2014

An Improved ID-Based Group Key Agreement Protocol

Kangwen Hu
School of Software, Beijing Institute of Technology, Beijing 100081, China.

Jingfeng Xue
School of Software, Beijing Institute of Technology, Beijing 100081, China.

Changzhen Hu
School of Software, Beijing Institute of Technology, Beijing 100081, China.

Rui Ma
School of Software, Beijing Institute of Technology, Beijing 100081, China.

Zhiqiang Li
School of Software, Beijing Institute of Technology, Beijing 100081, China.

Follow this and additional works at: https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology

Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Recommended Citation

This Research Article is brought to you for free and open access by Tsinghua University Press: Journals Publishing. It has been accepted for inclusion in Tsinghua Science and Technology by an authorized editor of Tsinghua University Press: Journals Publishing.
An Improved ID-Based Group Key Agreement Protocol

Kangwen Hu, Jingfeng Xue, Changzhen Hu, Rui Ma, and Zhiqiang Li*

Abstract: ID-based constant-round group key agreement protocols are efficient in both computation and communication, but previous protocols did not provide valid message authentication. An improvement based on attack analysis is proposed in this paper. The improved method takes full advantage of the data transmitted at various stages of the protocol. By guaranteeing the freshness of authentication messages, the authenticity of the generator of authentication messages, and the completeness of the authenticator, the improved protocol can resist various passive and active attacks. The forward secrecy of the improved protocol is proved under a Katz-Yung (KY) model. Compared with existing methods, the improved protocol is more effective and applicable.

Key words: group key agreement protocol; ID; forward secrecy; nonsuper-singular elliptic curve

1 Introduction

The Identity-Based Cryptosystem (IBC) proposed by Shamir in 1984 is simpler than the PKI/CA that is currently widely used in key management[1]. In 2000, Joux proposed a tripartite key agreement with one round of communication using both Weil and Tate pairing[2]. Now, group key agreement protocols based on bilinear pairings of identity have elicited a lot of research. In 2002, Reddy first proposed an IDentity-based Authenti-cated Group Key Agreement (ID-AGKA) protocol with HOFT[3]. This protocol used Weil pairing, and provided implicit key authentication attribute; however, it just analyzed security attributes did not give rigorous proof. In 2003, Du et al. proposed a constant-round group key agreement protocol based on Burmester-Desmedt (BD) structure[4, 5]. In 2004, Choi et al.[6] proposed a similar ID-AGKA and proved its security; it is efficient on both computation and communication but, because of a lack of entity authentication, it cannot resist internal impersonation or external attacks[7-9]. Many researchers have put forward a series of ways to improve Choi’s ID-AGKA[8-13]. In this paper, a series of ID-AGKAs, represented by Choi’s protocol, are analyzed and a new, improved scheme is proposed. This scheme can resist known internal impersonation attacks as well as external attacks.

2 Choi’s ID-AGKA Protocol

In this section, we will briefly introduce the Choi’s protocol; its detailed description is in Ref. [6].

Through the paper, we assume that $G_1$ is a cyclic additive group of big prime order $q$ and $G_2$ is a cyclic multiplicative group of the same order $q$. $P$ is $G_1$’s generator. The discrete logarithm problem is intractable in both $G_1$ and $G_2$. $e: G_1 \times G_1 \rightarrow G_2$ is a valid bilinear map and satisfies the decisional bilinear Diffie-Hellman problem (DBDH) assumption[14]. $H: \{0, 1\}^* \rightarrow Z_q$ and $H_1: \{0, 1\}^* \rightarrow G_1$ are two hash functions. In the security proof, $H$ and $H_1$ are treated as random oracles.

Setup: The private Key Generation Center (KGC) randomly chooses a number $s \in Z_q^*$ as its master secret key, chooses $G_1$’s generator $P$, computes $P_{pub} = sP$, and publishes system parameters $\text{params} = \{e, G_1, G_2, q, P, P_{pub}, H, H_1\}$.

Extract: The user $U_{ID}$ with the identity of the ID sends the ID to the KGC. The KGC computes the public key $Q_{ID} = H_1(\text{ID})$ and the private key $S_{ID} =$
There are mainly four types of attacks on the Choi’s protocol:

1. **Impersonation attacks by malicious neighbors.** Zhang and Chen pointed out that $U_{i-1}$ and $U_{i-2}$, two malicious neighbors of $U_i$, may collude to replay $(P_i, T_i)$, which is $U_i$’s authentication message in group $G_A$. The attackers can impersonate $U_i$ in a new group, $G_B$, without being noticed.

2. **Impersonation attacks by colluding verifiers.** Shim pointed out that if $U_{i-2}$, $U_{i-1}$, and $U_{i+1}$ ($U_i$’s three malicious neighbors) collude, they can impersonate $U_i$ without replaying $(P_i, T_i)$ in a new group with randomly chosen $a_i$ and $D_i$.

3. **External attacks that lead to group members computing different session keys.** Li and He pointed out that adversaries may divide the group into two subgroups and transmit different data to each. Thus, these two subgroups will get different session keys which means the protocol is broken.

4. **Passive attacks that compute session keys simply by monitoring transcripts.** Li pointed out that the protocol proposed by Liu and Xu cannot resist the three attacks mentioned above and, according to the nature of bilinear pairings, as long as communication in the group is monitored, the attackers can compute session keys.

In fact, we find that as long as the adversary can pass verification in Round 1, it can break the protocol successfully. So these series of protocols will not be able to resist man-in-the-middle attacks. If adversary $A$ can control all transmitted/received data of $U_i$, and it can save data transmitted by $U_i$ in a normal agreement at sometime denoted by $(P'_1, T'_1, D'_1)$. When $U_i$ is involved in a new agreement, adversary $A$ can launch attacks as follows:

- $U_i$ will transmit $(P'_1, T'_1)$ according to the protocol in Round 1, $A$ intercepts $(P'_1, T'_1)$ of $U_i$, broadcasts $(P'_1, T'_1)$ to the other members of the group and forwards $(P'_1, T'_1)$ transmitted by the other nodes to $U_i$. In Round 2, $A$ replays $D_i$ to the other members of the group and forwards $D_i$ from the other nodes to $U_i$. After the agreement, the session key shared by $A$ and $U_i$ is $K_1 = e(P, P)^{a_1a_2a_3+\ldots+a_{n-1}a_1a_2+a_na_1a_2}$; the session key shared by $A$ and the other users is $K'_1 = e(P, P)^{a_1a_2a_3+\ldots+a_{n-1}a_1a_2+a_1a_2a_3}$. Thus, $A$ can decrypt data that encrypted by any session keys of the two subgroups.

In summary, there are three reasons why impersonation attacks work:

1. The protocol does not authenticate the $D_i$ in Round 2.
2. The authenticators of $(P_i, T_i)$ are $U_i$’s neighbors only.
3. The authentication data associates with protocol’s current execution status little.

The reason for the success of passive attacks is that the temporary secret key is not used correctly in the session key formula.

Therefore, protocol improvement should be focused on the freshness of the authentication messages, the authenticity of the generator of the authentication messages, and the completeness of the authenticators.

(1) $P_i$ broadcasted by each user in Round 1 can
identify current run of protocol, we can integrate all $P_i$ values into the signature computation to ensure its freshness.

(2) The signature must be generated by a long-term private key to ensure the authenticity of the entities being authenticated.

(3) All users in the group must contribute to authentication information to ensure the completeness of the authenticators.

In addition, we should minimize unnecessary computations and messages for efficiency.

4 Improvement of the Protocol and Analysis

4.1 Improvement of the protocol

Setup: System parameters and the protocol initialization are the same as the original protocol.

Round 1: Each user $U_i$ randomly chooses $a_i \in Z_q^*$, computes $P_i = a_iP$, broadcasts $P_i$ to others and keeps $a_i$ secret.

Round 2: Upon the receipt of all data broadcasted by other members of the group, user $U_i$ computes $D_i = e(a_i(P_{i+1} - P_{i-1}), P_{pub}), T_i = a_iP_{pub} + h_{iS_i}$; where $h_i = H(D_i \| P_{ID} \| S_{ID}), P_{ID} = P_1 \cdots \| P_n,$ and $S_{ID} = ID_1 \cdots \| ID_n$. Then, it broadcasts $(D_i, T_i)$ to other members of the group.

Key computation: The user $U_i$ checks whether $e(T_j, P) = e(P_j + h_jQ_j, P_{pub}), (1 \leq j \leq n$ and $j \neq i)$ is true. If not, $U_i$ aborts and broadcasts “failure”. Otherwise, $U_i$ computes session key $K_i$: 

$$K_i = e(a_iP_{i-1}, P_{pub})^aD_i^{n-1}D_i^{n-2} \cdots D_i = e(P, P)^{(a_1a_2 + a_2a_3 + \cdots + a_na_i)x}.$$ 

The authentication mechanism $\Gamma$ of the protocol is similar to Hess’s signature scheme. The definition is as follows.

Signature generation: Given a secret key, compute $S_i = sH_i(ID_i)$, then $T = aP_{pub} + hS_i$, where $a \in \mathbb{F}_q$, $h = H(D_i \| P_{ID} \| S_{ID})$, and $(aP, T) \leftarrow \Gamma_{gen}(S_{ID})$.

Signature verification: Given public key $Q_{ID}$ and signature $(D_i, T_i)$, check whether $e(T, P) = e(aP + hQ_{ID}, P_{pub})$ is true, where $h = H(D_i \| P_{ID} \| S_{ID})$; True or False $\leftarrow \Gamma_{ver}(Q_{ID}, (aP, T))$.

The batch validation reduces the number of calculations of bilinear pairings and improves verification efficiency[13]. Our improvement is similar to those in Refs. [4, 6, 16]. According to the nature and symmetry of bilinear pairings, $\prod_{i=1}^{n} D_i = 1$ is true. After each user receives all $D_i$, they should check $D_i$ by this formula.

4.2 Proof of security

This protocol transmits messages through broadcasting. All participants (including the adversary who controls the network) will receive the same message. The KY model satisfies this feature[17]. The security definitions can be found in Refs. [6, 17]. To prove this protocol, we measure indistinguishability by the hybrid argument method[18]. We name this improved protocol IB-AGKA and will prove that it still provides forward secrecy.

Theorem 1 After the active rival issues $q_{ex}$ Execute inquiries and $q_s$ Send inquiries within time $t$, we define $Adv^{IB-AGKA}_{IB-AGKA}(t, q_{ex}, q_s)$ as the maximum advantage of the attacker. We define $Forger_\Gamma$ as a Probabilistic Polynomial Time (PPT) forger of authentication scheme $\Gamma$ under the adaptively chosen ID attack, and $Forge^{IB-AGKA}_\Gamma$ as a PPT forger of $\Gamma$ under given ID attack. We take hash functions $H$ and $H_1$ as random oracles.

$$Adv^{IB-AGKA}_{IB-AGKA}(t, q_{ex}, q_s) \leq 2n(q_{ex} + q_s)Adv^{DBDH}_{G_1, G_2, e}(t) + Adv^{Forge}_\Gamma(t).$$

Here, $Adv^{Forge}_\Gamma(t)$ is the maximum advantage of any forger $Forger_\Gamma$ running in time $t$.

Proof Let $A$ be an active attacker, who can get advantage in attacking the protocol in two ways:

(1) Forging an authentication message or impersonating a user.

(2) Breaking the protocol without modifying any message.

Assuming that Adversary $A$ breaks IB-AGKA by adaptive impersonation attack, we can construct a forger $Forger_\Gamma$ of $\Gamma$ of an authentication scheme $\Gamma$ by $A$. This forger can produce a valid ternary $(U_i, D_i, T_i)$ of authentication scheme $\Gamma$ in the following ways.

Forger$^\_\Gamma$ honestly generates the public/private key pair of all other users except $U_i$. $C$ simulates the oracle inquiries of adversary $A$ in the natural way; this results in a perfect simulation unless $A$ issues Corrupt($U_i$), in which case, $C$ aborts. If $A$ produces a new valid $(U_i, D_i, T_i)$, we denote this event by $Forge$ and make $A$ pass the result to $C$. Thus, we believe that $C$ is a successful forger of authentication scheme $\Gamma$. So
the probability of $C$’s success meets $\Pr_A[\text{Forge}] \leq Adv_{C,t}(r) \approx Adv_{Y}(t)$. 

Next, we assume that $A$ can also break IB-AGKA without altering transcripts; thus, we can use IB-AGKA to solve the MDBDH problem. According to Ref. [6], MDBDH and DBDH are computationally equivalent, namely, $Adv_{G_1,G_2,e}(t) = Adv_{DBDH}^{MDBDH}(t)$. We first consider the case that $A$ issues only a single $\text{Execute}$ query $\text{Execute}(\text{ID}_1,\cdots,\text{ID}_n)$ and then extend this to multiple $\text{Execute}$ queries. Let $n$ be the number of users selected by $A$; the distribution of transcripts $T$ and the group session key $K$ are given by:

$$
\text{Params} = \left\{ (G_1, G_2, e) \leftarrow \text{IGBDH}(1^k); \ P \leftarrow G_1; s \leftarrow Z_q^*; P_{\text{pub}} = sP; Q_1, \cdots, Q_n \leftarrow G_1; S_1 = sQ_1, \cdots, s_n = sQ_n; \ (G_1, G_2, e, P, P_{\text{pub}}) \right\}
$$

$$
\text{Real}^{\text{def}} = \left\{ a_1, \cdots, a_n, h_1, \cdots, h_n \leftarrow Z_q^*; P_1 = a_1 P, \cdots, P_n = a_n P; T_1 = a_1 P_{\text{pub}} + h_1 S_1, \cdots, T_n = a_n P_{\text{pub}} + h_n S_n; D_1 = e(a_1 P_2, P_{\text{pub}}); D_2 = e(a_2 P_3, P_{\text{pub}}); \cdots, D_n = e(a_n P_1, P_{\text{pub}}); T = \langle P_1, \cdots, P_n, T_1, \cdots, T_n, D_1, \cdots, D_n \rangle; K = e(a_1 P_1, P_{\text{pub}})^n D_1^{-1} \cdots D_n^{-1}(T, K) \right\}
$$

where $\text{IGBDH}$ is a PPT algorithm that takes a security parameter $1^k$, runs in polynomial time, and outputs two groups $G_1$ and $G_2$ of the same order $q$ and an reasonable bilinear map $e: G_1 \times G_2 \rightarrow G_2$.

Now we construct a serial of mixed distributions, $\text{Fake}_t (t = 1, \cdots, n)$ which is defined as follows:

$$
\text{Fake}_t^{\text{def}} = \left\{ r_{s,n,1}, \cdots, a_1, \cdots, h_n \leftarrow Z_q^*; P_1 = a_1 P, \cdots, P_n = a_n P; T_1 = a_1 P_{\text{pub}} + h_1 S_1, \cdots, T_n = a_n P_{\text{pub}} + h_n S_n; D_1 = e(a_1 P_2, P_{\text{pub}}); D_2 = e(a_2 P_3, P_{\text{pub}}); \cdots, D_n = e(r_{s,n,1} P_1, P_{\text{pub}}); T = \langle P_1, \cdots, P_n, T_1, \cdots, T_n, D_1, \cdots, D_n \rangle; K = e(r_{s,n,1} P_1, P_{\text{pub}})^n D_1^{-1} \cdots D_n^{-1}(T, K) \right\}
$$

According to this construction method, we can obtain the distribution:

$$
\text{Fake}_n^{\text{def}} = \left\{ r_{s,n,1}, \cdots, r_{s,n-1,n}, a_1, \cdots, a_n, h_1, \cdots, h_n \leftarrow Z_q^*; P_1 = a_1 P, \cdots, P_n = a_n P; T_1 = a_1 P_{\text{pub}} + h_1 S_1, \cdots, T_n = a_n P_{\text{pub}} + h_n S_n; D_1 = e(r_{s,1,2} P, P_{\text{pub}}); D_2 = e(r_{s,2,3} P, P_{\text{pub}}); \cdots, D_n = e(r_{s,n,1} P, P_{\text{pub}}); T = \langle P_1, \cdots, P_n, T_1, \cdots, T_n, D_1, \cdots, D_n \rangle; K = e(r_{s,n,1} P, P_{\text{pub}})^n D_1^{-1} \cdots D_n^{-1}(T, K) \right\}
$$

Adversary $A$ can obtain all long-term secret keys $S_i$ and hash values $h_i (i = 1, \cdots, n)$ through multiple $\text{Corrupt}$ and $H$ (in phase; compute $a_i P_{\text{pub}} = T_i - h_i S_i = sa_i P$ (since $P_{\text{pub}} = sP$ is a global parameter, and $P_i = a_i P$ can be obtained from transcripts). Let $\varepsilon(t) = Adv_{G_1,G_2,e}(t)$, according to the MDBDH assumption; any distinguishing algorithms $A$ running in time $t$ can result:

$$
| \Pr[(T, K) \leftarrow \text{Real}: A(T, K) = 1] - \Pr[(T, K) \leftarrow \text{Fake}_1: A(T, K) = 1]| \approx \varepsilon(t).
$$

The reason is that adversary $A$ has to distinguish $e(P, P)^{r_{s,1,2}}$ from $e(P, P)^{r_{s,n,1}}$, which satisfies MDBDH.

Similarity:

$$
| \Pr[(T, K) \leftarrow \text{Fake}_1: A(T, K) = 1] - \Pr[(T, K) \leftarrow \text{Fake}_2: A(T, K) = 1]| \approx \varepsilon(t).
$$

Let $e(P, P) = g \in G_2$. In experiment $\text{Fake}_n$, the value $r_{s,1,2}, \cdots, r_{s,n,1}$ are constrained by $T$ according to the following $n$ equations:

$$
\log_g D_1 = r_{s,1,2} - r_{s,n,1},
\log_g D_2 = r_{s,2,3} - r_{s,1,2},
\vdots
\log_g D_n = r_{s,n,1} - r_{s,n-1,n}.
$$

The coefficient matrix of the equation set is

$$
\begin{pmatrix}
1 & 0 & \cdots & \cdots & -1 \\
-1 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & \cdots & \cdots & 1
\end{pmatrix}
$$

Its rank is $n - 1$; $n - 1$ vectors are linearly independent. In distribution $\text{Fake}_n$, $K_{\text{Fake}_n} = \ldots
e(P, P)'s.n.1 + r.s,2,3 + + + r.s,n,1. If both sides of
the equation are logarithmic, we get log g K Fake n = r.s,1,2 +
r.s,2,3 + + + r.s,n,1. Obviously, it is independent of
T. This indicates that for any adversary A, the equation
is true:
\[
\Pr[(T, K) ← Fake_n : A(T, K) = 1] = \\
Pr[T ← Fake_n, K ← Random: A(T, K) = 1].
\]
This means that adversary A cannot distinguish session
key K Fake n from a random number of the same length
by distribution Fake n.

Similarly, under MDBDH assumption, any algorithm
A running in time t, we can get:
\[
\begin{align*}
|Pr[T ← Fake_n; K ← Fake_n : A(T, K) = 1]| &= \epsilon (t), \\
Pr[T ← Fake_{n-1}; K ← Random : A(T, K) = 1]| &\leq \epsilon (t), \\
& \vdots \\
Pr[T ← Real; K ← Random : A(T, K) = 1]| &\leq \epsilon (t).
\end{align*}
\]
According to the above equations, we can get:
\[
\begin{align*}
|Pr[T ← Real; K ← Real : A(T, K) = 1]| &\leq 2n\epsilon (t).
\end{align*}
\]
This means, when the adversary A intercepts all
transcripts of the subgroup he chooses, the advantage of
distinguishing real session key K from a random value
of the same length is 2n\epsilon (t).

Because \(\epsilon (t) = \text{Adv}_{\text{MDDBDH}}^G (t) = \text{Adv}_{\text{DBDH}}^G (t)\),
\(\text{Pr}_A[\sim \text{Forge}] \leq 2n\text{Adv}_{G_1, G_2, e}^G (t)\) is established.

In summary:
\[
\text{Adv}_{IB-AGKA}^{ID-AGKA} (t, 1) \leq 2n\text{Adv}_{G_1, G_2, e}^G (t) + \epsilon_{\text{Forge}} (t).
\]
After issuing qS \text{ Send} inquires and qE \text{ Execute}
inquires, the result is:
\[
\text{Adv}_{IB-AGKA}^{ID-AGKA} (t, qS, qE) \leq 2n(qS + qE)\text{Adv}_{G_1, G_2, e}^G (t) + \epsilon_{\text{Forge}} (t).
\]
Detailed calculations regarding the forger’s advantage are in Refs. [6, 19].

### 4.3 Analysis of improvement

This improvement uses some of the ideas in Refs. [11,
13, 20]. In addition to its forward security, it can resist
all kinds of known attacks:

1. **Active attacks**\[7-10\]: \(U_i\)’s malicious neighbors
\(U_{i-1}\) and \(U_{i+1}\) save the message \((P_i', D_i', T_i')\),
which was transmitted in the old group \(G_A\) by
\(U_i\). They send \(P_i'\) in Round 1 of the new group
\(G_B\), thus \(a_i = a_i'\). According to the protocol,
\(U_{i-1}\) and \(U_{i+1}\) in \(G_B\) can get \(P_i\) broadcasted
by all users in the first round. If the collusion
attackers want to pass verification in Round 2, they
must construct the correct signature of
\(D_i\), which is \(T_i = a_i P_{pub} + h_i S_i\). Although the
attacker can compute \(h_i = H(D_i || P_{ID} || S_i)\), it
still needs to compute \(a_i P_{pub}\) and obtain \(U_i\)’s
long-term private key \(S_i\). Under the Elliptic
Curve Discrete Logarithm Problem (ECDLP)
assumption and secure transmission of the private
key, neither can be obtained, i.e., the adversary
cannot calculate \(T_i\). The malicious neighbors
cannot generate valid \(U_i\) signatures in new group,
and the protocol’s honest participants are able to
detect the attacks.

2. **Passive attacks**\[10\]: The adversary can obtain
system parameters \(P_i, T_i,\) and \(D_i\). According to
\(K = e(a_i P_{pub}^i, P_{pub})^n D_i^{n-1} D_{i+1}^{n-2} \cdots D_{i-1}^{n-2},\)
the adversary knows that \(P_i = a_i P_i, P_{pub} = sP_i,
\) and \(P_{i-1} = a_{i-1} P_i\). Under the Computational
Bilinear Diffie-Hellman Problem (CBDHP)
problem assumption, the adversary cannot
compute \(e(a_i P_{i-1}^i, P_{pub}^i) = e(P, P)^{a_{i-1} a_i}\), so
they cannot get the session key.

In addition to the Internet, LAN, and other
common networks, these improvements also apply to
environments where the node’s calculation capability
is weak, the node’s buffer is small, and the channel
bandwidth is low—such as Wireless Sensor Networks
(WSNs). In this type of network, protocol data can be
transmitted in batches. In Round 1, \(U_i\) first unicasts
\(P_i\) to \(U_{i-1}\) and \(U_{i+1}\) and then broadcasts \(P_i\) to the other
nodes. After \(U_i\) receives \(P_{i-1}\) and \(P_{i+1}\), it computes
\(D_i\) and broadcasts it. Thus, although the \(P_i\) values of
the other nodes required in Round 2 for \(T_i\) calculation
are received relatively late, we can first calculate session
key \(K\) and verify \(T_i\) later. The improvement can avoid
channel congestion caused by all node’s simultaneous
broadcasting. Nodes need not wait for all data to arrive
and then calculate. The parallelism of system is best.

Regardless of whether Tate or Weil pairing
is used, when parameters \(P\) and \(Q\) are linearly
dependent, their safety is not guaranteed. We denote,
\(Q = k P (k \in R, Z_m, P \in E[m])\), thus, we have
\(e(P, Q) = e(P, k P) = e(P, P)^k = 1\) with Weil
pairing, according to its identity element and bilinear
nature\[14\]. Obviously, it must not occur in practice. In
Ref. [6], Choi made an admisible bilinear map that
satisfies \(e(P, P) \neq 1\), that may need extra work on
existing pairing. Although the hypersingular curve can
map two related points to different groups\[14, 21-23\] by
distortion mapping, the nonhypersingular curve does not have this property. In the original protocol\(^6\) and its improvement\(^{9,10,13}\), the two bilinear parameters both belong to \(\langle P \rangle\); if some user’s temporary private key is multiple of some other’s, then \(P\) and \(Q\) are linearly dependent. In our protocol, the second parameter of bilinear pairing is always \(P_{\text{pub}}\). When we configure the global parameters, we can set it outside \(\langle P \rangle\), such as \(P_{\text{pub}} = sQ, Q \not\in \langle P \rangle\). The session key is \(K = e(P, Q)^{(a_1d_2 + a_2d_3 + \cdots + a_n d_1)}\). Under the MDBDH assumption, the modification still holds forward secrecy. Thus, our protocol can be implemented by supersingular/nonsupersingular curves.

In addition, there are some other improvements to resist existing attacks. Shim\(^8\) proposed to use the long-term private keys to sign \(T_i\) and \(D_i\), which can resist the impersonation attacks mentioned in Refs. \([7, 8]\). However, Ref. \([13]\) proposed that such long-term private keys cannot resist replay attacks. Park proposed that the user index can be randomized by KGC\(^{12}\); each user calculates \(D_i\) by new index\(^{7,8}\). This improvement can resist impersonation attacks to some extent, but relies too much on KGC, and massive encryption/decryption operations make KGC a bottleneck. Du et al.\(^{25}\) proposed that all users maintain a counter together, increase it by 1 when agreement occurs, and participants use the product of the counter and the original long-term private key as the new private key to make every \(\langle P_i, T_i \rangle\) different. Thus, users can detect replayed authentication message in Round 1, but system synchronization is heavy to maintain. Choi and Li proposed to integrate protocol messages of current run into the signature\(^{10,11,13}\), and this is by far the most comprehensive method. However, Choi’s improvement requires two signatures\(^{13}\); in fact, the signature of Round 2 also provides verification of messages in Round 1.

Table 1 provides a comprehensive comparison of several improvements. Our protocol does not need either extra KGC assistance or a global counter, and is compatible with non-supersingular curves. Compared with other improved methods, our protocol applies more widely.

Table 2 compares the amount of calculation operations required of individual participants, including point multiplication, point addition, hash calculation, and bilinear pairing calculation. The multiplication and exponentiation in the session key calculation are the same in each protocol and are not compared.

Because the data of Round 1 can be verified by KGC, the calculation payload of individual participant of Park’s protocol is minimal. But, in Park’s protocol, there is an extra public key encryption operation in Round 2. On the whole, our protocol is the most efficient.

### 5 Conclusions

Our proposed protocol takes full advantage of the data transmitted at various stages of the protocol and provides both entity authentication and freshness. We extend authenticators from three neighbors to all nodes in the group. Our protocol can resist various known internal impersonation attacks and external attacks. In our protocol, the global public key can be placed in the noncyclic group, thus the protocol can be implemented by supersingular/non-supersingular elliptic curve.

Asymmetric identity based group key agreement protocols and related security models have been proposed by some scholars\(^{26,27}\), which are different from previous methods of session key agreement. In this type of protocol, participants will negotiate a common group key, but get a different decryption key individually. It is a very significant research direction.

### Acknowledgements

This work was supported by the Key Project of National Defense Basic Research Program of China (No. B1120132031).
Table 2 Comparison of the amount of calculation required by individual participants.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Point multiplication</th>
<th>Point addition</th>
<th>Hash</th>
<th>Bilinear pairing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Park and Choi[12]</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Du et al.[25]</td>
<td>$n + 4$</td>
<td>$3n - 1$</td>
<td>$n$</td>
<td>4</td>
</tr>
<tr>
<td>Choi[13]</td>
<td>$n + 7$</td>
<td>$3n + 7$</td>
<td>$n + 3$</td>
<td>5</td>
</tr>
<tr>
<td>Li and He[11]</td>
<td>$n + 4$</td>
<td>$n + 1$</td>
<td>$n$</td>
<td>$2n - 1$</td>
</tr>
<tr>
<td>This protocol</td>
<td>$n + 4$</td>
<td>$3n$</td>
<td>$n$</td>
<td>4</td>
</tr>
</tbody>
</table>

References


Kangwen Hu is a master student in the School of Computer at the Beijing Institute of Technology. He received his BEng degree from Beijing Institute of Technology in 2006. His current research interest is software security.

Jingfeng Xue received the PhD degree from Beijing Institute of Technology in 2003. He is currently the professor with the School of Software, Beijing Institute of Technology. His current research interests is software security.
Changzhen Hu received the PhD degree from Beijing Institute of Technology in 1996. He is currently the professor with the School of Software, Beijing Institute of Technology. His current research interest is information security.

Rui Ma received the PhD degree from Beijing Institute of Technology in 2002. She is an associate professor with the School of Software, Beijing Institute of Technology. Her current research interests include software security and Internet of Things.

Zhiqiang Li received the MEng degree from Beijing Institute of Technology in 2003. He is a lecturer with the School of Software, Beijing Institute of Technology. His current research interests include software security and big data.