A Certificateless-based One-Round Authenticated Group Key Agreement Protocol to Prevent Impersonation Attacks

Huimin Ren[1], Suhyun Kim[2], Daehee Seo[3], and Imyeong Lee[1*]
[1] Department of Software Convergence, Soonchunhyang University
Asan 336745, South Korea
[e-mail: ren1732413936@gmail.com]
[2] National IT Industry Promotion Agency
Jincheon 27872, South Korea
[e-mail: ksh34@nipa.kr]
[3] Department of Faculty of Artificial Intelligence and Data Engineering, Sangmyung University
Jongno-gu 35005, South Korea
[e-mail: daehseo@smu.ac.kr]
[*]Corresponding author: Imyeong Lee

Received March 24, 2022; revised March 30, 2022; accepted April 24, 2022; published May 31, 2022

Abstract

With the development of multiuser online meetings, more group-oriented technologies and applications for instance collaborative work are becoming increasingly important. Authenticated Group Key Agreement (AGKA) schemes provide a shared group key for users with after their identities are confirmed to guarantee the confidentiality and integrity of group communications. On the basis of the Public Key Cryptography (PKC) system used, AGKA can be classified as Public Key Infrastructure-based, Identity-based, and Certificateless. Because the latter type can solve the certificate management overhead and the key escrow problems of the first two types, Certificateless-AGKA (CL-AGKA) protocols have become a popular area of research. However, most CL-AGKA protocols are vulnerable to Public Key Replacement Attacks (PKRA) due to the lack of public key authentication. In the present work, we present a CL-AGKA scheme that can resist PKRA in order to solve impersonation attacks caused by those attacks. Beyond security, improving scheme efficiency is another direction for AGKA research. To reduce the communication and computation cost, we present a scheme with only one round of information interaction and construct a CL-AGKA scheme replacing the bilinear pairing with elliptic curve cryptography. Therefore, our scheme has good applicability to communication environments with limited bandwidth and computing capabilities.

Keywords: Certificateless, Authenticated group key agreement, One-Round, Pairing-Free, ECC, Impersonation Attacks.

http://doi.org/10.3837/tiis.2022.05.015 **ISSN : 1976-7277**

1. Introduction

With the accelerated development of network communication technology, group communication provides the services required for distributed application systems, such as online conferences, and collaborative work systems. Because these applications usually transmit information in an open network environment, communication data can easily be hijacked, eavesdropped upon, or tampered with. Therefore, how to provide users with secure group communication services in an open and insecure network environment is very important. Ensuring the privacy of communication between multiple participants is a basic security guarantee. Using authenticated group key agreement (AGKA) protocols to negotiate group key is a common method. An AGKA protocol enables multiple participants to negotiate the same group key through an algorithm in an open and insecure channel [1]. This group key can only be generated by authenticated users, who can use it to encrypt and decrypt data to ensure the security of their communications.

On the basis of the Public Key Cryptography (PKC) system used, the AGKA protocols can be divided into Public Key Infrastructure (PKI)-based [2-4], Identity (ID)-based [5-7], and Certificateless (CL) [8-10]. As a CL-PKC system can solve the certificate management overhead and key escrow problems of the first two PKC systems, CL-AGKA protocols have become a research focus. However, most CL-AGKA protocols are vulnerable to public key replacement attacks because of the lack of public key authentication. How to resist such attacks is another research focus.

In addition to security, reducing communication and computational overheads to improve scheme efficiency is an important research direction within CL-AGKA protocols. When the group is large, the number of communication rounds is the major regard, because the number of communication rounds directly affects the users' processing time and communication efficiency. Many AGKA schemes with a constant number of rounds have been proposed [11,12]. In order to improve efficient, in addition to reducing communication complexity, reducing computation complexity is also very important. So given the high computational complexity of the pairing operation, more researchers have begun to investigate how to reduce pairing in CL-AGKA schemes.

This paper proposes an AGKA protocol based on CL-PKC that prevents an impersonation attack caused by public key replacement. Moreover, to improve the efficiency, we propose an efficient group key agreement protocol with only one round of communication. And we propose a scheme replacing the bilinear pairing with their large computational complexity with Elliptic Curve Cryptography (ECC), which is of lower computational complexity.

Our proposed scheme has the following prominent merits:

- Our protocol can solve the certificate management overhead of PKI and key escrow problems of ID-PKC by using CL-PKC.
- Strong security: Our CL-AGKA scheme not only supplies Mutual Authentication (MA) and Public Key Authentication (PKA), but also can provide Impersonation Attack Prevention (IAP).
- Efficient performance: We propose a one-round CL-AKGA protocol to reduce the communication overheads and we replace the bilinear pairing with ECC to improve computing efficiency.

We have structured this paper as follows. First, in Section 2, we present the related works of AGKA protocols. Second, we explain the relevant features of ECC, the system model for

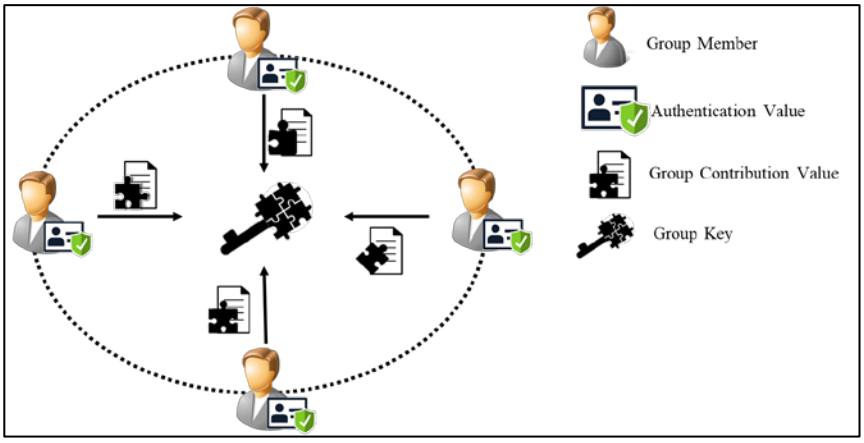

Fig. 1. Authentication and group key agreement.

CL-AGKA, our security model, security requirements in Section 3. The proposed one- round CL-AGKA protocol is represented in Section 4. Then we describe security and performance analyses of our CL-AGKA protocol in Section 5. Lastly, we conclude with Section 6.

2. Related Works

An AGKA protocol is a cryptographic primitive that can realize secure group communication. This communication protocol enables multiple users involved in group communication to negotiate a group key in an open and insecure network space. **Fig. 1** depicts the process of AGKA. Under AGKA protocols, each authenticated user can generate a group key by sharing contributions from every group member in advance. Then group members can then encrypt the subsequent communication content using the same group key to ensure the confidentiality and integrity of group communication.

In our scheme, we classify AGKA protocols according to Public Key Cryptography (PKC) system and the round of communication rounds.

2.1 Classification of AGKA Schemes by The PKC System

A PKC cryptography system can be used for encryption and signature generation using public and private keys [13]. The AGKA scheme uses a PKC system to realize authentication and key agreement. Based on the PKC system used, AGKA schemes can be divided into traditional PKI-based, ID-based, and CL-based schemes.

In a PKC system if users' public keys cannot be authenticated, the system is vulnerable to man-in-the-middle attacks. PKI cryptography system was proposed [14] to overcome this problem, which uses certificates to authenticate the users' public keys. In this system, digital certificates are issued a Certification Authority (CA) to bind the identity information and public key of users. Users can verify the authenticity of the public key through this certificate to prevent man-in-the-middle attacks. However, there are overhead problems in these PKI systems because they require the generation, distribution, storage, and revocation of certificates.

In order to overcome the complexity of certificate management and use, in 1984 Shamir proposed an ID-PKC system [15]. In ID-PKC system, the identity of a user functions as its public key, so that users do not need to apply for and use certificates to authenticate their public keys. However, in ID-PKC systems, the private keys of users are calculated and produced by

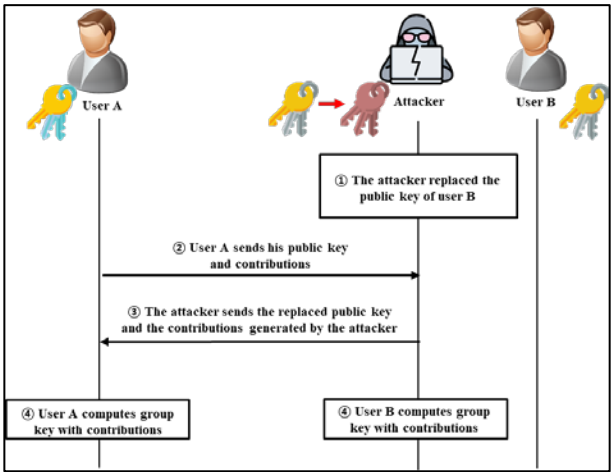

Fig. 2. Public key replacement attack in a certificateless based key agreement protocol.

Key Generation Center (KGC) which is a trusted third-party. This introduces a problem that a malicious KGC could use a user's private key to view his information at any time, which called the key escrow problem.

Demand has risen for CL-PKC systems that solve the key escrow problems of ID-PKC system in 2003, Al-Riyami proposed a PKC system without a certificate which was a hybrid of traditional PKI and ID-PKC where users' private keys are decided jointly by both the KGC and the users themself [16]. The KGC cannot calculate the partial key generated by the user, and the user cannot calculate the partial private key generated by the KGC. Thus, this CL-PKC system can overcome the problems of key escrow. Also, the CL-PKC system removes the overhead problems caused by certificate management in PKI system. Therefore, more and more AGKA protocols based on CL-PKC system have been proposed [17-19]. CL-AGKA schemes are greatly improved compared with the PKI-AGKA and ID-AGKA schemes.

However, in the CL-PKC system, no certificate proves the authenticity of the public key, and it is easy for malicious attackers to impersonate users via public key replacement attacks. As shown in **Fig. 2**, in a CL-PKC based two-party key agreement scheme, a malicious attacker can replace the user B's public key and cannot be discovered. Then the attacker can forge as user B to negotiate session key with user A. Finally, the attacker can use the session key to communicate with user A. Similarly, the PKRA will occur in a CL-AGKA protocol. Therefore, research on CL-AGKA to prevent public key replacement attacks is in progress [20, 21].

Our scheme proposes an AGKA protocol based on CL-PKC that solves the problem of impersonation attacks caused by public key replacement.

2.2 Classification of AGKA Schemes According to The Number of Communication Rounds

The AGKA protocols can be divided into constant-round and non-constant-round types, according to whether their number of communication rounds is stable. The number of communication rounds is an important indicator used to quantify the processing time online for users in such protocols. The number of communication rounds means the number of information interactions between users. In the process of group key agreement, each user must wait for the end of all communication rounds to calculate the group key in order to complete the key agreement [22]. A constant-round AGKA protocol has a fixed number of communication rounds, where this number is independent of the size of group. This reduces

the time of network communication for users and improves communication efficiency.

In a non-constant-round AGKA [23-25], the number of communication rounds increases either linearly or logarithmically with the number of users. Thus, more members mean more rounds of communication. The efficiency of such agreement schemes is very low. For example, in a AGKA protocol based on a binary tree structure, the number of communication rounds growth logarithmically with the number of users [26]. That is for a group with 1,000 members, the number of rounds of information interactions during the key negotiation process is at least $log2\ (1000)\ \approx\ 300$. All group members must remain online to perform the key agreement algorithm during all the communication rounds. Therefore, the users' time of network communication is long so that the communication efficiency is very low. However, in a AGKA protocol with a constant number of communication rounds, the time of network communication will not increase with the number of users. For example, such a scheme [27] might require only one communication round, which means the group members need only send and receive messages once during the communication process, no matter how many members there are in the group. Therefore, a constant-round AGKA protocol can significantly reduce the time of network communication while ensuring system security and thus has broad applicability.

In 1994, Burmester and Desmedt presented a protocol for constant-round group key agreements regardless of the number of members [28]. However, in this protocol, the algorithm only verifies the accuracy of the member $\boldsymbol{U_i}$'s signature value index $\boldsymbol{z_i}$, it does not authenticate the identity of the members, leaving it vulnerable to impersonation attacks. In 2003, Katz and Yung [29] proposed a protocol for authentication using five rounds of communication to agree on the group keys. However, this process only authenticates the correctness of the index value $K = \alpha^{S_A x' + S_B x}$ based on the discrete logarithm problem , it does not provide mutual authentication requires to verify the users' identities. It's meaning that this scheme cannot prevent impersonation attacks from malicious group members.

3. Preliminaries

This section describes key previously defined concepts used in our proposed scheme.

3.1 Elliptic Curve Cryptography

We define an elliptic curve E on a finite field F_p according to the following equation:

$$y^2 = x^3 + a \cdot x + b(mod\ p) \tag{1}$$

Where p is a prime number, and a and b are two nonnegative integers smaller than p. They satisfy following:

$$\Delta = (4a^3 + 27b^2) mod\ p \neq 0 \tag{2}$$

The points (x, y) satisfying (1) on $E\ /\ F_p$ and an infinite point O form a group G.

$$G = \left\{(x, y): x, y \in F_p\ and (x, y) \in \frac{E}{F_p}\right\} \cup \{O\} \tag{3}$$

A detailed description of elliptic curve cryptography may be found in previous works [30, 31].

It is difficult to solve the following problems defined on G in polynomial time and the security of various cryptographic scheme is based on the difficulty of solving these problems.

- **Problem 1, Elliptic Curve Discrete Logarithm Problem (ECDLP)**: We assume that an elliptic curve contains a large prime subgroup of order p, which is big enough to make

solving discrete logarithms in F_p unfeasible. Suppose we have two points P, Q on E, and let $Q = a \cdot P$, where a is an integer. According to the ECDLP, it is not computationally feasible to find a from $Q = a \cdot P$ given Q and P.

- **Problem 2, Computational Diffie–Hellman Problem (CDHP)**: Given a generator P of G and $(a \cdot P, b \cdot P)$ for unknown $a, b \in Z_p$, The probability of any polynomial-time algorithm finding $a \cdot b \cdot P$ is negligible.
- **Problem 3, One-Way Hash Function (OWHF)**: Given the output h of a one-way hash function, it is hard to obtain any message m such that $h = H(m)$.

3.2 System Model for Certificateless Authenticated Group Key Agreement

CL-PKC systems were proposed in 2003 to prevent the key escrow problem of the ID-PKC systems. Later, many AGKA schemes based on CL-PKC have been proposed. At present, the research of CL-AGKA schemes have become a hot issue.

In CL-AGKA, the KGC and the user himself generated users' private keys together. The KGC generates a key pair and private key of that pair is used as the user's partial private key. The remainder of the user's private key is a random secret value selected by the user and known only to the user. The user combines their secret value with the partial key from the KGC to calculate the user's public key. Then in CL-AGKA system, user verify each other and agree on a group key with shared contributions.

Typically, a CL-AGKA protocol consists of the following algorithms [32].

- **Setup**: This process is performed by the KGC. It inputs a security parameter and generates the master private key of KGC and the system parameters.
- **Partial-Private-Key-Extract**: This process is performed by the KGC. It takes system parameters and an identity of user as input and computes the user's partial private key
- **Set-Secret-Value**: This process is performed by the user. This process accepts system parameters and a user's identity as inputs and returns the user's secret value.
- **Set-Private-Key**: This process is also performed by the user. It uses the system parameters, user's identity, partial private key generated by the KGC, and a secret value selected by the user to generate the user's private key.
- **Set-Public-Key**: This process is also performed by the user. It uses the system parameters, the user's identity value and private key to generate the user's public key.
- **Authenticated-Group-Key-Agreement**: This process is also performed by the user. It allows users to authenticate the other users involved and agree on a single group key. It uses the system parameters, the users' identity, and public and private keys to generates a group key.

3.3 Security Model

In PKC, the security of a system can be ranked according to one of three levels [33]:

- The KGC can know or calculate the users' private keys and can use the users' private keys to pretend that a user is completing operations.
- The KGC doesn't know and cannot calculate the private keys of users but can forge any user's public key to impersonate the user.
- The KGC doesn't know and cannot calculate the private key of users and cannot forge the user's public key to impersonate the user.

ID-PKC systems, CL-PKC systems, and PKI systems fall into levels 1, 2, and 3 respectively. Most CL-AGKA schemes [34, 35] also have a secondary security level because the KGC doesn't know or cannot calculate the users' private keys but can forge the public keys of users to impersonate a user. Therefore, attacks by two kinds of adversaries are typically considered

in CL-AGKA schemes: a general user attack adversary $\mathcal{A}_{\mathrm{I}}$ and a malicious KGC attack adversary $\mathcal{A}_{\mathrm{II}}$ [16].

$\mathcal{A}_{\mathrm{I}}$: The KGC is trusted. The adversary cannot obtain the KGC's master private key or generate the partial private key of a user, but they can replace the public key of any user.

$\mathcal{A}_{\mathrm{II}}$: The KGC is untrusted. The adversary can obtain the KGC's master private and generate the partial private key of a user but cannot replace the user's public key or request their secret value.

In CL-PKC systems, a partial private key of user is made by the KGC. Although a malicious KGC cannot calculate the entirety of a private key of user, it can leak the partial private key to adversaries, who can then execute public key replacement attacks. In this way, they can negotiate the group key as legitimate users. To expand the security levels, an $\mathcal{A}_{\mathrm{II}}$ attack based on KGC untrustworthiness can be subdivided into those based on active and passive malicious KGCs.

Active Malicious KGC: The KGC is untrustworthy. The adversary can get the master private key of the KGC and generate user partial private keys but cannot replace users' public keys.

Passive Malicious KGC: The KGC is untrustworthy. The adversary can obtain the master key of the KGC and generate users' partial private keys but cannot replace and leak users' partial private keys to an external adversary.

With this expansion, the attackers of CL-AGKA schemes can be divided into three types, as follows [36]:

- $\mathcal{A}_{\mathbf{I}}$: The KGC is trusted. The adversary $\mathcal{A}_{\mathrm{I}}$ cannot get the KGC's master private key and cannot get the users' private keys but can replace the public key of any user.
- $\mathcal{A}_{\mathbf{II}}$: The KGC is untrusted and active malicious. The adversary $\mathcal{A}_{\mathrm{II}}$ can obtain the master private key of KGC and then generate partial private keys of users but cannot replace the public keys of users.
- $\mathcal{A}_{\mathbf{III}}$: The KGC is untrusted and passive malicious. The malicious KGC can collude with some malicious users. The malicious KGC can generate users' partial private keys with master private key, and leaks users' partial private keys to the malicious users $\mathcal{A}_{\mathrm{III}}$. And the malicious user $\mathcal{A}_{\mathrm{III}}$ can replace users' public keys.

3.4 Security Requirements

CL-AGKA should be designed to meet the following security requirements. If a protocol satisfies all of them, a system can be protected from threats such as man-in-the-middle, public key replacement, and impersonation attacks.

3.4.1 Mutual Authentication (MA)

Mutual authentication is a procedure in which both parties participating in a communication protocol can identify each other. In AGKA protocols group members must authenticate each other's identities before agreeing on the group key to against man-in-the-middle attacks. After authentication they can proceed securely with group key agreement.

3.4.2 Public Key Authentication (PKA)

In CL-PKC systems, because there is no certificate for public keys, the authenticity of public keys cannot be verified, and they are weak to public key replacement attacks. Therefore, a

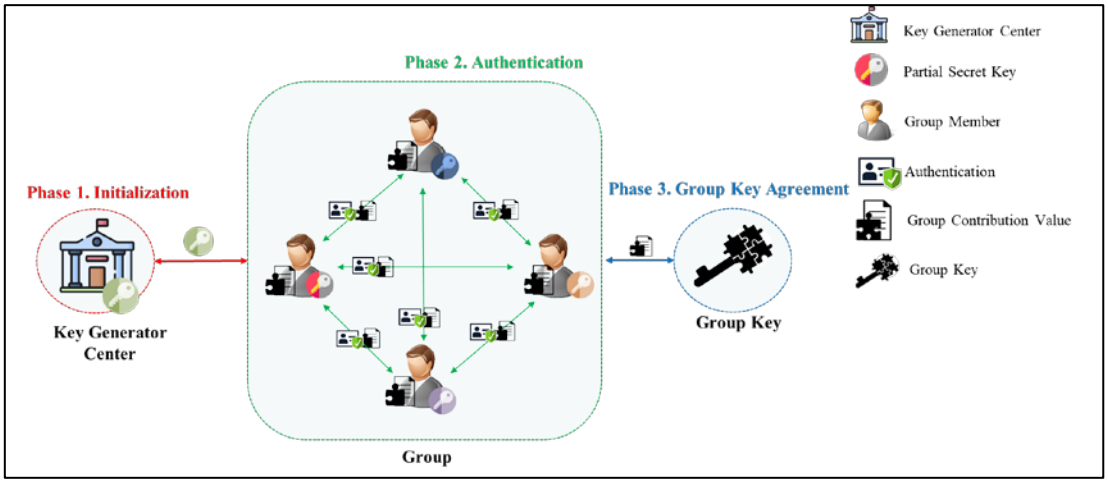

Fig. 3. Proposed scheme overview.

CL-AGKA protocol requires a function to verify others' public keys before communicating with them. Through this mechanism, it is possible for entities to confirm the authenticity of each other's public keys and prevent public key replacement attacks.

3.4.3 Impersonation Attack Prevention (IAP)

An attacker involved in impersonation attack involves has access to all publicly available information. He tries to accomplish a protocol with one user by impersonating another user. Recall that a AGKA scheme is successful if each of the parties accepts the identities of the others and compute with the same key.

4. Proposed CL-AGKA Scheme

In this section, we present a secure and efficient one-round CL-AGKA scheme. Our proposed scheme solves the problems of certificate overhead in PKI-AGKA schemes and key escrow problems in ID-AGKA schemes. It provides mutual authentication by verifying users' public keys and signatures also even when the KGC is not reliable it is secure resist impersonation attacks. Furthermore, our scheme reduces computation by using the concepts of ECC, which are more efficient than pairing operations. It also reduces the latency of the group key agreement process by completing this with a single round of communication. Our one-round CL-AGKA scheme can be divided into three phases as shown in **Fig. 3**: Initialization, Authentication, and Group Key Agreement. And the system symbols for this scheme are listed in **Table 1**.

4.1 Initialization

As shown in **Fig. 4**, the initialization phase is composed of six steps: Setup, Set-Secret-Value, Set-Public-Value, Partial-Private-Key-Extract, Set-Private-Key, and Set-Public-Key. In our scheme, users set the secret values by themselves first. Then KGC must use users' secret value as input into the Set-Partial-Secret-Key step in order to generate the users' partial private keys. Therefore, the KGC cannot fully control the generation of the partial private keys to prevent malicious KGC from forging partial private keys and not being found.

- **Setup**: In this step, the KGC generates the system parameters with a given security parameter: an integer k. The following steps are implemented: The KGC chooses a k-bit prime number p then defines an elliptic curve E over a prime finite field F_p of prime

Table 1. List of Notations Used in the Proposed Scheme

Symbol	Description
k	Security parameter
p	k-bits prime number
E/F_p	Elliptic Curve over a prime finite field F_p
G_p	Addition group of E/F_p with prime order p
Z_p^*	Multiplicative group of integers modulo p
P	Generator of group G_p
P_{pub}, s	KGC's master public key and KGC's master private key
ID_i	User U_i's identifier
pk_i	User U_i's public key, $pk_i = (X_i, R_i, Z_i)$
sk_i	User U_i's private key, $sk_i = (x_i, z_i)$
$H_1(\bullet)$	Hash function, $\{0,1\}^*$ x G_p x $G_p \rightarrow Z_p^*$
$H_2(\bullet)$	Hash function, $Z_p^* \rightarrow Z_p^*$
$H_3(\bullet)$	Hash function, Z_p^* x $G_p \rightarrow Z_p^*$
$H_4(\bullet)$	Hash function, $\{0,1\}^*$ x G_p x G_p x $Z_p^* \rightarrow Z_p^*$
$H_5(\bullet)$	Hash function, $\{0,1\}^* \rightarrow \{0,1\}^k$
sig_{ij}	User U_i's signature
t_i	User U_i's temporary private key
T_i	User U_i's temporary public key
GK	The group key
$Params$	System parameters, $\{p, E, G_p, P, P_{pub}, H_1(\bullet), H_2(\bullet), H_3(\bullet), H_4(\bullet), H_5(\bullet)\}$

numbers. And the KGC lets G_p be an additive group formed by the points on E/F_p. The KGC randomly selects a generator $P \in G_p$. Then the KGC chooses a random value $s \in Z_p^*$ as the KGC's master private key and uses s to computes the master public key, the KGC uses the master private key to compute the master public key P_{pub}. And the KGC selects five secure hash functions as (5-9). Finally, the KGC publishes system parameters as (10).

$$P_{pub} = s \cdot P \tag{4}$$

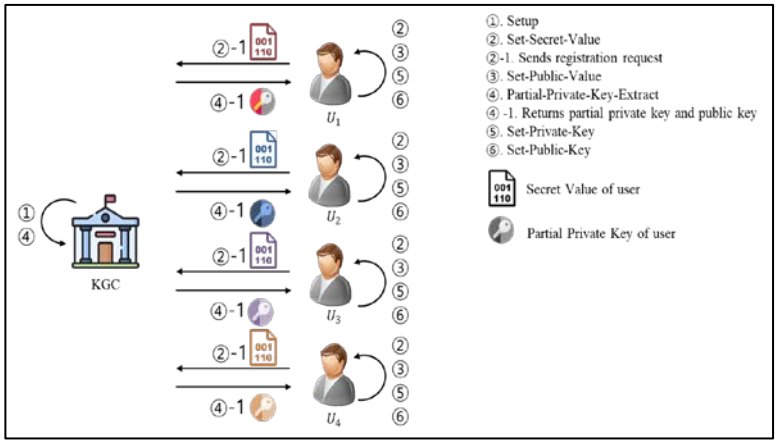

Fig. 4. Initialization Phase.

$$H_1(\bullet) : \{0,1\}^* \times G_p \times G_p \rightarrow Z_p^* \quad (5)$$

$$H_2(\bullet) : Z_p^* \rightarrow Z_p^* \quad (6)$$

$$H_3(\bullet) : Z_p^* \times G_q \rightarrow Z_p^* \quad (7)$$

$$H_4(\bullet) : \{0,1\}^* \times G_p \times G_p \times Z_p^* \rightarrow Z_p^* \quad (8)$$

$$H_5(\bullet) : \{0,1\}^* \rightarrow \{0,1\}^k \quad (9)$$

$$\text{Params} = \begin{Bmatrix} E/F_p, G_p, P, P_{pub}, H_1(\bullet), H_2(\bullet), \\ H_3(\bullet), H_4(\bullet), H_5(\bullet) \end{Bmatrix} \quad (10)$$

- **Set-Secret-Value**: A user U_i randomly selects a value $x_i \in Z_p^*$ as his secret value.

$$x_i \in Z_p^* \quad (11)$$

- **Set-Public-Value**: U_i uses x_i to compute a public value X_i.

$$X_i = x_i \cdot P \quad (12)$$

- **Set-Partial-Private-Key-Extract**: This step is performed by the KGC. User U_i send $<ID_i, X_i>$ to the KGC. The KGC selects a random value $r_i \in Z_p^*$ and multiplies r_i by P to compute a partial public key R_i for U_i. The KGC inputs $<r_i, ID_i, X_i, R_i, s>$ into (14) and computes the partial private key z_i by using elliptical multiplication and hash function $H_1(\bullet)$. KGC sends z_i through a secure channel to U_i. Then KGC uses z_i to calculate the public key Z_i for verification by (15) and sends it with an insecure channel to U_i. After receiving z_i and Z_i, user U_i verifies (16) through elliptical multiplication to authenticate the validity of z_i.

$$R_i = r_i \cdot P \quad (13)$$

$$z_i = r_i + s \cdot H_1(ID_i \parallel X_i \parallel R_i) \quad (14)$$

$$Z_i = z_i \cdot P \quad (15)$$

$$z_i \cdot P \stackrel{?}{=} R_i + H_1(ID_i \parallel X_i \parallel R_i) \cdot P_{pub} \quad (16)$$

- **Set-Private-Key**: This step is performed by the users. U_i sets his full private key S_i as follows:

$$S_i = (x_i, z_i) \quad (17)$$

- **Set-Public-Key**: This step is performed by the users. U_i sets his full public key P_i. Then KGC publishes $U_i's$ public keys $<X_i, R_i, Z_i>$:

$$P_i = (X_i, R_i, Z_i) \quad (18)$$

4.2 Authentication

The second authentication phase is run by users, and it is composed of four steps: set-temporary-public-key, sign, verify-public-key, and verify-signature. As shown in **Fig. 5**, all users who want to participate in group communication conduct mutual authentication before proceeding with the group key agreement.

- **Set-Temporary-Public-Key**: U_i randomly selects a temporary private key t_i and computes a temporary public keyT_i by using (19). And users make the value $H_2(t_i)$ as a group key contribution that can be used for negotiating a group key. The confidentiality of temporary private key t_i is guaranteed via the OWHF and ECDLP.

$$T_i = H_2(t_i) \cdot P \quad (19)$$

- **Sign**: U_i uses (20) to compute a value h_{ij} by inputting his secret value x_i and the receiver user $U_j's$ public value X_j which was published by KGC. Then values $H_2(t_i), h_{ij}$ and z_i are taken as the inputs to compute value ∂_{ij} in (21). Then U_i uses (22) to compute

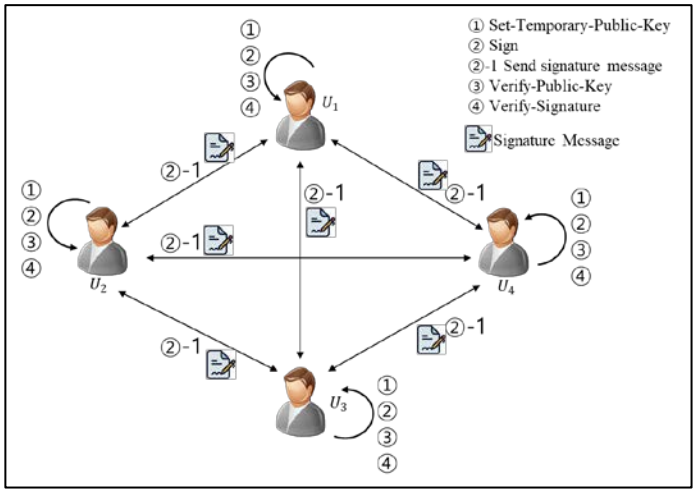

Fig. 5. Authentication Phase.

signature sig_{ij} and sends Msg_{ij}<ID_i, T_i, ∂_{ij}, sig_{ij}> to the receiver U_j. And the value h_{ij} cannot be calculated without the U_i's private key x_i and U_j's private key x_j.

$$h_{ij} = H_3(x_i \cdot X_j) \tag{20}$$

$$\partial_{ij} = \frac{H_2(t_i)}{h_{ij} + z_i} \tag{21}$$

$$sig_{ij} = H_2(t_i) \oplus H_4(ID_i \parallel Z_i \parallel T_i \parallel h_{ij}) \tag{22}$$

- **Verify-Public-Key**: U_j first uses (23) to verify $U_i's$ public key P_i which is published by KGC.

$$Z_i \stackrel{?}{=} R_i + H_1(ID_i, X_i, R_i) \cdot P_{pub} \tag{23}$$

- **Verify-Signature**: On receiving Msg_{ij}<ID_i, T_i, ∂_{ij}, sig_{ij}> from U_i. If the result of Verify-Public-Key is true, U_j verifies U_i's signature using the following equation:

$$\left(sig_{ij} \oplus H_4(ID_i \parallel Z_i \parallel \partial_{ij} \cdot (h_{ji} \cdot \mathrm{P} + Z_i) \parallel h_{ji})\right) \cdot P \stackrel{?}{=} T_i \tag{24}$$

4.3 Group Key Agreement

If the result of Verify-Signature is true, users negotiate a common group key GK using contribution values of all users. If the scheme is performed successfully, all the group keys computed by the members will have the same value.

- **Group-Key-Agreement**: U_j calculates $H_2(t_i)$ using (25) and takes $H_2(t_i)$ as input and computes GK, with which group members can perform secure communication.

$$\begin{aligned} H_2(t_i) &= sig_{ij} \oplus H_4(ID_i \parallel Z_i \parallel T_i \parallel h_{ji}) \\ &= sig_{ij} \oplus H_4\left(ID_j \parallel Z_j \parallel T_j \parallel H_2(x_j \cdot X_i)\right) \\ &= sig_{ij} \oplus H_4\left(ID_j \parallel Z_j \parallel T_j \parallel H_2(x_i \cdot x_j \cdot P)\right) \\ &= sig_{ij} \oplus H_4\left(ID_j \parallel Z_j \parallel T_j \parallel H_2(x_i \cdot X_j)\right) \\ &= sig_{ij} \oplus H_4(ID_j \parallel Z_j \parallel T_j \parallel h_{ij}) \end{aligned} \tag{25}$$

$$GK = H_5(H_2(t_1) \parallel H_2(t_2) \parallel \cdots \parallel H_2(t_n))(i \neq n) \tag{26}$$

5. Security Analysis

5.1 Security Analysis of Proposed Scheme

In this section, we determine whether our scheme meets the security requirements of mutual authentication, public key authentication, and impersonation attack prevention described in Section III. Our scheme's security is based on the ECDLP, CDHP and OWHF.

5.1.1 MA

This scheme provides mutual authentication by verifying users' public key and signatures based on the ECC cryptography system. When this one-round CL-AGKA scheme executes phase 2, user U_j authenticates the public key and signature sent by U_i. U_j can ensure that the message was sent by U_i and verify the identity of U_i. In the phase 2 after receiving Msg_i <ID_i, T_i ,$\mathrm{Sig}_{ij}, \partial_{ij}$> from U_i, user U_j can use the public key Z_i of U_i and their own private key x_j to check whether $(sig_{ij} \oplus H_4(ID_i, Z_i, \partial_{ij}(h_{ji}\mathrm{P} + Z_i) \parallel h_{ji})) \cdot P = T_i$. It means that the Msg_{ij}<ID_i, T_i, ∂_{ij}, sig_{ij}> was sent by U_i and $U_i's$ identity is authenticated by U_j if the equation holds. The proof of verifying the validity of the signature is described by (27). In our scheme, we bind the public key, private key, and identity information of the user in signatures. According to the security of the ECDLP, CDLP and OWHF, if an attacker does not have a private key of the user, then they cannot forge the signature. The correctness of the signature verification can be justified as follows:

$$
\begin{aligned}
&\left(sig_{ij} \oplus H_4\left(ID_i \parallel Z_i \parallel \partial_{ij} \cdot \left(h_{ji} \cdot \mathrm{P} + Z_i\right) \parallel h_{ji}\right)\right) \cdot P \\
&= (sig_{ij} \oplus H_4(ID_i \parallel Z_i \parallel \\
&\frac{H_2(t_i)}{H_3\left(x_i \cdot X_j\right) + z_i}(H_2(x_j \cdot X_i) \cdot \mathrm{P} + Z_i) \parallel H_3(x_j \cdot X_i)) \cdot \mathrm{P} \\
&= (sig_{ij} \oplus H_4(ID_i \parallel Z_i \parallel \frac{H_2(t_i)}{H_3\left(x_i \cdot X_j\right) + z_i} \cdot \\
&(H_3(x_j \cdot x_i \cdot P) \cdot P + Z_i) \parallel H_3(x_j \cdot x_i \cdot P))) \cdot P \\
&= sig_{ij} \oplus H_4(ID_i \parallel Z_i \parallel \frac{H_2(t_i)}{H_3\left(x_i \cdot X_j\right) + z_i} \cdot \\
&(H_3(x_i \cdot X_j) + z_i) \cdot P) \parallel H_3(x_i \cdot X_j)) \cdot P \\
&= \left(sig_{ij} \oplus H_4\left(ID_i \parallel Z_i \parallel H_2(t_i) \cdot P \parallel h_{ij}\right)\right) \cdot P \\
&= \left(sig_{ij} \oplus H_3\left(ID_i \parallel Z_i \parallel T_i \parallel h_{ij}\right)\right) \cdot P \\
&= H_2(t_i) \cdot P \\
&= T_i
\end{aligned} \tag{27}
$$

5.1.2 PKA

In CL-PKC system, because the user's public key has no certificate to prove its authenticity, the main problem for a CL-AGKA protocol is a public key replacement attack. As part of our scheme, we propose a way to prevent public key replacement attacks by malicious users as follows:

- The KGC binds user's public value X_i previously calculated by the user themselves and identity information ID_i when calculating the partial private key of the user U_i. Before the KGC calculates the partial public keys of U_i, U_i first selects a secret value x_i, and then uses x_i to calculate his public value X_i. U_i sends X_i to the KGC by an open channel, and

KGC uses its master private key s and U_i's public value X_i, U_i's partial public R_i and U_i's identity value ID_i to compute partial private key z_i for U_i according to the equation $z_i = r_i + s \cdot H_1(ID_i \parallel X_i \parallel R_i)$. The KGC then uses a secure channel to send the partial private key z_i to U_i. Then KGC uses ECC multiplication to calculate the U_i's public key $Z_i = z_i \cdot P$ and sends it to U_i by a insure channel so that $Z_i = (r_i + s \cdot H_1(ID_i \parallel X_i \parallel R_i)) \cdot P$.

- In this way, the behavior of the KGC with regard to generating partial private keys is restricted by users. Although a malicious KGC know the master private key s and could randomly select r_i' to calculate the public key R_i', if the malicious KGC does not know the partial private key x_i of users, it cannot easily forge the U_i's public key X_i. In our proposed scheme if malicious KGC wants to generate an effective public key X_i'. The equation $R_i + H_1(ID_i \parallel X_i' \parallel R_i) \cdot P_{pub} = Z_i$ must be passed by the user's public key as follows:

$$\begin{aligned} & R_i + H_1(ID_i \parallel X_i' \parallel R_i) \cdot P_{pub} \\ = \ & R_i + s \cdot H_1(ID_i \parallel X_i' \parallel R_i) \cdot P \cdot P_{pub} \\ = \ & \left(r_i + s \cdot H_1(ID_i \parallel X_i \parallel R_i)\right) \cdot P \cdot P_{pub} \\ = \ & Z_i \end{aligned} \tag{28}$$

Thus,

$$\begin{aligned} & H_1(ID_i \parallel X_i' \parallel R_i) \\ = \ & H_1(ID_i \parallel X_i \parallel R_i) \end{aligned} \tag{29}$$

This conclusion violates the strong non-collision of hash functions assumption, so it is impossible for a malicious KGC to compute the partial private key to forge the public key X_i', which meets the conditions needed for preventing a PKRA.

- The KGC calculates the U_i's partial public key R_i, Z_i and publishes the U_i's public key $P_i = (X_i, R_i, Z_i)$ with the system parameters to prevent malicious users from using PKRA. Therefore, at any time, a user can use the equation $Z_i \stackrel{?}{=} R_i + H_1(ID_i \parallel X_i \parallel R_i) \cdot P_{pub}$ to make sure whether a malicious user replaces his public key. Meanwhile, other users can use the same equation to verify whether the public key is indeed the public key of U_i before communicating with U_i. The validation process is as follows:

$$Z_i \stackrel{?}{=} R_i + H_1(ID_i \parallel X_i \parallel R_i) \cdot P_{pub} \tag{30}$$

Proof:

$$\begin{aligned} & Z_i = z_i \cdot P \\ = \ & \left(r_i + s \cdot H_1(ID_i \parallel X_i \parallel R_i)\right) \cdot P \\ = \ & r_i \cdot P + s \cdot H_1(ID_i \parallel X_i \parallel R_i) \cdot P \\ = \ & R_i + H_1(ID_i \parallel X_i \parallel R_i) \cdot P_{pub} \end{aligned} \tag{31}$$

5.1.3 IAP

Impersonation attacks are classified into those by $\mathcal{A}_{\mathrm{I}}$ (who can replace the public key but cannot get the KGC's master private key), $\mathcal{A}_{\mathrm{II}}$ (an untrusted and active malicious KGC who has the master key and can use it to generate a partial private key of the user but cannot replace public key of the user) and $\mathcal{A}_{\mathrm{III}}$ (an untrusted and passive malicious KGC who cannot substitute the public key but can leak the partial private key of the user to another external adversary who can replace the public key of the user). Even if the receiver's public key is replaced by $\mathcal{A}_{\mathrm{I}}$, or the partial private key is determined by $\mathcal{A}_{\mathrm{II}}$, or the partial private key is leaked to another

external adversary who can substitute the receiver's public key, the adversary cannot impersonate a legitimate user.

- **Preventing impersonation attacks by $\mathcal{A}_{\mathrm{I}}$:**

An adversary has succeeded if he can successfully forge the signature and group key used to communicate with other users after authentication. First, $\mathcal{A}_{\mathrm{I}}$ replaces the U_i's public key $P_i = (X_i, R_i, Z_i)$ with $P_i' = (X_i', R_i', Z_i')$ and attempts to forge an effective message Msg_{ij}'<ID_i, $\mathrm{T_i'}$,∂_{ij}' sig_{ij}'>, which can satisfy $(sig_{ij}' \oplus H_4(ID_i \parallel Z_i' \parallel \partial_{ij}' \cdot (h_{ji}' \cdot P + Z_i') \parallel h_{ji}')) \cdot P = T_i'$. However, according to (14,15), we know that the user's public key Z_i is obtained by the ECC multiplication $z_i \cdot P$ using the partial private key z_i, which is generated by KGC. The KGC combines user's public key X_i, identity information ID_i, and the KGC's master key s via a hash function and ECC multiplication. Because $\mathcal{A}_{\mathrm{I}}$ does not know the user's private key x_i or the master key s, $\mathcal{A}_{\mathrm{I}}$ cannot generate a public key Z_i that would pass $Z_i' = R_i + H_1(ID_i \parallel X_i \parallel R_i) \cdot P_{pub}$. Therefore, $\mathcal{A}_{\mathrm{I}}$ cannot undertake an impersonation attack by public key replacement.

- **Preventing Impersonation Attacks by $\mathcal{A}_{\mathrm{II}}$:**

$\mathcal{A}_{\mathrm{II}}$ has the master private key s, and can generates U_i's partial private key z_i, but $\mathcal{A}_{\mathrm{II}}$ cannot replace the user's public key P_i or obtain the U_i's own secret value x_i. If $\mathcal{A}_{\mathrm{II}}$ can impersonate successfully, they can forge an effective signature with information that satisfies the following conditions:

$$sig_{ij}' = \frac{H_2(t_i')}{H_2(x_i' \cdot X_j) + z_i} = \frac{H_2(t_i)}{H_2(x_i \cdot X_j) + z_i} = sig_{ij} \quad (32)$$

Thus,

$$H_2(x_i' \cdot X_j) = H_2(x_i \cdot X_j) \quad (33)$$

and,

$$H_2(t_i') = H_2(t_i) \quad (34)$$

The adversary can obtain the U_i's partial private key z_i but does not know the temporary value t_i and the secret value x_i generated by himself (user). Based on the EDCLP, OWHF and CDLP, the adversary cannot calculate $H_2(x_i' \cdot X_j)$ and $H_2(t_i')$ that are the same as $H_2(x_i \cdot X_j)$ and $H_2(t_i)$. Therefore, the adversary cannot calculate a value sig_{ij}' that is the same as sig_{ij}.

- **Preventing Impersonation Attacks by $\mathcal{A}_{\mathrm{III}}$:**

In this type of attack, the malicious KGC can collude with some malicious users. The malicious KGC can use his master private key s to generate users' partial private keys z_i, and leaks users' partial private keys z_i to the malicious users $\mathcal{A}_{\mathrm{III}}$. The malicious user $\mathcal{A}_{\mathrm{III}}$ can replace users' public keys X_i. They then use the new forged public key X_i', which passes the verification test, for a signature forgery attack. If the adversary successfully falsifies a signature that passes the verification test, the adversary can be authenticated by other users and negotiate a group key that can be used to communicate with them.

Let us assume that after $\mathcal{A}_{\mathrm{III}}$ obtains the partial private key z_i of U_i, he can forge a valid key pair (x_i', X_i'), and his public key can pass the public key verification. Then the conclusions follow:

Table 2. Comparative analysis of paper security requirements

	Hafizu [27]	Kumar [37]	Shan [38]	Semal [39]	Teng [40]	Ours
MA	O	X	X	O	O	O
PKA	X	X	X	X	X	O
IAP($\mathcal{A}_{\mathrm{I}}$)	O	O	O	O	O	O
IAP($\mathcal{A}_{\mathrm{II}}$)	O	O	O	O	O	O
IAP($\mathcal{A}_{\mathrm{III}}$)	X	X	X	X	X	O
MA: Mutual Authentication						
PKA: Public Key Authentication						
IAP: Impersonation Attacks Prevention						

$$\begin{aligned} & R_i + H_1(ID_i \| X_i' \| R_i) \cdot P_{pub} \\ = \; & r_i \cdot P + H_1(ID_i \| x_i' \cdot P \| R_i) \cdot P_{pub} \\ = \; & r_i \cdot P + H_1(ID_i \| x_i \cdot P \| R_i) \cdot P_{pub} \\ = \; & (r_i + H_1(ID_i \| X_i \| R_i) \cdot s) \cdot P \\ = \; & z_i \cdot P \end{aligned} \tag{35}$$

Thus,

$$H_1(x_i' \cdot P) = H_1(x_i \cdot P) \tag{36}$$

Equation (36) violates the strong non-collision of hash functions assumption. In (35, 36), we can know even if an adversary obtains U_i's partial private key z_i via a malicious KGC, the adversary cannot forge a valid public key X_i' which can pass public key verification. Also, according to the ECDLP, even if the adversary knows the U_i's public key X_i, they cannot calculate the user's private value x_i by $X_i = x_i \cdot P$. Therefore, our scheme can resist $\mathcal{A}_{\mathrm{III}}$ impersonation attacks under the assumptions of the ECDLP and strong non-collision of the hash functions assumption.

5.2 Comparison of Schemes

Table 2 shows a comparison of our proposed one-round CL-AGKA scheme with existing schemes according to various security requirements. "O" and "X" mean that the scheme achieves or does not meet each of these, respectively.

Consulting **Table 2** regarding mutual authentication, we see that in schemes [37] and [38], users cannot authenticate the other group members that communicate with them. Mutual Authentication cannot be completed in the communication process. Analysis performed in [38] indicates that, in [37], users send their contributions to all other users in the second round of communication after passing authentication in the first round. However, because identity authentication is performed in the first round, the message in the second round cannot reflect any fresh information, and a signature cannot resist a reply attack initiated by malicious participants. If there are two malicious users U_{i-1} and U_{i+1} in the group, they will have participated in the normal protocol in that group and recorded all the messages for that process. Thus, U_{i-1} and U_{i+1} can forge the signature of a legitimate user U_i to pass authentication and then negotiate the new group key with other users as if they were U_i. Therefore, there is a risk of forging a signature for the mutual authentication in [38].

In terms of public key authentication, schemes [27, 37-40] do not provide a way to authenticate the public keys of users communicating with them. In such schemes, the partial

public and of users' private keys are computed by the KGC using its own master key, which is not controlled by users. Therefore, a malicious KGC can forge valid public and private keys without user constraints and then forge the signatures of users. In the scheme of [27], the KGC cannot arbitrarily generate the public and private keys that can forge valid signatures since when the KGC computes the partial private and public keys of users, it binds the public key information pre-generated by the user. However, because the user's public key cannot be authenticated, external adversaries can still replace the user's public key without being discovered.

In terms of the security requirements for impersonation attack prevention, an adversary $\mathcal{A}_{\mathrm{I}}$ can replace the user's public key. Without either the KGC master key nor the partial private key of the user. If the adversary $\mathcal{A}_{\mathrm{I}}$ can impersonate successfully, they can forge an effective signature to pass the authentication. Under [27, 37-40], it is easy to execute public key replacement attacks and remain undiscovered. However, while the adversary can replace the public key of user, this is impossible because he cannot obtain the private key of the user and the KGC's master key. Due to the ECDLP, an adversary cannot calculate the user's partial private key or forge an authenticated signature. Therefore, in the attack model for first type of adversary, these schemes are secure.

In the second attack model, $\mathcal{A}_{\mathrm{II}}$ has the KGC's master private key, which can generate a user's partial private key. However, $\mathcal{A}_{\mathrm{II}}$ cannot replace the public key of the user nor can they obtain the partial private key which is computed by the user themselves. If $\mathcal{A}_{\mathrm{II}}$ can impersonate successfully, then they can forge an effective signature that passes authentication. In schemes [27, 37-40], even if the adversary knows that the partial private keys computed by the KGC for the users, they cannot replace the users' public keys. The adversary cannot generate a valid key for forging a signature based on the ECDLP. Therefore, in the second attack model, these schemes are secure.

In the third attack model, $\mathcal{A}_{\mathrm{III}}$ has the master key of the KGC, which can generate the partial private key of the user and leaks it to an external adversary. The external adversary uses the partial private keys of users to substitute the public keys of users. It then uses the public key, which pass verification, to successfully forge a signature that can be authenticated. The adversary can then negotiate the group key with other users to communicate with them. Through our analysis of the public key authentication section above, we know that in other related schemes the public key of the user can be easily replaced without this being discovered.

5.3 Efficiency

In this subsection, we compare the computational cost and communication cost of our AGKA scheme with others. **Table 3** uses the notation given in [27, 31] for the comparison parameters. The time complexities can be respectively written as EM $\approx 29t_m$, EA $\approx 0.12t_m$, P $\approx 87t_m$, and ME $\approx 240t_m$.

Table 3. Definitions and values of Selected operational Timings

Symbol	Definitions	Time
t_m	Cost of modular multiplication	t_m
EM	Elliptic curve point multiplication	$\approx 29t_m$
EA	Elliptic curve point addition	$\approx 0.12t_m$
P	Bilinear pairing	$\approx 87t_m$
ME	Modular exponentiation	$\approx 240t_m$

Table 4. Performance Comparison of Schemes

Scheme	Computation Cost		
	Operations	Time	Round
Hafizul [27]	(3n-2)EM + (n-1)P	(174n-145)t_m	1
Kumar [37]	9EM + 8EA	261.96 t_m	2
Shan [38]	(3n+13)EM + (3n+20)EA	87.36n+379.4t_m	2
Semal [39]	(3n-3)EM + (n-1)EA + (n-1)P+ (n-1)ME	412.12(n-1) t_m	2
Teng [40]	(3n-3)EM + (n-1)P + (n-1) EA + (n-1)ME	412.12(n-1) t_m	2
Ours	(5n-4)EM + (7n-7)EA	(145.36n-116.36)t_m	1

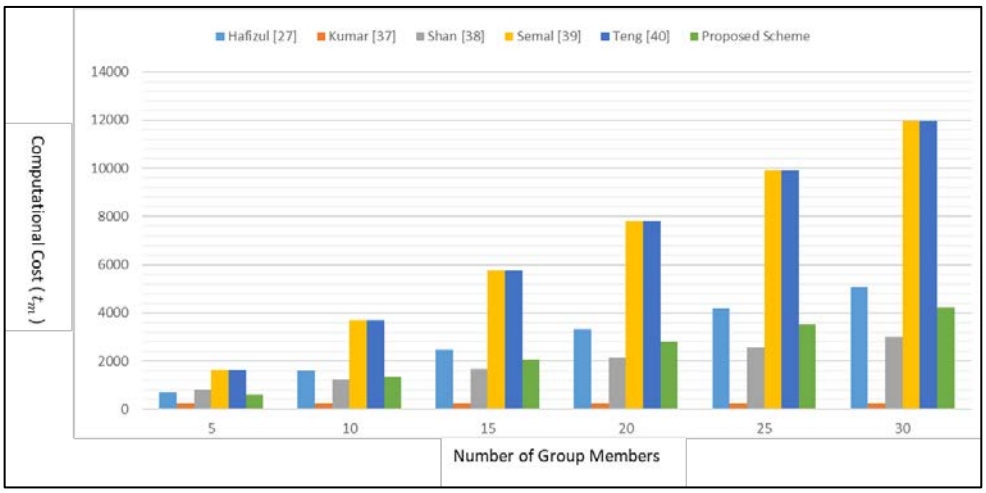

Fig. 6. Computational Cost for Various Group Sizes.

Consulting **Table 4**, when the number of group users is n, the computational complexities of the related schemes are as follows: For the scheme of Teng et al., the complexity is (3n − 3)EM + (n − 1)P + (n − 1)EA + (n − 1)ME ≈ ME 412.12(n − 1)t_m and there are two communication rounds. In the scheme of Kumar et al., the computational complexity is 9EM + 8EA ≈ 261.96t_m and there are two communication rounds. In the scheme of Hafizul et al., the computational complexity is (3n − 2)EM + (n − 1)P ≈ (174n − 145)t_m and there is one communication round. In the scheme of Shan et al., the computational complexity is (3n + 13) EM + (3n + 20)EA ≈ 87.36n + 379.4t_m and there are two communication rounds. In the scheme of Semal et al., the computational complexity is (3n − 3)EM + (n − 1)EA + (n − 1)P + (n − 1)ME ≈ 412.12(n − 1)t_m and there are two communication rounds. In our scheme, the computational complexity is (5n − 4)EM + (7n − 7)EA ≈ (145.36n − 116.36) t_m and there is one communication round. As shown in **Table 4** and **Fig. 6**, the total of computational complexity of our protocol is higher than [37] and [38] but the number of communication rounds is fewer than them.

6. Conclusion

AGKA is a key technology to make sure the secure communication among group members. An AGKA based CL-PKC can solve the problems include storage of keys, certificates, and management overhead in PKI-AGKA schemes, and the key escrow problems in ID-AGKA

schemes. Therefore, in our work, we design a one-round AGKA protocol using a CL-PKC system which is secure and efficient and can prevent impersonation attacks.

Our scheme also provides mutual authentication by verifying signatures to prevent man-in-the-middle attacks and allows public key authentication to resist PKRA associated with CL-PKC. Furthermore, our CL-AGKA scheme is resistant to attack models by $\mathcal{A}_{\mathrm{I}}$, $\mathcal{A}_{\mathrm{II}}$, and $\mathcal{A}_{\mathrm{III}}$. We have given a security analysis for these adversaries in this protocol. And the security of our protocol is based on the hardness assumptions of the ECDLP, CDHP, and OWHF.

We propose the computational complexity of our scheme by using ECC multiplication operations, which are more efficient than pairing operations, and our plan reduces the time of network communication during key agreement by reducing the number of communication rounds to one. Therefore, this proposed scheme is more secure and efficient than other existing schemes. It has good applicability to group communication environments with limited bandwidth, storage space, and computing power.

Currently, this scheme applies to a static group environment. But nowadays More and more group communication environments need to satisfy members' joining or exiting. In those dynamic environments, AGKA protocols must can update group key when group members join and leave to improve forward and backward security of communication. In the next stage, we would like to enlarge the present work to make it suitable to the dynamic group communication environment, such as VANET, distributed cloud communication environment, etc.

Acknowledgement

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2022R1A2B5B01002490) and the BK21 FOUR (Fostering Outstanding Universities for Research) (no. 5199990914048) and the Soonchunhyang University Research Fund.

References

[1] E. Bresson, O. Chevassut, A. Essiari, et al, "Mutual authentication and group key agreement for low-power mobile devices," *Computer Communications*, vol. 27, no. 17, pp. 1730-1737, 2004. Article (CrossRef Link)

[2] M. J. Beller, Y. Yacobi, "Fully-fledged two-way public key authentication and key agreement for low-cost terminals," *Electronics Letters*, vol. 29, no. 11, pp. 999-1001, 1993. Article (CrossRef Link)

[3] C. Zemao, Z. Junge, H. Biyi, "Optimizing PKI for 3GPP authentication and key agreement," in *Proc. of Fourth International Conference on Multimedia Information Networking and Security*, IEEE, pp. 79-82, 2012. Article (CrossRef Link)

[4] V. S. Naresh, N. V. E. S. Murthy, "A new two-round dynamic authenticated contributory group key agreement protocol using elliptic curve Diffie–Hellman with privacy preserving public key infrastructure," *Sadhana*, vol. 40, no. 7, pp. 2143-2161, 2015. Article (CrossRef Link)

[5] K. Y. Choi, J. Y. Hwang, D. H. Lee, "Efficient ID-based group key agreement with bilinear maps," in *Proc. of International Workshop on Public Key Cryptography*, Springer, Berlin, Heidelberg, pp. 130-144, 2004. Article (CrossRef Link)

[6] J. Zheng, C. Yang, J. Xue, et al, "A dynamic id-based authenticated group key agreement protocol," in *Proc. of the 4th National Conference on Electrical, Electronics and Computer Engineering*, 2015. Article (CrossRef Link)

[7] L. C. Li, Y. P. Tsai, R. S. Liu, S. Nathani, B. P. Tripathi, S. Khatoon, "A Dynamic ID Based Authenticated Group Key Agreement Protocol from Pairing," *Int. J. Netw. Secure*, vol. 21, no. 4, pp. 582-591, 2019. Article (CrossRef Link)
[8] S. Heo, Z. Kim, K. Kim, "Certificateless authenticated group key agreement protocol for dynamic groups," in *Proc. of IEEE GLOBECOM 2007-IEEE global telecommunications conference*, IEEE, pp. 464-468, 2007. Article (CrossRef Link)
[9] E. J. Lee, S. E. Lee, K. Y. Yoo, "A certificateless authenticated group key agreement protocol providing forward secrecy," in *Proc. of International Symposium on Ubiquitous Multimedia Computing*, IEEE, pp. 124-129, 2008. Article (CrossRef Link)
[10] M. Geng, F. Zhang, M. Gao, "A secure certificateless authenticated group key agreement protocol," in *Proc. of International conference on multimedia information networking and security*, IEEE, pp. 1: 342-346, 2009. Article (CrossRef Link)
[11] E. Bresson, D. Catalano, "Constant round authenticated group key agreement via distributed computation," in *Proc. of International Workshop on Public Key Cryptography*, Springer, Berlin, Heidelberg, pp. 115-129, 2004. Article (CrossRef Link)
[12] L. Wang, Y. Tian, D. Zhang, et al, "Constant-round authenticated and dynamic group key agreement protocol for D2D group communications," *Information Sciences*, vol. 503, pp. 61-71, 2019. Article (CrossRef Link)
[13] W. Diffie, M. Hellman, "New directions in cryptography," *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644-654, 1976. Article (CrossRef Link)
[14] C. Adams, S. Lloyd, "Understanding public-key infrastructure: concepts, standards, and deployment considerations," *Sams Publishing*, 1999.
[15] A. Shamir, "Identity-Based Cryptography systems and Signature Schemes," *Advances in Cryptology*, pp. 47-53, 1984. Article (CrossRef Link)
[16] S. S. Al-Riyami, K. G. Paterson, "Certificateless Public Key Cryptography," in *Proc. of International conference on the theory and application of cryptology and information security*, pp. 452-473, 2003. Article (CrossRef Link)
[17] J. Yeh, S. Sridhar, G. G. Dagher, et al, "A certificateless one-way group key agreement protocol for end-to-end email encryption," in *Proc. of IEEE 23rd Pacific Rim International Symposium on Dependable Computing (PRDC)*, IEEE, pp. 34-43, 2018. Article (CrossRef Link)
[18] S. Mandal, S. Mohanty, B. Majhi, "CL-AGKA: Certificateless authenticated group key agreement protocol for mobile networks," *Wireless Networks*, vol. 26, no. 4, pp. 3011-3031, 2020. Article (CrossRef Link)
[19] I. A. Kamil, S. O. Ogundoyin, "A lightweight certificateless authentication scheme and group key agreement with dynamic updating mechanism for LTE-V-based internet of vehicles in smart cities," *Journal of Information Security and Applications*, vol. 63, pp. 102994, 2021. Article (CrossRef Link)
[20] L. Zhang, Q. Wu, B. Qin, et al, "Certificateless and identity-based authenticated asymmetric group key agreement," *International Journal of Information Security*, vol.16, no. 5, pp. 559-576, 2017. Article (CrossRef Link)
[21] S. Bala, G. Sharma, A. K. Verma, "Impersonation attack on CertificateLess key agreement protocol," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 27, no. 2, pp.108-120, 2018. Article (CrossRef Link)
[22] H. Xiong, Y. Wu, Z. Lu, "A survey of group key agreement protocols with Constant-Rounds," *ACM Computing Surveys (CSUR)*, vol. 52, no. 3, pp. 1-32, 2019. Article (CrossRef Link)
[23] S. Heo, Z. Kim, K. Kim, "Certificateless authenticated group key agreement protocol for dynamic groups," in *Proc. of IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference*, IEEE, pp. 464-468, 2007. Article (CrossRef Link)
[24] A. Kumar, S. Tripathi, "Ternary tree based group key agreement protocol over elliptic curve for dynamic group," *International Journal of Computer Applications*, vol. 86, no. 7, 2014. Article (CrossRef Link)

[25] A. Rawat, M. Deshmukh, "Tree and elliptic curve based efficient and secure group key agreement protocol," *Journal of Information Security and Applications*, vol. 55, pp. 102599, 2020. Article (CrossRef Link)
[26] G. Xiaozhuo, X. Taizhong, Weihua Z, et al, "A pairing-free certificateless authenticated group key agreement protocol," in *Proc. of IEEE Intl Conf on High Performance Computing and Communications, 2014 IEEE 6th Intl Symp on Cyberspace Safety and Security, 2014 IEEE 11th Intl Conf on Embedded Software and Syst (HPCC, CSS, ICESS)*, pp. 510-513, 2014. Article (CrossRef Link)
[27] S. K. Islam, A. Singh, "Provably secure one-round certificateless authenticated group key agreement protocol for secure communications," *Wireless Personal Communications*, vol. 85, no.3, pp. 879-898, 2015. Article (CrossRef Link)
[28] M. Burmester, Y. Desmedt, "A secure and efficient conference key distribution system," in *Proc. of Workshop on the Theory and Application of of Cryptographic Techniques*, Springer, Berlin, Heidelberg, pp. 275-286, 1994. Article (CrossRef Link)
[29] J. Katz, M. Yung, "Scalable protocols for authenticated group key exchange," in *Proc. of Annual international cryptology conference*, Springer, Berlin, Heidelberg, pp. 110-125, 2003. Article (CrossRef Link)
[30] J. Lopez, R. Dahab, "An overview of elliptic curve cryptography," 2000.
[31] V. Kapoor, V. S. Abraham, R. Singh, "Elliptic curve cryptography," *Ubiquity*, pp. 1-8, 2008. Article (CrossRef Link)
[32] A. Kumar, S. Tripathi, "A pairing free anonymous certificateless group key agreement protocol for dynamic group," *Wireless Personal Communications*, vol. 82, no. 2, pp. 1027-1045, 2015. Article (CrossRef Link)
[33] M. Girault, "Self-certified public keys," in *Proc. of Workshop on the Theory and Application of of Cryptographic Techniques*, Springer, Berlin, Heidelberg, pp. 490-497, 1991. Article (CrossRef Link)
[34] M. Luo, J. Wu, X. Li, "Cross-domain certificateless authenticated group key agreement protocol for 5G network slicings," *Telecommunication Systems*, 74(4), 437-449, 2020. Article (CrossRef Link)
[35] L. Zhang, Q. Wu, B. Qin, et al, "Certificateless and identity-based authenticated asymmetric group key agreement," *International Journal of Information Security*, vol. 16, no. 5, pp. 559-576, 2017. Article (CrossRef Link)
[36] N. Q. Viet, W. Ogata, "Certificateless aggregate signature schemes with improved security," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 98, no. 1, pp. 92-99, 2015. Article (CrossRef Link)
[37] A. Kumar, S. Tripathi, P. Jaiswal, "A pairing free certificateless group key agreement protocol with constant round," *Advanced Computing, Networking and Informatics-Volume 2*, Springer, Cham, pp. 341-349, 2014. Article (CrossRef Link)
[38] S. Chun, H. U. Kangwen, X. U. E. Jingfeng, et al, "Improved pairing-free constant round certificateless authenticated group key agreement protocol," *Journal of Tsinghua University (Science and Technology)*, vol. 57, no. 6, pp. 580-585, 2017. Article (CrossRef Link)
[39] B. Semal, K. Markantonakis, R. N. Akram, "A certificateless group authenticated key agreement protocol for secure communication in untrusted UAV networks," in *Proc. of IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, IEEE, pp. 1-8, 2018. Article (CrossRef Link)
[40] Teng, Jikai, Wu. Chuankun, "A provable authenticated certificateless group key agreement with Constant-Rounds," *Journal of Communications and Networks*, 14.1, pp. 104-110, 2012. Article (CrossRef Link)

Huimin Ren is affiliate with Dept. of Software Convergence, Soonchunhyang University, South Korea. She received the B.S. degree in Software Engineering from Soonchunhyang University, Korea, in 2020 and the B.S. degree in Software Engineering from Anhui University of Chinese Medicine, China, in 2020. She is currently pursuing her M.S. degree at Soonchunhyang University, South Korea. Her research interests include cryptographic protocols, key management and authenticated group key agreement.

Suhyun Kim is currently a Manager in the ICT Industry Strategy at National IT Industry Promotion Agency (NIPA), Jincheon, South Korea. Recently, he has been actively working in the area of NPU (Neural Processing Unit), in particular with respect to government policies. His research interest includes cryptographic protocol, IoT security, AI security. He received Ph.D. degree in Computer Science from Soonchunhyang University (SCH), Asan, South Korea in 2016.He received M.S. degree in Computer Science from the Soonchunhyang University (SCH), Asan, South Korea in 2012 and B.S. degree in Computer Science from Soonchunhyang University (SCH), Asan, South Korea in 2010.

Daehee Seo (Member, IEEE) received the B.S. degree in electronic and electrical engineering from Dongshin University, Naju, South Korea, in February 2001, and the M.S. degree in computer science and engineering and the Ph.D. degree in computer science from Soonchunhyang University (SCH), Asan, South Korea, in February 2003 and February 2006, respectively. He is currently an Assistant Professor with the Faculty of Artificial Intelligence and Data Engineering, SangMyung University (SMU), Seoul, South Korea.

Imyeong Lee (Member, IEEE) received the B.S. degree in electronic engineering from Hongik University, Seoul, in 1981, and the M.S. and Ph.D. degrees in information and communication engineering from Osaka University, Osaka, Japan, in 1986 and 1989, respectively. He is currently a Professor with the Department of Computer Software Engineering, Soonchunhyang University (SCH), Asan, South Korea. His research interests include information security, cryptographic protocol, information theory, and data communication