_i =< x_{ID_i}, D_i >$.

Set public key: Takes **params** and, x_{ID_i} as input; returns $P_i = g^{x_{ID_i}}$.

Key Agreement:

Let $\mathbb{U} = \{u_1, \cdots, u_n\}$ be a set of users who want to establish a session key.

Round 1: Each user $u_i (1 \le i \le n)$ chooses random $r_i \in \mathbb{Z}_q^*, k_i \in \{0,1\}^k$ and computes $P_{i,j} = r_i(q_j P + P_{\text{pub}}) = r_i(q_j + s)P (1 \le j \le n, j \neq i)$. Then, u_i broadcasts $P_{i,j} (1 \le j \le n, j \neq i)$ and $H_3(k_i)$, keeping r_i, k_i secret.

Round 2: Upon receiving $P_{j,i}$ and $H_3(k_j)$, user u_i sets $sid_i^w = H_3(k_1) \| \cdots \| H_3(k_n)$ and computes $t_{j,i} = e(P_{j,i}, D_i)^{x_{ID_i}} P_j^{r_i} = g^{r_j x_{ID_i} + r_i x_{ID_j}}$, $V_{j,i} = H_2(t_{j,i} \| sid_i^w)$, and $K_{j,i} = V_{j,i} \oplus k_i$. Then, user u_i broadcasts $K_{j,i} (1 \le j \le n, j \neq i)$.

Key Computation: Upon receipt of $K_{i,j}$, user $u_i (1 \le i \le n)$ obtains $\widetilde{k}_j = V_{j,i} \oplus K_{i,j}$ and checks whether $H_3(\widetilde{k}_j) = H_3(k_j)$ holds $(1 \le j \le n, j \neq i)$. If the check process is invalid, u_i terminates the protocol. Otherwise, he computes the session key as $sk_i^w = H_3(k_1 \| \cdots \| k_n \| sid_i^w \| pid_i^w)$.

Since $t_{i,j} = t_{j,i}$, we have $V_{i,j} = V_{j,i}$. Therefore, each user u_i can obtain k_j from $K_{i,j}$ and compute the same session key. If the public key of user u_i is replaced, he will use his original private key for future computation. It follows that $V_{j,i} \neq V_{i,j}$. In this case, each user j $(1 \le j \le n, j \neq i)$ does not get the correct k_i, which can be detected in the check process before the session key computation.

V. SECURITY ANALYSIS

In this section, we prove the security of the proposed protocol under the security model described in Section III.

Theorem 2. The proposed protocol provides strong AKE-security against an active adversary, provided that H_1 and H_2 are random oracles and the m-BCCA1 and BDH assumptions are sound.

Proof: Assume that there is an adversary $\mathcal{A}^{\mathcal{AKE}}$ including type I and type II adversary against the protocol with non-negligible advantage ε. We will construct an algorithm F to solve the $(q_1 - 1)$-BCAA1 problem by applying type I adversary, and the problem mentioned in **Theorem 1** by applying type II adversary with non-negligible advantage, where q_1 is the number of queries to H_1 oracle. Since the replace public key attack does not work in our protocol, we do not consider it in our proof. Let $\mathbb{U} = \{u_1, u_2, \cdots, u_n\}$ be the set of users chosen by the adversary. Any subset of $\mathbb{U}$ may run the protocol.

For type I adversary, challenger F is given an instance of the $(q_1 - 1)$-BCAA1 problem $(\mathbb{G}_1, \mathbb{G}_2, e, P, sP, h_0, (h_1, \frac{1}{h_1+s}P), \cdots, (h_{q_1-1}, \frac{1}{h_{q_1-1}+s}P))$. F tries to compute $e(P,P)^{\frac{1}{h_0+s}}$. At the beginning of the game, F gives $\{\mathbb{G}_1, \mathbb{G}_2, e, P, P_{\text{pub}} = sP, g, H_1, H_2, H_3\}$ to the adversary as public parameters. Then, F simulates the queries as follows:

H_1**-queries** (ID_i): F maintains a list, referred to as the H_1^{list}, consisting of tuples of the form $(ID_i, h_i, x_{ID_i}, D_i)$. The list is initially empty. When $\mathcal{A}^{\text{AKE}}$ queries H_1 oracle on ID_i, F responds as follows:

If ID_i already appears in a tuple $(ID_i, h_i, x_{ID_i}, D_i)$ in H_1^{list}, F responds with $H_1(ID_i) = h_i$.

Otherwise, if the query is on ID_I, F chooses random $x_{ID_I} \in \mathbb{Z}_q^*$, stores $(ID_I, h_0, x_{ID_I}, \perp)$ into H_1^{list}, and responds with $H_1(ID_I) = h_0$.

Otherwise, F chooses a random value h_i from the given instance, and $x_{ID_i} \in \mathbb{Z}_q^*$ which have not been chosen, and stores $(ID_i, h_i, x_{ID_i}, \frac{1}{h_i+s}P)$ into H_1^{list}. Then, F responds with $H_1(ID_i) = h_i$.

H_2**-queries** (T_i): The challenger F maintains a list called the H_2^{list}. Each entry is a tuple of the form (T_i, K_i). F responds as follows:

If (T_i, K_i) already appears in H_2^{list}, F responds with $H_2(T_i) = K_i$.

Otherwise, F randomly chooses new $K_i \in \{0,1\}^k$, returns K_i as the response, and inserts (T_i, K_i) into H_2^{list}.

Send (Π_j^t, M): If M is not empty, F answers this query according to the description of the security model. If M is empty, F looks through the list H_1^{list} and checks whether ID_j and each identity who has an instance partnered with Π_j^t are in H_1^{list}. If $ID_{j_1}, \cdots, ID_{j_m}$ are not in the list, F queries $H_1(ID_{j_1}), \cdots, H_1(ID_{j_m})$.

If $\Pi_j^t \neq \Pi_i^w$, which is partnered with Π_I^v, F chooses random $r_j \in \mathbb{Z}_q^*$ and $k_j \in \{0,1\}^k$, which have not been previously chosen, and responds with $P_{j,l} = r_j(h_l + s)P (1 \le l \le n, l \neq j)$ and $H_3(k_j)$.

If $\Pi_j^t = \Pi_i^w$, F randomly chooses new $r_i \in \mathbb{Z}_q^*$, $k_i \in \{0,1\}^k$ and responds with $P_{i,l} = r_i(h_l + s)P (1 \le l \le n, l \neq i, I), P_{i,I} = r_i P$, and $H_3(k_i)$. This is indistinguishable from the adversary's view since r_i is randomly chosen from $\mathbb{Z}_q^*$. According to the protocol, $t_{i,I} = e(P,P)^{r_i \frac{x_{ID_I}}{h_0+s}} e(P,P)^{r_I x_{ID_i}}$

Reveal key (Π_i^t): F maintains an initially empty list Λ. Each entry of Λ is a tuple of the form (Π_k^v, sk_k^v).

If Π_i^t already appears in a tuple (Π_i^t, sk_i^t) in the list Λ, F responds with sk_i^t.

Otherwise, if one of the partners Π_j^s of Π_i^t already appears in a tuple (Π_j^s, sk_j^s) in Λ, F responds with sk_j^s.

Otherwise, F chooses random $sk_i^t \in \{0,1\}^l$, which has not been chosen previously, and responds with sk_i^t. Then, it inserts (Π_i^t, sk_i^t) into Λ.

Reveal partial private key (ID_i): F looks through H_1^{list}. If ID_i is not in the list, F queries $H_1(ID_i)$. Then, F checks whether $D_i = \perp$. If $D_i = \perp$, F aborts the game. Otherwise, F returns $\frac{1}{h_i+s}P$.

Reveal secret value (ID_i): F looks through H_1^{list}. If ID_i is not in the list, F queries $H_1(ID_i)$. Then, it returns x_{ID_i} as the response.

Reveal state (Π_j^t): F returns state_j^t to the adversary.

Request public key (ID_i): F looks through H_1^{list}. If ID_i is not in the list, F queries $H_1(ID_i)$. Then, F returns $g^{x_{ID_i}}$ as the response.

Test (Π_i^s): F aborts the game if one of the followings occurs: $\Pi_i^s \neq \Pi_I^v$ and Π_i^s is not partnered with Π_I^v; there exists a user $u_j \in pid_I^v$ who has been corrupted; Π_I^v or any of its partners has been asked the **Reveal key** query or **Reveal state** query; otherwise, F randomly chooses a value in $\{0,1\}^l$ as the response.

When the adversary finishes all queries and returns its guess, F chooses a random value $T_j = t_{j,k} \| sid_k^t$ from H_2^{list} and computes $D = (t_{j,k}/g^{x_{ID_i} r_I})^{r_i^{-1} x_{ID_I}^{-1}}$ as the solution of the given (q_1-1)-BCCA1 challenge.

Claim 1: *Let H denote the event that $H_2(t_{i,I} \| sid_I^v)$ query has been asked; we have* $\Pr[H] \geq 2\varepsilon$.

Proof: As H_2 is a random oracle, we have $\Pr[\textbf{Succ} \mid \neg H] = \frac{1}{2}$. By assumption, we have $\mid \Pr[\textbf{Succ}] - \frac{1}{2} \mid \geq \varepsilon$. Since $\Pr[\textbf{Succ}] = \Pr[\textbf{Succ} \mid H]\Pr[H] + \Pr[\textbf{Succ} \mid \neg H]\Pr[\neg H] \leq \Pr[\textbf{Succ} \mid \neg H]\Pr[\neg H] + \Pr[H] = \frac{1}{2}\Pr[\neg H] + \Pr[H] = \frac{1}{2} + \frac{1}{2}\Pr[H]$ and $\Pr[\textbf{Succ}] \geq \Pr[\textbf{Succ} \mid \neg H]\Pr[\neg H] = \frac{1}{2} - \frac{1}{2}\Pr[H]$, we have $\varepsilon \leq |\Pr[\textbf{Succ}] - \frac{1}{2}| \leq \frac{1}{2}\Pr[H]$. It follows that $\Pr[H] \geq 2\varepsilon$.

F aborts the game if one of the following events happens: $E1$, the adversary has asked the **Reveal partial private key** (u_I) query; $E2$, the adversary has issued the **Revealstate** (Π_j^t) query or **Reveal key** (Π_j^t) query, where $\Pi_j^t = \Pi_I^v$ or Π_j^t is partnered with Π_I^v; $E3$, there exists a user $u_j \in pid_I^v$ who has been corrupted; $E4$, the adversary does not choose Π_I^v or any of its partners as a challenge fresh oracle. If the adversary chooses Π_I^v or any of its partners as a challenge fresh oracle, no user in pid_I^v is corrupted and no instance partnered with Π_I^v (including Π_I^v) has been asked the **Revealstate** query or **Reveal key** query. Hence, $\neg E4$ implies $\neg E2$ and $\neg E3$. Then, we have $\Pr[\neg E1 \wedge \neg E2 \wedge \neg E3 \wedge \neg E4] = \Pr[\neg E1 \wedge \neg E4] \geq \frac{1}{q_1 q_s}$, where q_s is the number of **Send** queries. The probability that F chooses $t_{i,I} \| sid_I^v$ correctly from H_2^{list} is $\frac{1}{q_2}$, with q_2 being the number of H_2 queries. Therefore, challenger F solves the given (q_1-1)-BCCA1 problem with probability being at least $\frac{2}{q_1 q_2 q_s}\varepsilon$.

For type II adversary, F gives public parameters $\{\mathbb{G}_1, \mathbb{G}_2, e, P, P_{\text{pub}} = sP, g, H_1, H_2, H_3\}$ and the master secret key s to the adversary at the beginning of the game. Given $(\mathbb{G}_1, \mathbb{G}_2, e, aP, g^b)$, F tries to compute $e(P,P)^{ab}$. F simulates the H_2 oracle, **Reveal key** oracle, and **Reveal state** oracle in the same way as for the type I adversary.F simulates the H_1, **Reveal secret value**, **Request public key**, and **Send** oracles in the following way:

H_1-**queries** (ID_i): F maintains a list, which is referred to as H_1^{list}, with tuples of the form (ID_i, h_i, x_{ID_i}). The list is initially empty. When $\mathcal{A}$ queries H_1 oracle on ID_i, F responds as follows:

If ID_i already appears in a tuple (ID_i, h_i, x_{ID_i}) in H_1^{list}, F responds with $H_1(ID_i) = h_i$.

Otherwise, if $ID_i \neq ID_I$, F chooses random $h_i, x_{ID_i} \in \mathbb{Z}_q^*$, which have not been chosen, and stores (ID_i, h_i, x_{ID_i}) into H_1^{list}. Then, F responds with $H_1(ID_i) = h_i$.

If the query is on ID_I, F chooses random $h_I \in \mathbb{Z}_q^*$, stores $(ID_I, h_I, \perp)$ into H_1^{list}, and responds with $H_1(ID_I) = h_I$.

Reveal secret value (ID_i): F looks through H_1^{list}. If ID_i is not in the list, F queries $H_1(ID_i)$. Then, it checks whether $x_{ID_i} = \perp$. If $x_{ID_i} \neq \perp$, it returns x_{ID_i} as the response. Otherwise, it aborts the game.

Request public key (ID_i): F looks through H_1^{list}. If ID_i is not in the list, F queries $H_1(ID_i)$. Then, it checks whether $x_{ID_i} = \perp$. If $x_{ID_i} \neq \perp$, F returns $g^{x_{ID_i}}$ as the response. Otherwise, it returns g^b.

Send(Π_j^t**, M):** If M is not empty, F answers this query according to the description of the security model. If M is empty, F looks through the list H_1^{list} and checks whether ID_j and every other identity who has an instance partnered with Π_j^t are in H_1^{list}. If $ID_{j_1}, \cdots, ID_{j_m}$ are not in the list, F queries $H_1(ID_{j_1}), \cdots, H_1(ID_{j_m})$.

If $\Pi_j^t \neq \Pi_i^w$, which is partnered with Π_I^v, F chooses random $r_j \in \mathbb{Z}_q^*$ and $k_j \in \{0,1\}^k$, which have not been chosen previously, and responds with $P_{j,l} = r_j(h_l + s)P, (1 \leq l \leq n, l \neq j)$ and $H_3(k_j)$.

If $\Pi_j^t = \Pi_i^w$, F randomly chooses new $r_i \in \mathbb{Z}_q^*$ and $k_i \in \{0,1\}^k$, returns $P_{i,l} = r_i(h_l + s)P, (1 \leq l \leq n, l \neq i, I)$, $P_{i,I} = r_i(h_I + s)aP$, and $H_3(k_i)$. This is indistinguishable from the adversary's view since r_i is randomly chosen from $\mathbb{Z}_q^*$. According to the protocol, $t_{i,I} = e(P,P)^{r_i ab + r_I x_{ID_i}}$.

Test (Π_j^t): F aborts the game if one of the followings occurs: $\Pi_j^t \neq \Pi_I^v$ and Π_j^t is not partnered with Π_I^v; there exists a user $u_j \in Pid_I^v$ whose secret value has been revealed to the adversary by being asked the **Reveal secret value** query; Π_I^v or any of its partners has been asked the **Reveal key** query or **Reveal state** query. Otherwise, F randomly chooses a value in $\{0,1\}^l$ as the response.

Once the adversary finishes all queries and returns its guess, F randomly chooses $T_j = t_{j,k} \| sid_k^t$ from H_2^{list} and computes $D = (t_{j,k}/g^{r_I x_{ID_i}})^{r_i^{-1}}$ as the solution of the given challenge.

Claim 2: *Let H' denote the event that $H_2(t_{i,I} \| sid_I^v)$ query has been asked; we have* $\Pr[H'] \geq 2\varepsilon$.

Proof: The proof of this claim is similar to the proof of **Claim 1**.

Challenger F aborts the game if one of the following events happens: $E'1$, $\mathcal{A}^{\text{AKE}}$ has asked the **Reveal secret value** (ID_I) query; $E'2$, $\mathcal{A}^{\text{AKE}}$ has asked the **Reveal state** (Π_j^t) or **Reveal key** (Π_j^t), where $\Pi_j^t = \Pi_I^v$ or Π_j^t is partnered with Π_I^v; $E'3$, there exists a user $u_j \in pid_I^v$ whose secret value has been revealed to the adversary by being asked the **Reveal secret value** query; $E'4$, $\mathcal{A}^{\text{AKE}}$ does not choose Π_I^v or any of its partners as a challenge fresh oracle. If the adversary chooses Π_I^v or any of its partners as a challenge fresh oracle, no user in pid_I^v is corrupted and no instance partnered with Π_I^v (including Π_I^v) has been asked the **Reveal state** query or **Reveal key** query. It follows that $\neg E'4$ implies $\neg E'2$ and $\neg E'3$. We have $Pr[\neg E'1 \wedge \neg E'2 \wedge \neg E'3 \wedge \neg E'4] = Pr[\neg E'1 \wedge \neg E'4] \geq \frac{1}{q_1 q_s}$, with q_s being the number of **Send** queries. The probability that F correctly chooses $t_{i,I} \| sid_I^v$ from H_2^{list} is $\frac{1}{q_2}$, with q_2 being the number of H_2 queries. Thus, F solves the given problem with probability being at least $\frac{2}{q_1 q_2 q_s}\varepsilon$.

Theorem 3. The proposed protocol achieves weak MA-security, provided that H_1 and H_2 are random oracles and the m-BCCA1 and BDH assumptions hold.

Proof: We assume that there are n users with at least two honest users, u_I and u_j, and the adversary successfully impersonates u_j with non-negligible probability ε in session Π_j^t partnered with Π_I^v. Then, we will construct an algorithm F to solve the (q_1-1)-BCAA1 problem and the problem mentioned in **Theorem 1** with non-negligible probability by respectively applying type I adversary and type II adversary, where q_1 is the number of queries to H_1 oracle.

For type I adversary, given an instance of the $(q_1 - 1)$-BCAA1 problem $(\mathbb{G}_1, \mathbb{G}_2, e, P, sP, h_0, (h_1, \frac{1}{h_1+s}P), \cdots, (h_{q_1-1}, \frac{1}{h_{q_1-1}+s}P))$, F tries to compute $e(P,P)^{\frac{1}{h_0+s}}$. At the beginning of the game, F gives $\{\mathbb{G}_1, \mathbb{G}_2, e, P, P_{\text{pub}} = sP, g, H_1, H_2, H_3\}$ to the adversary as public parameters. F simulates the H_1, H_2, **Reveal secret value**, **Request public key**, **Reveal key**, and **Send** queries and solves the given instance of the (q_1-1)-BCAA1 problem in the same way as in **Theorem 2**. F aborts the game if one of the following events happens: $EV1$, $\mathcal{A}^{\text{MA}}$ has asked the **Reveal partial private key** (u_I) query; $EV2$, u_j has been corrupted; $EV3$, the adversary does not impersonate Π_j^t partnered with Π_I^v, with u_I and u_j being honest. If the adversary impersonates Π_j^t partnered with Π_I^v, with u_I and u_j being honest, u_j is uncorrupted. Hence, $\neg EV3$ implies $\neg EV2$. Then, we have $\Pr[\neg EV1 \wedge \neg EV2 \wedge \neg EV3] = \Pr[\neg EV1 \wedge \neg EV3] \geq 1/(q_1 q_s n)$, where n is the number of participant users and q_s is the number of **Send** queries. The probability that F chooses $t_{i,I} \| sid_I^v$ correctly from H_2^{list} is $1/q_2$, with q_2 being the number of H_2 queries. Therefore, challenger F solves the given (q_1-1)-BCCA1 problem with probability at least $2\varepsilon/(q_1 q_2 q_s n)$.

For type II adversary, F gives public parameters $\{\mathbb{G}_1, \mathbb{G}_2, e, P, P_{\text{pub}} = sP, g, H_1, H_2, H_3\}$ and the master secret key s to the adversary at the beginning of the game. Given $(P, aP, e(P,P)^b)$, F tries to compute $e(P,P)^{ab}$. F simulates the H_1, H_2, **Reveal secret value**, **Request public key**, **Reveal key**, and **Send** queries, and finds the solution of the given problem in the same way as in **Theorem 2**. F aborts the game if one of the following events happens: $EV'1$, $\mathcal{A}^{\text{MA}}$ has asked the **Reveal secret value** (ID_I) query; $EV'2$, the secret value of user u_j has been revealed to the adversary by being asked the **Reveal secret value** query; $EV'3$, $\mathcal{A}^{\text{MA}}$ does not impersonate Π_j^t partnered with Π_I^v, with u_I and u_j being honest. If the adversary impersonates Π_j^t partnered with Π_I^v and users u_I and u_j are honest, u_j is uncorrupted. It follows that $\neg EV'3$ implies $\neg EV'2$. Thus, we have $\Pr[\neg EV'1 \wedge \neg EV'2 \wedge \neg EV'3] = \Pr[\neg EV'1 \wedge \neg EV'3] \geq 1/(q_1 q_s n)$, with n being the number of users, q_s being the number of **Send** queries. The probability that F correctly chooses $t_{i,I} \| sid_I^v$ from H_2^{list} is $1/q_2$, where q_2 is the number of H_2 queries. Thus, F solves the given problem with probability being at least $2\varepsilon/(q_1 q_2 q_s n)$. Condition 1 in definition 4 holds. Therefore, at the time Π_I^v accepts, there exists a corresponding oracle Π_j^t, with $(pid_I^v, sid_I^v) = (pid_j^t, sid_j^t)$. According to condition 2 in definition 4, $\mathcal{A}^{\text{MA}}$ wins the game if Π_j^t and Π_I^v accepts with $sk_j^t \neq sk_I^v$. The construction of the session ID ensures that Π_j^t and Π_I^v hold the same $H_3(k_1)\|\cdots\|H_3(k_n)$. The probability that $\mathcal{A}^{\text{MA}}$ violates condition 2 in definition 4 is negligible since $sk_j^t = H_3(k_1\|\cdots\|k_n\|sid_j^t\|pid_j^t)$ and H_3 is a collusion resist hash function.

No Key Control: A protocol is said to resist key control attack if one can not determine the final session key. In the proposed protocol, the commitments of contributions to the session key are broadcasted. Thus, if the adversary is not allowed to ask the **Reveal state** query, every one has an equal contribution to the session key and no one can control it due to the one way hash function H_3 and the check process before session key computation. Thus, the proposed protocol resists key control in a weak corruption model.

Table 1. Comparision of complexity.

	R	Msize	Comp
[21]	lgn	$O(n) \mid P \mid$	$O(n)p + O(n)M$ $+O(nlgn)E$
[22]	lgn	$O(n) \mid P \mid$	$O(n)p + O(n)M$ $+O(nlgn)E$
[23]	2	$O(n^2) \mid q \mid +O(n) \mid P \mid$	$O(n)p + O(n^2)M$ $+O(1)E$
[24]	2	$O(n^2) \mid P \mid$	$O(n)p + O(n^2)M$ $+O(n)E$
Ours	2	$O(n^2) \mid P \mid$	$O(n^2)p + O(n)M$ $+O(n)E$

VI. COMPLEXITY ANALYSIS AND COMPARISON

In this section, we will compare the proposed protocol with certificateless group key agreement protocols in [21]–[24]. We will use the following notations.

R: The total number of rounds.
Msize: The total size of transmitted messages.
Comp: The total number of computations.
p: The cost of paring computations.
M: The cost of scalar multiplication in $\mathbb{G}_1$.
E: The cost of modular multiplication and modular exponentiation in $\mathbb{G}_2$.
$|q|$: The length of the element in $\mathbb{Z}_q^*$.
$|P|$: The length of the point in $\mathbb{G}_1$.
n: The number of group members.

As shown in Table 1, our protocol needs more pairing computation and message transmissions as compared to protocols [21] and [22]. However, our protocol needs constant rounds, that is, two broadcast rounds to establish a session key and no signature is involved, which improves the efficiency of the entire protocol. Protocol [24], with two rounds, where there is one broadcast round and protocol [23] with two broadcast rounds need less pairing computations and less computations in $\mathbb{G}_2$ as compared to the proposed protocol. However, it is possible to make the number of paring computations of the proposed protocol achieve $2n$ if we do not consider insider attack. The security of [21], [22] and [24] is not analyzed formally. The security of protocol [23] is analyzed formally, but its security model is limited and it does not provide forward security.

VII. CONCLUSIONS

In this paper, we present a security model for a certificateless group key agreement protocol and propose a group key agreement based on CL-PKC with constant rounds. It does not use a signature scheme to achieve mutual authentication, making the protocol more scalable. The proposed protocol is formally proven to provide strong AKE-security and weak MA-security in random oracle model. It also resists key control attack in weak corruption model.

REFERENCES

[1] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. CRYPTO'84*, Santa Barbara, California, USA, Aug. 1984, pp.47–53.

[2] S. Al-Riyami and K. Paterson, "Certificateless public key cryptography," in *Proc. ASIACRYPT 2003*, Taibei Taiwan, Nov.–Dec. 2003, pp. 452–473.

[3] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[4] I. Ingemarsson, D. T. Tang, and C. K. Wong, "A conference key distribution system," *IEEE Trans. Inf. Theory*, vol. 28, no. 5, pp.714–720, Sept. 1982.

[5] E. Bresson, O. Chevassut, and D. Pointcheval "Provably authenticated group Diffie-Hellman key exchange," in *Proc. CCS*, Philadelphia, Pennsylvania, USA, Nov. 2001, pp. 255–264.

[6] E. Bresson, O. Chevassut, and D. Pointcheval, "Dynamic group Diffie-Hellman key exchange under standard assumptions," in *Proc. Eurocrypt*, Amsterdam, The Netherlands, Apr.–May. 2002, pp. 321–336.

[7] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably authenticated group Diffie-Hellman key exchange-the dynamic case," in *Proc. Asiacrypt*, Gold Coast, Australia, Dec. 2001, pp. 290–309.

[8] Y. Kim, A. Perrig, and G. Tsudik, "Simple and fault-tolerant key agreement for dynamic collaborative groups," in *Proc. CCS*, Athens, Greece, Nov. 2000, pp. 235–244.

[9] R. Dutta and R. Barua, "Dynamic group key agreement in tree-based setting," in *Proc. ACISP*, Brisbane, Australia, July 2005, pp. 101–112.

[10] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," in *Proc. CRYPTO*, Santa Barbara, California, USA, Aug. 2003, pp. 110–125.

[11] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system," in *Proc. Eurocrypt*, Perugia, Italy, May 1994, pp. 275–286.

[12] R. Dutta and R. Barua, "Provably secure constant round contributory group key agreement in dynamic setting," *IEEE Trans. Inf. Theory*, vol. 54, no. 5, pp. 2007–2025, May 2008.

[13] H. J. Kim, S. M. Lee, and D. H. Lee, "Constant-round authenticated group key exchange for dynamic groups," in *Proc. Asiacrypt*, Jeju Island, Korea, Dec. 2004, pp. 245–259.

[14] E. Bresson and M. Manulis, "Securing group key exchange against strong corruptions," in *Proc. ASIACCS*, Tokyo, Japan, Mar. 2008, pp. 249–260.

[15] M. C. Gorantla, C. Boyd, and J. M. G. Nieto, "Modeling key compromise impersonation attacks on group key exchange protocols," in *Proc. PKC*, Irvine, CA, USA, Mar. 2009, pp. 105–123.

[16] R. Barua, R. Dutta, and P. Sarker, "Extending Joux's protocol to multiparty key agreement," in *Proc. Indocryopt*, New Delhi, India, Dec. 2003, pp. 205–217.

[17] K. C. Reddy and D. Nalla, "Identity-based authenticated group key agreement protocol," in *Proc. Indocryopt*, Hyderabad, India, Dec. 2002, pp. 215–233.

[18] K. Y. Choi, J. Y. Hwang, and D. H. Lee, "Efficient ID-based group key agreement with bilinear maps," in *Proc. PKC*, Singapore, Mar. 2004, pp. 130–144.

[19] T. K. Mandt and C. H. Tan, "Certificateless authenticated two-party key agreement protocols," in *Proc. ASIAN*, Tokyo, Japan, Dec. 2006, pp. 37–44.

[20] G. Lippold, C. Boyd, and J. G. Nieto, "Strongly secure certificateless key agreement," in *Proc. Pairing*, Palo Alto, CA, USA, Sept. 2009, pp. 206–230.

[21] S. Heo, Z. Kim, and K. Kim, "Certificateless authenticated group key agreement protocol for dynamic groups," in *Proc. GLOBECOM*, Washington, D.C, USA, Nov. 2007, pp. 464–468.

[22] E-J. Lee, S-E. Lee, and K-Y. Yoo, "A certificateless authenticated group key agreement protocol providing forward security," in *Proc. Int. Symp. Ubiquitous Multimedia Comput.*, Hobart, Australia, Oct. 2008, pp. 124–129.

[23] C. Cao, J. Ma, and S. Moon, "Provable efficient certificatelesss group key exchange," *Wuhan University J. Natural Sciences*, vol. 12, no. 1, pp. 41–45, Dec. 2007.

[24] M. Geng, F. Zhang, and M. Gao, "A secure certificateless authenticated group key agreement protocol," in *Proc. Int. Conf. Multimedia Inf. Netw. Security*, Wuhan, China, Nov. 2009, pp. 342–346.

[25] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proc. CRYPTO*, Santa Barbara, California, USA, Aug. 2001, pp. 213–229.

[26] P. S. L. M. Barreto, H. Y. Kim, and M. Scott, "Efficient algorithms for pairing based cryptosystems," in *Proc. Crypto*, Santa Barbara, California, USA, Aug. 2002, pp. 354–368.

[27] J. M. Bohli, M. I. G. Vasco, and R. Steinwandt, "Secure group key establishment revisited," *Int. J. Inf. Secur.*, vol. 6, no. 4, pp. 243–354, July 2007.

Jikai Teng received his B.S. degree in Science from Tianjin Normal University in 2003, and M.S. degree in Science from Guangzhou University in 2008. Currently, he is a Ph.D. Candidate in Information Security at the State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences. His research interests include group key agreement and broadcast encryption.

Chuankun Wu received his B.S. degree in Science from Qufu Normal University in 1985. He then received his M.S. degree in Science in 1988 and Ph.D. degree in 1994, both from Xidian University. He did his Postdoc at Queensland in 1995, was a Research Fellow at University of Western Sydney in 1997, and was a Senior Lecturer at the Australian National University. Since 2003, he has been a Professor at Institute of Software, Chinese Academy of Science. His research interests include group oriented cryptography and pairing based cryptography.