A Group Key Agreement With Efficient Communication for Ad Hoc Networks

Zongyu Song[1], Pengfei Cai[1], Jie Yang[2]
1. School of Computer Science and Engineering , ZhengZhou Normal University, Zhengzhou, China
2. School of Information Engineering, Wuhan University of Technology, Wuhan, China
Email: zongyusong@yahoo.com.cn; 13949066560@163; jieyang509@163.com

***Abstract*— In distributed ad hoc sensor networks, scalable group key agreement protocol plays an important role. They are designed to provide a group of users with a shared secret key such that the users can securely communicate with each other over a public network. In most of previous group key agreement protocols, the number of messages sent by all users increases with the number of all participants. In this paper, a dynamic authenticated group key agreement protocol is presented using pairing for ad hoc networks. In $Join$ algorithm, the number of transmitted messages does not increase with the number of all group members, which makes the protocol more practical. The protocol is provably secure. Its security is proved under Decisional Bilinear Diffie-Hellman assumption. The protocol also provides many other security attributes.**

***Index Terms*— Ad hoc networks, dynamic authenticated group key agreement, provable security, admissible pairing**

I. Introduction

[1] Ad hoc sensor network is a special type of network in which a set of mobile nodes may form a temporary network. In ad hoc network, all devices are able to establish direct communication with other devices that are within its communication range, and there is no centralizing entity such as the access point. Therefore, designing group key agreement protocols for such networks is a big challenge to achieve secure communication due to host mobility and lack of infrastructure.

Up to now, several key agreement protocols have been proposed, most of which are extensions of the first two-party key agreement protocol [1] proposed by Diffie and Hellman in 1976 and the tripartite key agreement protocol proposed by Joux in [2]. The security properties of key agreement protocols have been extensively studied. Tan [3] extended key agreement protocols to multi-server environments. In 1982, Ingemarsson [4] proposed the first group key agreement protocol. But no formal security analysis appears in their work. Formal security analysis of group key agreement protocol first appears in Bression et al. 's protocol [5]. The security model they adpoted is based on the 2-party key exchange models [6]. Protocols [4] and [5] require $O(n)$ rounds. Barua [7] attempted to extend Joux's tripartite protocol to multi-party one. Reddy proposed a group key agreement protocol in [8] under the employment of identity-based cryptosystem. There is no formal security analysis in [7], [8]. Dutta $et\ al.$ proposed a group key agreement with formal security analysis in [9]. However, protocols proposed in [7]–[9] require $O(lgn)$ communication rounds. In 2003, Katz and Yung [10] proposed a scalable compiler which can transform any group key agreement protocol to an authenticated one and and they obtained a three-round authenticated group key agreement with formal security analysis by applying their compiler to protocol [11]. Protocols in [12]–[14] are more efficient since they require two rounds. Protocols in [15], [16] require only one round to establish group session keys.

There are many two-party key agreement protocols [17], [18] and group key agreement protocols for ad hoc networks. In ad hoc network, the users are usually mobile. The group member is not known in advance and the users may join and leave the group very frequently. In such scenarios, dynamic group key agreement protocols are required. Such schemes must ensure that the group session key updates upon group member changing such that subsequent session keys are protected from the leaving members and previous session keys are protected from the joining members. There are quite a number of dynamic group key agreement protocols. Bresson $et\ al.$ improved the protocol [5] into dynamic group key agreement protocols in [19], [20]. However, in Bresson $et\ al.$'s protocols, $O(n)$ rounds are required in setup/join algorithms, so they are not suitable for ad hoc network. Protocols [21] and Dutta [22] require key trees to establish group session keys. In [23], [24], a ring structure among group members is considered. In such protocols, special ordering of the group members is required, which is not easily achieved in ad hoc networks. Since the mobile devices in ad hoc networks have limited resources and most cryptographic algorithms require expensive computations, the design of secure and efficient group key agreement protocols for ad hoc networks is one of the important problems. Protocol [25], [26] are constant round group key agreement protocols for mobile ad hoc networks.

Our Contributions In this paper, an efficient dynamic group key agreement(DAGKA) protocol for ad hoc networks is proposed. It is provably secure. Its security is proved in random oracle model under Decision Bilinear Deffie-Hellman assumption. It provides forward security and resists key control attack. In the proposed protocol,

[1]This work is supported by the Natural Science Research Project of Education Office of Anhui Province (KJ2013B185) and the Natural Science Research Project of Chuzhou University (2012kj001Z).

doi:10.4304/jsw.8.10.2406-2411

constant rounds are required and the number of transmitted messages in $Join$ algorithm increases only with the number of joining members. Therefore, it is suitable for ad hoc mobile networks.

II. Preliminary and Security Model

In this section, we will review some basic facts related to the proposed protocol and describe the security model in which the security of the proposed protocol is proved. Throughout the paper, we assume that $\mathbb{G}_1$ and $\mathbb{G}_2$ are cyclic multiplicative groups of prime order q , and the discrete logarithm problems in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are intractable.

A. Notions

Admissible Bilinear Pairing:

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

(1) **Bilinear:**
$\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ for any $g_1, g_2 \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$.

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

(3) **Computable** :
There exits a polynomial time algorithm to compute $\hat{e}(g_1, g_2)$for all $g_1, g_2 \in \mathbb{G}_1$.

Modified weil pairing [27] and tate paring [28] are examples of admissible pairings.

CMA-secure signature scheme

$\Sigma = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ is a signature scheme, where $\mathcal{K}, \mathcal{S}$ and $\mathcal{V}$ are key generation, signature and verification algorithms, respectively. If it is computationally infeasible that $\mathcal{A}$ generates a valid signature with any message under a chosen message attack, we say that the signature scheme is CMA-secure. Formally, let $Succ_{\Sigma,A}^{cma}$ be the success probability of $\mathcal{A}' s$ existential forgery under a chosen message attack against Σ. Σ is CMA-secure if $Succ_{\Sigma,A}^{cma}$ is negligible.

Bilinear Diffie-Hellman (BDH) Problem

BDH problem in ($\mathbb{G}_1, \mathbb{G}_2$, $\hat{e}$) is as follows: Given random $P \in \mathbb{G}_1$ and aP, bP, cP, where $a, b, c \in \mathbb{Z}_q^*$, compute $\hat{e}(P, P)^{abc}$. Formally, given a tuple (P, aP, bP, cP), where $P \in \mathbb{G}_1, a, b, c \in \mathbb{Z}_q^*$, we say that the advantage of an algorithm $\mathbf{A}$ in solving BDH problem is $Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e},\mathbf{A}}^{BDH} = Pr[\hat{e}(P, P)^{abc} \leftarrow \mathbf{A}(P, aP, bP, cP)]$. Let $Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e}}^{BDH} = Max_{\mathbf{A}}\{Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e},\mathbf{A}}^{BDH}\}$.

BDH assumption means that any probabilistic polynomial time (PPT) algorithm $\mathbf{A}$ has negligible advantage in solving BDH problem,i.e. $Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e}}^{BDH}$ is negligible.

Decisional Bilinear Diffie-Hellma (DBDH) Problem

DBDH problem in $(\mathbb{G}_1, \mathbb{G}_2, \hat{e})$ is as follows: Given random $P \in \mathbb{G}_1$ and aP, bP, cP, dP with $a, b, c, d \in \mathbb{Z}_q^*$, distinguish between tuples of the form $(P, aP, bP, cP, \hat{e}(P, P)^{abc})$ and $(P, aP, bP, cP, \hat{e}(P, P)^d)$. Formally, given a tuple (P, aP, bP, cP, dP), where $P \in \mathbb{G}_1, a, b, c, d \in \mathbb{Z}_q^*$, we say that the advantage of an algorithm $\mathbf{A}$ in solving DBDH problem is $Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e},\mathbf{A}}^{DBDH} =| \ Pr[\mathbf{A}(P, aP, bP, cP, e(P, P)^{abc}) = 1]$-$Pr[\mathbf{A}(P, aP, bP, cP, e(P, P)^d) = 1] \ |$. Let $Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e}}^{DBDH} = Max_{\mathbf{A}}\{Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e},\mathbf{A}}^{DBDH}\}$

DBDH assumption means that any PPT algorithm $\mathbf{A}$ has negligible advantage in solving DBDH problem, i.e. $Adv_{\mathbb{G}_1,\mathbb{G}_2,\hat{e}}^{DBDH}$ is negligible.

B. Security Model

In the model, there exists an adversary $\mathcal{A}$ which is assumed to control the network completely. The adversary is not a group member. The adversary may delay, replay, modify, interleave, delete or redirect messages. At any time, the adversary can make the following queries:

Send(m): This query allows the adversary to make the user run the protocol normally. This query returns to the adversary the result that an honest user would generate if the message **m** is sent according to the protocol rules.

Join$(\mathbb{U}, \mathbb{J})$: This query models the insertion of a set of users in $\mathbb{J}$ in the current group $\mathbb{U}$. The output of this query is the transcript generated by the invocation of algorithm **Join**$(\mathbb{U}, \mathbb{J})$. This query is initiated by a $Send$ query.

Leave$(\mathbb{U}, \mathbb{L})$: This query models the removal of users in $\mathbb{L}$ from the current group $\mathbb{U}$. It returns the transcript generated by the invocation of algorithm **Leave**$(\mathbb{U}, \mathbb{L})$. This query must be initiated by a $Send$ query.

Reveal $(\prod_u^i)$**:** This query models the attacks resulting in the session key being revealed. It is available to the adversary if the oracle $\prod_u^i$ has accepted (see below). The session key is output to the adversary.

Corrupt(ID_u)**:** This query outputs the long-term private key of user u to the adversary. But it does not output any internal data of user u.

Test$(\prod_u^i)$**:** 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 which is fresh (see below).

An oracle may be in one of the following states:

Accepted: The oracle decides to accept the session key after receiving properly formatted messages.

Rejected: The oracle aborts the run of the protocol.

Opened: A $Reveal$ query has been performed against the oracle for its last run of the protocol.

C. Security Notions

Session IDs and Partnering : Session ID for instance $\prod_u^i$ is defined as the concatenation of all messages sent and received by instance $\prod_u^i$, denoted by $SID(\prod_u^i)$. Partner ID for instance $\prod_u^i$ is a set containing the identities of all users with whom $\prod_u^i$ intends to establish a session key including user u himself, denoted by $PID(\prod_u^i)$. $\prod_u^i$ and $\prod_v^j$ are partnered if and only if $SID(\prod_u^i)$=$SID(\prod_v^j)$ and $PID(\prod_u^i)$=$PID(\prod_v^j)$

Freshness : Instance $\prod_u^i$ is fresh if the following conditions are satisfied:

(1) $\prod_u^i$ has accepted a session key .
(2) The adversary has not asked $Reveal(\prod_u^i)$ query or $Reveal(\prod_v^j)$ query, where $\prod_u^i$ and $\prod_v^j$ are partnered.
(3) No user $u \in PID(\prod_u^i)$ has been asked for a $Corrupt$ query prior to a query of the form $Send(\prod_v^j, \mathbf{m})$, where $\prod_u^i$ and $\prod_v^j$ are partnered.

Definition of Security The security of a protocol is defined by the following game played between the adversary and an infinite set of instances for $ID_i \in \mathbb{U}$.

(1) Firstly, long-term keys are assigned to each user in the initialization phase.
(2) Then, the adversary will interact with the oracles through queries above and get answers from the corresponding oracles.
(3) At some point, the adversary decides to make a $Test$ query to a fresh oracle. The adversary $\mathcal{A}$ outputs a random bit $b^{'}$ in answering this query.

We say that event **Succ** occurs if $b^{'}$=b. The advantage of $\mathcal{A}$ in attacking protocol $\mathcal{P}$ is defined as $Adv_{A,\mathcal{P}}(k) =|$ **Pr[Succ]** $- \frac{1}{2} |$.
A protocol is secure if $Adv_{A,\mathcal{P}}(k)$ is negligible with respect to security parameter k.

III. Protocol

In this section, we will propose an authenticated group key agreement protocol for ad hoc networks. Each user i holds a pair of signature/verification key (SK_i, PK_i). $\Sigma = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ is a CMA-secure signature scheme. In order to explain our protocol, we firstly describe the following algorithm to generate system parameters

Generation: Given a security parameter $k \in \mathbb{Z}^+$, the algorithm works as follows:

On input k, output a prime q, two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q, an admissible bilinear map $\hat{e}$:$\mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$, a generator g of $\mathbb{G}_1$ and $h \in \mathbb{G}_1$.

Let $H : \{0,1\}^* \longrightarrow \{0,1\}^l$ and $H_1 : \{0,1\}^l \longrightarrow \mathbb{Z}_q^*$ be one-way hash functions with $l \geq| q |$. The group session key space belongs to $\{0,1\}^l$. We also assume that the member in a group with the maximum index is the group leader.

The protocol is as follows:

Setup Let $\mathbb{U}_0$ be an initial group with $\mathbb{U}_0$=$\{u_1, u_2, ..., u_n\}$, $I_0 = ID_{u_1} \parallel ... \parallel ID_{u_n}$

Round 1 Each user $i(1 \leq i \leq n-1)$ randomly chooses $k_i \in \{0,1\}^l, r_i \in \mathbb{Z}_q^*$, computes $y_i = g^{r_i}$ and the signature δ_i on $y_i \parallel k_i$ using sk_i. Then it broadcasts $\delta_i \parallel y_i \parallel k_i$ keeping r_i secret.

Round 2 User n checks the signatures δ_i on $y_i \parallel k_i$ using $pk_i(1 \leq i \leq n-1)$. If one of the verifications fails, it aborts the protocol. Otherwise, it chooses random $r_n \in \mathbb{Z}_q^*, k_n \in \{0,1\}^l$ and computes $V_j = g^{r_n r_j}$, $W_n = H(\hat{e}(g,h)^{r_n + r_1 r_n + ... + r_{n-1} r_n}) \oplus k_n, U_n = H(k_n \| ID_0)$. Then it computes the signature δ_n on $V_1\|...\|V_{n-1}\|W_n$ using sk_n. Subsequently, it broadcasts $\delta_n\|V_1\|...\|V_{n-1}\|W_n\|U_n$.

Key Computation Each user $i(1 \leq i \leq n-1)$ firstly verifies the signature δ_n using pk_{u_n}. If the verification fails, it aborts the protocol. Otherwise it computes $s_i = \hat{e}(h, V_i)^{r_i^{-1}} \hat{e}(h, V_1 + ... + V_{n-1})$. Then user i computes $\widetilde{k}_n = H(s_i) \oplus W_n$ and checks whether $H(\widetilde{k}_n \parallel ID_0) = U_n$ holds. If the check process is valid, it computes the final session key $sk = H(k_1 \parallel ... \parallel k_n \parallel ID_0)$. User n can compute the session key directly.

Post Computation User $i(1 \leq i \leq n)$ computes and stores $x = H_1(sk)$.

Join Let $\mathbb{U}_{v-1} = \{u_1, u_2, ..., u_n\}$ be the current group, $\mathbb{J} = \{u_{n+1}, u_{n+2}, ..., u_{n+m}\}$ be the set of users who will join the group $\{u_1, u_2, ..., u_n\}$, $U_v = \{u_1, ..., u_{n+m}\}, ID_v = ID_{u_1} \parallel ... \parallel ID_{u_{n+m}}$

The protocol is as follows:

Round 1 User n randomly chooses new $k_n \in \{0,1\}^l$, sets $r_n = x$, computes $y_n = g^x$ and the signature δ_n on $y_n \parallel k_n$ using sk_n. Then it broadcasts $\delta_n \parallel y_n^{'}\|k_n^{'}$ keeping x secret. Each user $n+i$ $(1 \leq i \leq m-1)$ chooses random $k_{n+i} \in \{0,1\}^l, r_{n+i} \in \mathbb{Z}_q^*$, computes $y_{n+i} = g^{r_{n+i}}$. Then he computes the signature δ_{n+i} on $y_{n+i}\|k_{n+i}$ and broadcasts $\delta_{n+i}\|y_{n+i}\|k_{n+i}$.

Round 2 User $n+m$ first verifies the signatures δ_{n+i} on $y_{n+i} \parallel k_{n+i}$ using $pk_{n+i}(0 \leq i \leq m-1)$. If one of the verifications fails, it aborts the protocol. Otherwise, he chooses random $r_{n+m} \in \mathbb{Z}_q^*, k_{n+m} \in \{0,1\}^l$ and computes $V_{n+j} = g^{r_{n+m} r_{n+j}} (0 \leq j \leq n+m-1)$ $W_{n+m} = H(\hat{e}(g,h)^{r_{n+m} + r_1 r_{n+m} + ... + r_{n-1} r_{n+m-1}}) \oplus k_{n+m}, U_{n+m} = H(k_{n+m} \parallel ID_v)$. Then it computes the signature δ_{n+m} on $V_n\|...\|V_{n+m-1}\|W_{n+m}$ using sk_{n+m}. Subsequently, it broadcasts $\delta_{n+m}\|V_n\|...\|V_{n+m}\|W_{n+m}\|U_{n+m}$.

Key Computation Each user $i(1 \leq i \leq n+m-1)$ firstly verifies the signature δ_{n+m} using $pk_{u_{n+m}}$. If the verification fails, it aborts the protocol. Otherwise each user $i(1 \leq i \leq n)$ computes $s_i = \hat{e}(h, V_n)^{r_n^{-1}} \hat{e}(h, V_n + ... + V_{n+m})$. User $n+i(1 \leq j \leq m-1)$ computes $s_{n+i} = \hat{e}(h, V_{n+i})^{r_{n+i}^{-1}} \hat{e}(h, V_n + V_{n+1} + ... + V_{n+m})$. Then user $i(1 \leq i \leq n+m-1)$ computes $\widetilde{k}_{n+m} = H(s_i) \oplus W_{n+m}$ and checks whether $H(\widetilde{k}_{n+m}\|ID_v) = U_{n+m}$ holds. If the check process is valid, it computes the final session key $sk = H(k_n \parallel ... \parallel k_{n+m} \parallel ID_v)$. User $n+m$ can compute the session key directly.

Post Computation All users compute and store $x = H_1(sk)$.

Leave Without loss of generality, we assume that $\mathbb{U}_{v-1} = \{u_1, u_2, ..., u_n\}$ is the current group and that $\mathbb{L} = \{u_{m+1}, u_{m+2}, ..., u_n\}$ is the set of leaving users. Then $\mathbb{U}_v = \{u_1, u_2, ..., u_m\}, ID_v = ID_{u_1} \parallel ... \parallel ID_{u_m}$

Round 1 Each user $i(1 \leq i \leq m-1)$ randomly chooses $k_i \in \{0,1\}^l, r_i \in \mathbb{Z}_q^*$, computes $y_i = g^{r_i}$ and $\delta_i = S(sk_i, y_i\|k_i)$. Then it broadcasts $\delta_i \parallel y_i \parallel k_i$ keeping r_i secret.

Round 2 User m verifies the signatures δ_i on $y_i\|k_i$ using $pk_i(1 \leq i \leq m-1)$. If one of the verifications fails, it aborts the protocol. Otherwise, it chooses random $r_m \in \mathbb{Z}_q^*, k_m \in \{0,1\}^l$ and computes $V_j = g^{r_m r_j}(1 \leq j \leq m-1)$, $W_m = H(\hat{e}(g,h)^{r_m + r_1 r_m + ... + r_{m-1} r_m}) \oplus k_m, U_m = H(k_m\|ID_v)$. Then it computes the signature δ_m on $V_1\|...\|V_{m-1}\|W_m$ using sk_m. Finally, it broadcasts

$\delta_m \parallel V_1\|...\|V_{m-1}\|W_m\|U_m$.

Key Computation Each user $i(1 \leq i \leq m-1)$ first checks the correctness of the signature δ_m on $V_1\|...\|V_{m-1}\|W_m$. If the check process is invalid, it aborts the protocol. Otherwise, it computes $s_i = \hat{e}(h, V_i)^{r_i^{-1}}\hat{e}(h, V_1 + ... + V_{m-1})$. Then user i computes $\tilde{k}_m = H(s_i) \oplus W_m$ and checks whether $H(\tilde{k}_m \parallel ID_v) = U_m$ holds. If it holds, it computes the final session key $sk = H(k_1\|...\|k_m \parallel ID_v)$. User m can compute the session key directly.

Post Computation Each user computes and stores $x = H_1(sk)$.

IV. Security Analysis of The Proposed Protocol

In this section, the security of the proposed protocol is proved under $DBDH$ assumption. In addition, the protocol is analyzed to provide other security attributes a group key agreement protocol should achieve.

Theorem 4.1. *The proposed protocol is secure against active adversary. Concretely,*

$$Adv_{A,P}(k) \leq 2n^2 Succ_{\Sigma}^{cma} + q_s(2q_h q_s^2 Succ_{G_1,G_2}^{BDH} + \frac{1}{2^{nl-1}})$$

.

where q_s is the number of $Send$ queries, q_h is the number of queries to hash oracle H and n is the number of group members.

Proof: In order to simulate the attack of the adversary, we define a sequence of games: $\{\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2, ..., \mathbf{G}_5\}$. In each game, the adversary executes $Test$ query and gets a challenge session key sk_b. $\mathbf{Succ}_i$ denotes the event that $\mathcal{A}'s$ guessing bit b' is equal to b in game $\mathbf{G}_i$. Each $\mathbf{G}_i$ is simulated as follows:

$Game$ $\mathbf{G}_0$: This game is equal to the real protocol in which all users are assigned a pair of valid sign/verification keys and generate messages honestly. It follows that

$$Pr[\mathbf{Succ}_0] = \frac{Adv_{A,P}(k) + 1}{2}. \tag{1}$$

$Game$ $\mathbf{G}_1$: In this game, we consider an event $Forge$ in which the adversary asks for a $Send(m, \delta_i)$ query with $V(pk_{u_i}, m, \delta_i)$=1. The message m was not previously used and no $Corrupt(u_i)$ query has ever been executed. Using the adversary $\mathcal{A}$, we can construct an algorithm F that forges a signature as follows: Given a public key pk, F sets $pk_u = pk$ with u being a random user of group $\mathbb{U}_v$. The public key and private key of other users are generated honestly by F. F answers all oracle queries of $\mathcal{A}$ by executing the protocol itself. It obtains the necessary signatures with respect to pk_u from its signing oracle. Thus the simulation of F for the adversary is perfect. If the adversary ever outputs a new valid message /signature pair with respect to pk_u, F outputs this pair as a forgery. The probability that F successfully forges a signature is $\frac{Pr[Forge]}{n}$. Thus we have

$$Pr[Forge] \leq nAdv_{\Sigma,A}^{cma}(t)$$

.

If the event $Forge$ occurs, the game halts and the adversary outputs a random bit b'. The games $\mathbf{G}_0$ and $\mathbf{G}_1$ are identical as long as event $Forge$ does not occur. If we can correctly guess the impersonated user, we can get:

$$|Pr[\mathbf{Succ}_1] - Pr[\mathbf{Succ}_0]| \leq nPr[Forge] \leq n^2 Succ_{\Sigma,A}^{cma}(t) \tag{2}$$

$Game$ $\mathbf{G}_2$: This game is the same as the previous one except for the following rule: F does not correctly guess the test session. If this event happens, a random bit b' is output and the game halts. Let E denote the event that F does not correctly guess the test session. We have

$$Pr[\mathbf{Succ}_2] = Pr[\mathbf{Succ}_2 \mid E]Pr[E] + Pr[\mathbf{Succ}_2 \mid \neg E]Pr[\neg E]$$

$$= Pr[\mathbf{Succ}_1]\frac{1}{q_s} + \frac{1}{2}(1 - \frac{1}{q_s}) \tag{3}$$

$Game$ $\mathbf{G}_3$: This game differs from the previous one in how **Send** queries in test session are answered.

Given an instance of BDH problem $(P, aP, bP, cP, \hat{e}(P, P)^{abc})$, F sets $y_1 = g^a, V_1 = g^b, h = g^c$. Then F chooses random $t_1, ..., t_{n-1}$ from $\mathbb{Z}_q^*$ and sets $y_2 = g^{t_1}, V_2 = V_1^{t_1}, ..., y_n = g^{t_{n-1}}, V_n = V_1^{t_{n-1}}$. Other values are obtained according to the description of the protocol. It follows that $s_i = \hat{e}(g, g)^{abc} e(h, V_1 + ... + V_{n-1})$. Since all ephemeral secret values are chosen randomly, this game is identical with $\mathbf{G}_2$. Thus we have

$$Pr[\mathbf{Succ}_3] = Pr[\mathbf{Succ}_2]. \tag{4}$$

$Game$ $\mathbf{G}_4$: In this game, a tuple (P, aP, bP, cP) is given and there is no information about $\hat{e}(P, P)^{abc}$. If any hash value involving s_i is asked, a random value $r \in \{0, 1\}^l$ is returned as the response. Let $Hash$ be an event in which the hash value $H(s_i)$ is incorrect by using hash oracle H. This is possible if $\mathcal{A}$ correctly guesses $\hat{e}(g, g)^{abc}$, sends it to the hash oracle and receives a value different from r. When event $Hash$ occurs, F aborts the game and output a random bit b'. Thus we have:

$$| Pr[\mathbf{Succ}_4] - Pr[\mathbf{Succ}_3] | \leq Pr[Hash]$$

Since there are at most q_s $Send$ queries and s_i is correctly guessed, we have

$$Pr[Hash] \leq q_H q_s^2 Succ_{G_1,G_2,\hat{e}}^{BDH}$$

.

Finally, we have

$$| Pr[\mathbf{Succ}_4] - Pr[\mathbf{Succ}_3] | \leq q_H q_s^2 Succ_{G_1,G_2,\hat{e}}^{BDH} \tag{5}$$

$Game$ $\mathbf{G}_5$: This game is identical to the previous one except that the adversary finds a collusion for

$H(k_1\|...\|k_n\|ID_v)$. The probability that $\mathcal{A}$ finds a collusion for $H(k_1\|...\|k_n\|ID_v)$ is at most $\frac{1}{2^{nl}}$. It follows that

$$| Pr[\mathbf{Succ}_5] - Pr[\mathbf{Succ}_4] | \leq \frac{1}{2^{nl}} \qquad (6)$$

If the adversary does not finds a collusion for $H(k_1\|...\|k_n\|ID_v)$, it has no advantage in guessing b in this game. Thus we have $Pr[\mathbf{Succ}_5] = \frac{1}{2}$.

By combining equations (1) to (6), we obtain the desired results. □

In the following, we will consider some other security attributes that are often used to judge key agreement protocols.

Forward Security:

A protocol is said to provide forward security if compromise of any user's private key does not allow the adversary to discover any past session keys.

In the proposed protocol, the long-term private key is not used for hiding session key but for authentication. Thus leakage of any user's long-term private key does not reveal anything about previous session keys.

No Key Control:

A key agreement protocol is said to resist key control attack if no one can predetermine the final session key.

In the proposed protocol, the final session key is of the form $H(k_1 \parallel ... \parallel k_n \parallel ID_v)$. Key control can be guaranteed by the check process $H(k_n \parallel ID_v) = U_n$ and one way of hash function H. No one can force the full session key to be a predicted value, because every one has an input into the key and no one can control it. However, the user can set some bits of the agreed session key by carefully selecting his contribution k_i until he achieves the desired result. Fortunately, it is not possible for a user to set a large number of bits in a reasonable time frame. It is advisable for all users to run the protocol in a short time.

V. Performance and Comparison

We now compare our protocol with another dynamic group key agreement protocol [25] which is suitable for ad hoc networks. We will use the flowing notations:

$Round$: The total number of rounds.

Mul : The total number of modular multiplications .

$Msize$: The maximum number of messages sent by per user.

P/E : The total number of pairing computations or exponentiations.

Table 2 Setup algorithm - A set of users $U_{[1,...,n]}$ will establish a session key.

$Protocol$	$Round$	$Msize$	Mul	P/E
[25]	3	$O(n)$	$O(n)$	$O(n^2)$
Ours	2	$O(n)$	$O(n)$	$O(n)$

Table 2 Join algorithm-A set of users $U_{[n+1,...,n+m]}$join the set of users $U_{[1,...,n]}$ resulting a set of size $n+m$

$Protocol$	$Round$	$Msize$	Mul	P/E
[25]	3	$O(n+m)$	$O(n+m)$	$O((n+m)^2)$
Ours	2	$O(m)$	$O(n+m)$	$O(n+m)$

Table 3 Leave algorithm-A set of users $U_{[l_1,...l_m]}$ leave the set of users $U_{[1,...,n]}$ resulting a set of size $n-m$

$Protocol$	$Round$	$Msize$	Mul	P/E
[25]	3	$O(n-m)$	$O(n-m)$	$O(n-m)^2$
Ours	2	$O(n-m)$	$O(n-m)$	$O((n-m))$

As shown above, the proposed protocol is more efficient than that in [25]. Moreover, the number of transmitted messages in $Join$ algorithm does not increase with the number of all members, which greatly improves the entire efficiency of the protocol.

VI. Conclusion

In this paper, a dynamic authenticated group key agreement protocol is presented for ad hoc networks. Its security is proved in random oracle model under DBDH assumption. The leaving members can get no information about subsequent session keys and joining members can get no information about previous session keys. It also provides forwards security and resists key control attack. Furthermore, in the proposed protocol, the number of messages transmitted in $Join$ algorithm only increases with the number of increasing members, which makes the protocol more practical.

References

[1] W. Diffie and M.Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, vol. 12, no. 6, pp. 644–654, Nov. 1976.

[2] A.Joux, "A One Round Protocol for Tripartite Diffie-Hellman," in *Proceedings of 8-th Algorithmic Number Theory Symposium*, July 2000, pp. 385–394.

[3] Z.W.Tan, "An Authentication and Key Agreement Scheme with Key Confirmation and Privacy-preservation for Multi-server Environments," *Journal of Computers*, vol. 6, no. 11, pp. 2295–2301, Nov. 2011.

[4] D. T. I. Ingemarsson and C. K. Wong, "A Conference Key Distribution System," *IEEE Transactions on Information Theory*, vol. 28, no. 5, pp. 714–720, Sept. 1982.

[5] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater, "Provably authenticated group Diffie-Hellman key exchange," in *Proceedings of CCS 2001*, Oct. 2001, pp. 255–264.

[6] R. C. M. Bellare and H. Krawczyk, " A Modular Approach to The Design and Analysis of Authentication and Key-Exchange Protocols," in *Proceedings of the 30th Annual Symposium on the Theory of Computing*, May 1998, pp. 419 – 428.

[7] R. D. R. Barua and P. Sarker, " Extending Joux's Protocol to Multi Party Key Agreement," in *Proceedings of Indocrypt 2003*, May 2003, pp. 205–217.

[8] K. C. Reddy and D. Nalla, "Identity based Authenticated Group Key Agreement Protocol," in *Proceedings of Indocrypt 2002*, Dec. 2002, pp. 215–233.

[9] R. Barua, R. Dutta, and P. Sarker, "Provably Secure Authenticated Tree Based Group Key Agreement," in *Proceedings of ICICS 2004*, Oct. 2004, pp. 93–104.

[10] J. Katz and M. Yung, "Scalable Protocols for Authenticated Group Key Exchange," in *Proceedings of CRYPTO-2003*, Aug. 2003, pp. 110–125.

[11] M.Burmester and Y.Desmedt, "A Secure and Efficient Conference Key Distribution System," in *Proceedings of Eurocrypt'94*, May 1995, pp. 275–286.

[12] K. Y. Choi, J. Y. Hwang, and D. H. Lee, “Effcient ID-based Group Key Agreement with Bilinear Maps,” in *Proceedings of PKC 2004*, Feb. 2004, pp. 130 – 144.
[13] J. Bohli, B. Glas, and R. Steinwandt, “Towards Provably Secure Group Key Agreement Building on Group Theory,” in *Proceedings of VIETCRYPT 2006*, Sept. 2006, pp. 322–336.
[14] J.K.Teng and C.K.Wu, “A Provable Authenticated Certificateless Group Key Agreement With Constant Rounds,” *Journal of Communications and Networks*, vol. 14, no. 1, pp. 104–110, Feb. 2012.
[15] J.K.Teng, C.K.Wu, and C. Tang, “An ID-based authenticated dynamic group key agreement with optimal round ,” *Science China Information Sciences*, vol. 55, no. 11, pp. 2542–2554, Nov. 2012.
[16] M. C.Gorantla, C.Boyd, J.M.G.Nieto, and M. Manulis, “Generic One Round Group Key Exchange in the Standard Model,” in *Proceedings of ICISC 2009*, Dec. 2009, pp. 1–15.
[17] C. Ma, J. Wang, P. Wu, and H. Zhang, “An Authentication and Key Agreement Scheme with Key Confirmation and Privacy-preservation for Multi-server Environments,” *Journal of Computers*, vol. 7, no. 8, pp. 1847–1852, Aug. 2012.
[18] K. Ammayappan, A. Negi, V. N. Sastry, and A. K. Das, “An ECC-Based Two-Party Authenticated Key Agreement Protocol for Mobile Ad Hoc Networks,” *Journal of Computers*, vol. 6, no. 11, pp. 2408–2416, Nov. 2011.
[19] E. Bresson, O. Chevassut, and D. Pointcheval, “Provably Authenticated Group Diffie-Hellman Key Exchange-The Dynamic Case,” in *Proceedings of Asiacrypt 2001*, Dec. 2001, pp. 290–309.
[20] E.Bresson, O.Chevassut, and D.Pointcheval, “Dynamic Group Diffie-Hellman Key Exchange Under Standard Assumptions,” in *Proceedings of Cryptology-Eurocrypt 2002*, May 2002, pp. 321–336.
[21] Y. Kim, A. Perrig, and G. Tsudik, “Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups,” in *Proceedings of CCS 2000*, Oct. 2000, pp. 235–244.
[22] R. Dutta and R. Barua, “Dynamic Group Key Agreement in Tree-based Setting,” in *Proceedings of ACISP 2005*, July 2005, pp. 101–112.
[23] H. J. Kim, S. M. Lee, and D. H. Lee, “Constant-Round Authenticated Group Key Exchange for Dynamic Groups,” in *Proceedings of Asiacrypt 2004*, Dec. 2004, pp. 245–259.
[24] R.Dutta and R.Barua, “Provably Secure Constant Round Contributory Group Key Agreement in Dynamic Setting,” *IEEE Transactions On Information Theory*, vol. 54, no. 5, pp. 2007–2025, May 2008.
[25] D.Augot, V. R.Bhaskar, and D.Sacchetti, “A Three Round Authenticated Group Key Agreement Protocol for Ad Hoc Networks,” *Pervasive and Mobile Computing*, vol. 3, no. 1, pp. 36–52, Jan. 2007.
[26] J.K.Teng and C.K.Wu, “Efficient Group Key Agreement for Wireless Mobile Networks,” in *Proceedings of IET-WSN 2010*, Nov. 2010, pp. 323–330.
[27] D. Boneh and M. Franklin, “Identity-based Encryption from The Weil Pairing,” in *Proceedings of Cryptology - CRYPTO '01*, Aug. 2001, pp. 213–229.
[28] P. S. L. M. Barreto, H. Y. Kim, and M. Scott, “Efficient Algorithms for Pairing Based Cryptosystems,” in *Proceedings of Cryptology - CRYPTO '02*, Aug. 2002, pp. 354–368.

Zongyu Song received his MS degree in science from School of Mathematical Science, Guangzhou University, China, in 2004.

He is currently a doctor candidate of Wuhan University Technology, China. His research interests include cryptographic protocols and network security.

Pengfei Cai is a of Zhengzhou Normal University, Zhengzhou China. His current research interests include cryptographic protocols and network security.

Jie Yang received his PhD degree in science from the Shanghai Jiaotong University in 1999. He is currently a Professor of Computer Science Wuhan University Technology, China. His current research interests include network security and provable security.