Lightweight Certificateless Authenticated Key Agreement Protocol

Mwitende Gervais

Pivot Access Ltd, Kigali, Rwanda

Abstract. Data security and privacy are important to prevent the reveal, modification and unauthorized usage of sensitive information. The introduction of using critical power devices for internet of things (IoTs), e-commerce, e-payment, and wireless sensor networks (WSNs) has brought a new challenge of security due to the low computation capability of sensors. Therefore, the lightweight authenticated key agreement protocols are important to protect their security and privacy. Several researches have been published about authenticated key agreement. However, there is a need of lightweight schemes that can fit with critical capability devices. Addition to that, a malicious key generation center (KGC) can become a threat to watch other users, i.e impersonate user by causing the key escrow problem. Therefore, we propose a lightweight certificateless Authenticated Key Agreement (AKA) based on the computation Diffie-Hellman problem (CDHP). The proposed protocol maintains the characteristics of certificateless public key cryptography. The protocol is split into two combined phases. In the first phase, our protocol establishes a session key between users (sender and receiver). In the second phase, we use a lightweight proxy blind signature based on elliptic curve discrete logarithm problem (ECDLP). The used proxy signature has small computation costs, and can fit for small devices such sensors and protects against un-authentication and un-authorization on decentralized system. Compared to the existing AKA schemes, our scheme has small computation costs. The protocol achieves the well known security features compared to the related protocols.

Keywords: Cerificateless AKA · distinguishability· Session key · proxy blind signature · forward secrecy · decentralized.

1 Introduction

AKA protocols are one of the most important primitive that are useful for information security and privacy. AKA protocol involves the participation of two or more parties that share their public parameters so that they can compute a secret key for their secure communication over an open network. The parties in AKA

David C. Wyld et al. (Eds): SIPR, NCWC, BIBC, SOFEA, DSML, ARIA, NLP, CSEN -2023
pp. 371-390, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.131928

can authenticate each other and encrypt messages in a way that only a computed session key can decrypt them. AKA protocol was introduced to prevent the passive and active attacks [1]. The implementation of AKA protocols can be realized by deploying a public-key infrastructure (PKI) or identity-based (ID-based) cryptography, which was proven to be difficult and vulnerable because of the PKI-based protocols suffer from heavy certificate management burden while ID-based cryptographic systems require all parties to trust a KGC. Our paper investigate AKA schemes and design a lightweight certificateless AKA which can be applied in different areas of technology such as on wireless sensor networks (WSN), wireless body area networks, as well as on other IoTs systems. The first introduced certificateless AKA without trust of third party was proved informally in [2]. Since then, alot of certificateless protocols have been designed. The formal proof of certificateless AKA was presented in [3], and has attracted many researches. Numerous certificateless AKA schemes using pairings have been proposed. However, the computation cost of a pairing is very higher than scalar multiplication over elliptic curve group. Therefore, certificateless AKA without pairing CLAKA protocols would be more useful in terms of efficiency. Recently, several lightweigh AKA protocols have been designed. Authors in [4] proposed an authentication and key agreement protocol to be used for WSNs. It established a session key between sensor node and management server. Their protocol can achieves important security properties for IoTs such as forward secrecy, known session key prevention, and key control. The KGC computes private keys for both entities. Authors in [5] proposed a certificateless AKA for WBAN. Their protocol achieves differents security properties and it is high in computation costs due to the bilinear pairings involved in their scheme. Gervais et al.[6] proposed a CLAKA protocol for healthcare based on decentralized system. Their protocol achieves the well known certificateless security properties and uses security mediated signature. Authors in [7] proposed a cloud-aided lightweight certificateless authentication protocol with anonymity for WBANs. Their protocol consists of a three layer structure, provides anonymity, and its cost for equipment could be high. Li et al. [8] designed an enhanced authentication protocol for body sensors. It provides mutual authentication and session key. The protocol is certificateless, resistant to offline-password guessing attack and was proved in BAN logic. Jiang et al. [9] designed a pairing-based anonymous authentication scheme for body sensors. The protocol establishes a session key between client and application server. It is a lightweight and it is proved under CDH assumption. An anonymous AKA fog computing for healthcare system was proposed in [10]. The protocol requires a password for user registration and fog revocation while establishing the session key between IoT devices, the fog devices and cloud. Authors [11] proposed an authentication scheme for WBAN. Their protocol offers some known security features such as anonymity and unlinkability. A mobile client and application server compute a shared key agreement. Authors of [12] proposed an authentication and key agreement protocol. Their protocol achieves some important security features such as anonymity and untraceability.

1.1 Motivation and Contribution

It is important to protect sensitive information transmitted and processed over small devices. A malicious can weaken a vulnerable medical system and gains access to the sensitive without authorization. Therefore, we set up a secure lightweight authenticated key agreement between communicating nodes/users. Additional to that, the service provider and storage should avoid security risks of single point of failure and management. Since the body sensors for healthcare are critical devices, the designed protocols require to be lightweight due to run on medical sensors. Hence, in this paper, we setup a lightweight certificateless AKA protocol to secure transmission in a decentralized system. The protocol prevents important security features for AKA and proxy blind signature while a decentralized architecture improves the security by avoiding the system administrator ownership of system. Our work contributes briefly in the following ways:

- A Certificateless AKA for decentralized-based system is designed to provide forward secrecy. A session key is established between user A who is the owner of the sensitive data and the user B who is the receiver of the data at the same time B acts as the decentralized node.
- A lightweight proxy blind signature based on ECDLP is used to provide some important security features such as authentication and verification of the data origin among the decentralized nodes. The proxy blinded signature provides distinguishability and unlinkability. It is an efficient signature suitable for resource constrained devices.
- The protocol has less computation overhead due to few point multiplication and hash function used during the protocol design.
- The formal and informal analysis prove that our protocol is secure in random oracle model under CDH and ECDL assumptions.

1.2 Organization

In section 2, the preliminaries are discussed. In section 3, the modeling of certificateless AKA protocol is presented. In section 4, the proposed protocol is designed. Section 5, the analysis of the proposed protocol is discussed. Section 6, we conclude our paper.

2 Preliminaries

2.1 Notations

The common notation used in this paper are listed as follows in Table 1.

Table 1. Notations and description

Notation	Description
P_{pub}	The public key of KGC
s	Master secret key
i	i^{th} Users
Q_i	The partial private key of user i
ID_i	The user's identity
u_i	The secret value of user i
X_i	The public key of user i
d_i	The private key of user i
K_{ij}, K_{ji}	The session key of user i and user j
a, b, y, and r_i	random numbers
p	A large prime number
$\mathbb{F}_p$	Prime field
E	An elliptic curve E over a prime field $\mathbb{F}_p$
G	Additive group
P	Generator of G
H, H_1, H_2	Hash functions
p, q	Prime numbers
x_B	The receiver's private key
P_B	The receiver's public key
A	User A considered as a sending user
B	User B considered as a requesting node
V	User V considered as a verifier node
Pr	User Pr considered as a proxy signing node
x_s	The signer's private key
y_s	The signer's public key
x_p	The proxy's private key
y_p	The proxy's public key
Q_s	The proxy's public key of the signer $Q_s = xs.\mathcal{G}$
$\mathcal{G}$	A finite point with order n in $E/\mathbb{F}_p$

2.2 Background of Certifcateless AKA

Authors in [2] were the first to propose a certificateless AKA protocol. Since then, several researches about certificateless AKA protocols have been conducted based on pairings and without pairings. Most of the AKA protocols have been designed based on PKI and ID-based cryptography. The PKI certificate management is difficult in terms of computation and storage and ID-based cryptography trusts KGC which can launch an active attack to eavesdrop the communication between users [13]. Therefore, we are required to solve the problem of PKI and avoid the trust on KGC in ID-based AKA. Much efforts have been made to alleviate the previous limitations in authentication and key agreement protocols. The approach to solve the PKI and ID-based AKA problems has been the implementation of certificateless cryptography [14]. During the design of certificateless AKA protocols, the KGC creates a half of the private keys for communicating parties. Users can generate their own private keys using the random selected secret values. Therefore, the key escrow problem is prevented in certificateless cryptography. Hence, certificateless cryptography does not need additional certificate to show the ownership of a public key [15].

2.3 The Finite Field

Let set p as a prime number. The finite field F_p is comprised of the set of integers $\{0, 1, 2, ..., p-1\}$ with the following arithmetic operations:

- Addition: If $a, b \in F_p$, then $a + b = r$ where r is the reminder when $a + b$ is divided by p and $0 \leq r \leq p - 1$. This is known as addition modulo p

- Multiplication: If $a, b \in F_p$, then $a.b = s.$ where s is the reminder when $a.b$ is divided by p and $0 \leq r \leq p-1$. This is known as multiplication modulo p
- Inversion: If a non-zero element in F_p, the inverse of a modulo p, denoted a^{-1}, is the unique integer $c \in F_p$ for which $a.c = 1$.

2.4 Elliptic curve defnition

For $p \geq 3$ let $E/\mathbb{F}^p$ be elliptic curve E over a finite field $\mathbb{F}_p$, defined by the equation:

$$y^2 = (x^3 + ax + b), a, b \in \mathbb{F}_p \tag{1}$$

the discriminant

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

The points on $E/\mathbb{F}_p$, and the point at infinity make a group of points G.

$$G = \{(x, y) : x, y \in \mathbb{F}_p, E(x, y) = 0\} \cup \{O\}. \tag{3}$$

Assume q to be the order of G. The scalar multiplication over $E/\mathbb{F}_p$ is defined as

$$tP = P + P + P + \cdots + P\,(t\,\text{times}). \tag{4}$$

The detailed mathematical operations related to elliptic curve can be found in [16]. The following defined problems over G are assumed to be intractable within polynomial time.

2.5 Hard assumptions

Definition 1. *DLP assumption: Given (P, aP), for an unknown selected value $a \in \mathbb{Z}_q^*$ and P generator of $\mathbb{G}$, compute aP. The DLP states that it is intractable to determine the value a for any probabilistic polynomial-time*

Definition 2. *CDH assumption: Given (P, aP, bP), for unknown $a, b \in \mathbb{Z}_q^*$ and P generator of $\mathbb{G}$, compute abP. The CDH hard assumption states that for any probabilistic polynomial-time, it is intractable to solve the CDH problem.*

2.6 Algorithms for Certifcateless AKA

We achieve a Certificateless AKA protocol following six algorithms.

- **Setup**: A KGC takes $\jmath$ as input security parameter, output a master key and system parameter list *params*.
- **Partial-Private-Key-Extract**: A KGC takes as input *params*, a master key, and a user identity ID_i to return a user partial private key Q_i.
- **Set-Secret-Value**: The algorithm takes as input *params* and a user ID_i to return user's secret value u_i.
- **Set-Private-Key**: The algorithm takes as inputs *params*, ID_i, a partial private key Q_i, and a secret value u_i to return a private key d_i for the user.

- **Set-Public-Key**: This algorithm takes as inputs *params*, user ID_i, and the secret value u_i to return a public key X_i for the user.
- **Key-Agreement**: It is a polynomial participative algorithm for both users. It takes as inputs *params* for users A and B, with (d_A, ID_A, X_A) for user A, and (d_B, ID_B, X_B) for user B; where d_A and d_B are private keys for users A and B; ID_A and ID_B are identities for users A and B. The X_A and X_B are set to be public key of users A and B. Finally both users compute a session key $K_{AB} = K_{BA} = K$.

3 Model of the proposed protocol

In this section, we present the security model and system model for the proposed certificateless AKA.

3.1 Security modeling of the proposed protocol

The certificateless AKA protocol requires to be resistant to the two types of attacks said Type I and Type II adversaries as described in [17].

- Type I Adversary A_1: The A_1 does not have access to the master secret key, but can replace public key of any party with a value of his choice.
- Type II Adversary A_2: The A_2 has access to the master secret key but can not replace public key of any party.

3.2 System model

The system model is composed of two phases with the following entities: KGC, user A and B for the first phase. The KGC registers and computes partial private keys for both users. Upon receiving partial private keys, both users computes a session key to authenticate themselves and secure data transmission. Note that the key is established in every session when the two entities want to communicate by preventing the known key share problem. In the second phase, we have A, proxy Pr, Verifier V, and other decentralized nodes. Two nodes of the second phase participate in proxy blind signature creation. Figure 1 illustrates the proposed system model architecture as explained in the following steps.

- The KGC is dedicated to register the users A and B. Also, it generates system parameters list. KGC cannot know about the private keys of users A and B.
- The user A can communicate and transmits data to user B via wireless network. User A computes its private key and establishes a session key with user B. The session key will encrypt data transmitted from A to the receiver B.
- The receiver B should be registered with KGC and get partial-private key and system parameters. It also establishes a session key with A. The session key is used to encrypt and decrypt data that is sent by A over an open network.
- User B and proxy Pr and the verifier V participate in the establishment of proxy blind signature for decentralized system

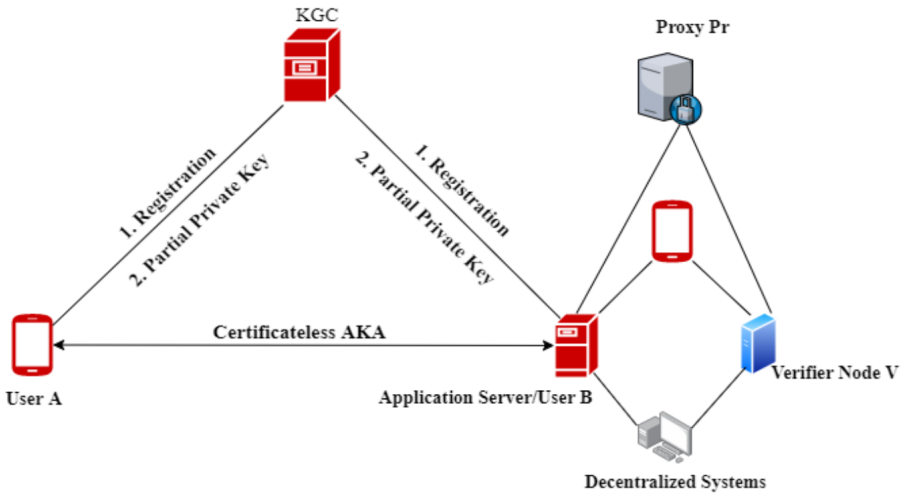

Fig. 1. System model

3.3 Formal security model

The formal security model follow the model discussed in [18]. It is modeled as the game between challenger $\mathcal{C}$ and adversary $A \in \{A_1, A_2\}$. The adversary monitors all interactions between two parties. Every party possesses an identity ID_i. The characteristics of A represented by the number of oracles kept by $\mathcal{C}$. Assume that an oracle $\phi_{i,j}^r$ represents r^{th} instance of party i and his counterpart j in a session. The game starts when $\mathcal{C}$ sets up algorithm with security parameter $\jmath$ to return master secret and system *pars*. If A is Type I adversary A_1, $\mathcal{C}$ transmits *params* to A and maintains master key secret; else A is Type II adversary A_2, $\mathcal{C}$ issues *params* and master key to A. Adversary A is a probabilistic polynomial time turing machine. All communications go through A. Parties answer to the queries from A and do not interact between them. A acts as benign, i.e A is deterministic and prefer to choosing two oracles $\phi_{i,j}^n$ and $\phi_{j,i}^l$ and takes each message from one oracle to another. in addition, A can ask for the following queries, including one **Test** query in the following way:

- **Create**(ID_i): This query permits A to request $\mathcal{C}$ to create a new party i whose identity is ID_i. Upon receiving such query, $\mathcal{C}$ creates private and public keys for i.
- **Public-Key**(ID_i): A can may ask for public key of a party i whose identity is ID_i. To answer, $\mathcal{C}$ replays with the public key X_i of party i.
- **Partial-Private-Key**(ID_i): A may ask for partial private key of party i whose identity is ID_i. To respond, $\mathcal{C}$ replays with partial private key Q_i of party i.
- **Corrupt**(ID_i): A may ask for private key of party i whose identity is ID_i. To respond, $\mathcal{C}$ replays with the private key d_i of party i.

- **Public-Key-Replacement**$(ID_i, X_i^{'})$: For a party i whose identity is ID_i; A may select another public key $X^{'}$ and set $X^{'}$ as the public key. $\mathcal{C}$ record this change to be used in the future.
- **Send**$(\phi_{i,j}^n, \mu)$: A may select and issues a message μ to an oracle $\Phi_{i,j}^n$, by which, a party i assumes to be sent from party j. A can also create a particular **Send** query with $\mu \neq \alpha$ to an oracle $\Phi_{i,j}^n$, which tells i to start a protocol runs with j. It is called an initiator oracle when the first message it has obtained is α. Otherwise, it is called a responder oracle.
- **Reveal**$(\phi_{i,j}^n)$: A may request a special oracle to reveal the session key, if any, it is currently holding to A.
- **Test**$(\phi_{i,j}^n)$: At certain level, A can choose one of the oracles, for example $\Phi_{I,J}^T$ to request for one **Test** query. Such oracle should be *fresh*. To answer the query, the oracle guesses a coin $b \in \{0, 1\}$, and outputs the session key held by $\Phi_{I,J}^T$ if $b = 0$, or a random sample from the distribution of session key if $b = 1$.

An oracle $(\phi_{i,j}^n)$ can be set to one of the three states

- *Accepted*: An oracle is in *Accepted* state if it has accepted the request to create a session key.
- *Rejected*: An oracle is in *Rejected* state if it has rejected the request to create a session key.
- *State**: If none of the previous states decision has been taken.
- *Opened*: If an oracle has answered the **Reveal** query.

Definition 3. *A matching conversation: Two oracles* $(\phi_{i,j}^n)$ *and* $(\phi_{j,i}^l)$ *have a matching conversation if they have identical session key.*

Definition 4. *Fresh Oracle: An oracle* $(\phi_{i,j}^n)$ *is fresh if it is in the accepted state; or it is not in the opened state; or party* $j \neq i$ *is not corrupted; or* $(\phi_{j,i}^l)$ *does not exist in opened state to have the matching conversation with* $(\phi_{i,j}^n)$*; or if A is Type I and has not requested the private key of party j and if A is Type II and has not replaced the public key of party j*

The fresh oracle definition can allow party i to be corrupted so that it is used to solve the key compromise impersonation attack.

After a **Test** query, A may go on to query the oracles except make **Reveal** query to test oracle $\Phi_{I,J}^T$, or to $\Phi_{J,I}^l$ who has a matched conversation with $\Phi_{I,J}^T$, and it can not corrupt the user J. In addition, if A is Type I, A can not ask for partial private key of the participant J; and if A is a Type II adversary, J cannot replace the public key of the user J. At the end of the game, A must output a guess bit $b^{'}$. A wins if and only if $b^{'} = b$. A's advantage to win the game, is defined as:

$$A^J = \left| \Pr[b^{'} - b] - \frac{1}{2} \right| \tag{5}$$

Definition 5. *A certificateless AKA protocol is secured if:*

- *In the presence of a benign adversary on* $\Phi_{i,j}^n$ *and* $\Phi_{j,i}^l$*, both oracles always agree on the same session key, and this key is distributed uniformly at random.*
- *For an adversary A, advantage* A^J *of winning game is negligible.*

4 The proposed protocol

The proposed protocol consists of a new lightweight certificateless AKA for WBAN sensors and other IoT environments.

4.1 The proposed Certifcateless AKA

In this section, a certificate AKA scheme is proposed. It consists of six polynomial time algorithms.

- **Setup**: This algorithm takes security parameter $\jmath$ as its input and returns system parameters and master key. KGC performs the following operations.
  1. Given a security parameter $\jmath$, KGC selects an additive group G of prime order q and P is a generator of the group.
  2. Selects a random master key $s \in \mathbb{Z}_q^*$ and calculates $P_{pub} = sP$ as master public key.
  3. Selects hash functions $H_1 : \{0,1\}^* \times G \longrightarrow \mathbb{Z}_q^*$ and $H_2 : \{0,1\}^* \times \{0,1\}^* \times G \times G \times G \times G \longrightarrow \{0,1\}^{\jmath}$
  4. KGC publishes system params $(\mathbb{F}_p, E/\mathbb{F}_p, G, q, P, P_{pub}, H_1, H_2)$ and keeps s secret.

- **Partial-Private-Key**: KGC takes as inputs *params*, the master key s and user identity ID_i and returns partial private key of users as follows
  1. KGC selects a random number $e_i \in \mathbb{Z}_q^*$ computes $R_i = e_iP$, $h_i = H_1(ID_i, R_i)$.
  2. KGC computes KGC computes $s_i = (e_i + sh_i) \bmod q$.
  3. KGC sets $Q_i = (s_i, R_i)$ as user's partial private key.
  4. User i verifies whether the partial private key is valid by computing the equation $s_iP = R_i + H_1(ID_i, R_i)P_{pub}$.

- **Set-Secret-Value**: This algorithm takes *params* and user's ID, selects randomly $u_i \in \mathbb{Z}_q^*$. u_i is sets as secret value.
- **Set-Private-Key**: The algorithm takes as inputs *params*, partial private key Q_i, user's ID_i, and secret value u_i and returns user's private key $d_i = (u_i, Q_i)$.
- **Set-Public-Key**: The algorithm takes as input *params*, user ID_i and user's secret value u_i to return user's public key $X_i = u_iP$.
- **Key-Agreement**: Assuming that user A can establish an authenticated key agreement with user B. Lets users A and B establish a certificateless AKA, and one is the sender another one receiver. The sender A with identity ID_A possesses the private key $d_A = (u_A, Q_A)$ and the public key $X_A = u_AP$. The receiver B with identity ID_B possesses the private key $d_B = (u_B, Q_B)$ and the public key $X_A = u_AP$. The sender A and receiver B compute the protocol as follows:
  1. User A selects $a \in \mathbb{Z}_q^*$, computes $T_A = aP$ and sends a message (ID_A, T_A) to B.

2. B selects $b \in \mathbb{Z}_q^*$, computes $T_B = bP$ and sends a message (ID_B, T_B) to A.

Both A and B can compute the secrets as the following: A computes

$$K_A = (R_B + h_B P_{pub}) + s_A P + aX_B + u_A T_B + aT_B \tag{6}$$

B computes

$$K_B = (R_A + h_A P_{pub}) + s_B P + bX_A + u_B T_A + bT_A \tag{7}$$

Correctness

$$\begin{aligned} K_A &= (R_B + h_B P_{pub}) + s_A P + aX_B + u_A T_B + aT_B \\ &= (e_B + sh_B)P + (e_A + sh_A)P + au_B P + u_A T_N + abP \\ &= s_B P + (e_A + sh_A)P + au_B P + bu_A P + baP \\ &= s_B P + (e_A + sh_A)P + bXA + u_B T_A + bT_A \\ &= (R_A + h_A P_{pub}) + s_B P + bX_A + u_B T_A + bT_A \\ &= K_B \\ &= K \end{aligned}$$

The established session key $SK = H_2(ID_A, ID_B, T_A, T_B, K)$.

Algorithm 1 Algorithm for Certificateless AKA scheme

Input: $\{ID_i, params, Q_i, R_i, h_i, s_i\}$
Output: $SK = H_2(ID_A, ID_B, T_A, T_B, K)$

User randomly selects $u_i \in \mathbb{Z}_q^*$
Compute $d_i = (u_i, Q_i)$
Compute $X_i = u_i P$
A session key is computed as follows
A randomly select $a \in \mathbb{Z}_q^*$ and
Compute $T_A = aP$
A send (ID_A, T_A) to B
B randomly select $b \in \mathbb{Z}_q^*$ and
Compute $T_B = bP$
B send (ID_B, T_B) to A
B computes $K_B = (R_A + h_A P_{pub}) + s_B P + bX_A + u_B T_A + bT_A$
A computes $K_A = (R_B + h_B P_{pub}) + s_A P + aX_B + u_A T_B + aT_B$
if $K_B = K_A = K$ **then**
    Return a session key $SK = H_2(ID_A, ID_B, T_A, T_B, K)$
end if

4.2 User authentication and verifcation

The adopted proxy blind signature scheme satisfies the following security properties:

- *Distinguishability.* The proxy signature must be distinguishable from other digital signature.
- *Strong unforgeability.* Only the dedicated proxy signer can create the proxy blind signature for the original signer.
- *Non repudiation.* Non among The signers either origin or the proxy cannot deny their signatures against anyone.
- *Verifiability.*The proxy blind signature can be verified by everyone. After verification, the verifier can be convinced of the original signer's message is from legit node.
- *Strong undeniability.* Due to fact that the delegation information is signed by the original signer and the proxy signature are generated by the proxy signer's secret key. Both the signer can not deny their behavior.
- *Unlinkability.* When the signer is revealed, the proxy signer can not identify the association between the message and the blind signature he generated.
- *Secret key dependencies.* Proxy key or delegation pair can be computed only by the original signer's secret key.
- *Prevention of misuses.*The proxy signer cannot use the proxy secret key for purposes other than generating valid proxy signatures. In case of misuse, the responsibility of the proxy signer should be determined explicitly.

Proxy blind signature In this section, We present a proxy blind signature from [19] which is lightweight with low computation costs and provides more security features compared to the existing blind signature schemes. The protocol involves three entities including signer or requester B, which is the receiver of data from user A; the proxy signer Pr acting as proxy, the verifier node V acting as the verifying node, and other decentralized nodes. The signature is designed through the following steps

- **Proxy delegation phase**
  1. **Proxy generation**. Given the $Q_s = x_s.\mathcal{G}$ as the public key of the signer. The signer B selects a random number $k \in [1, n-1]$ and then calculates $\alpha = k\mathcal{G} = (\alpha_1, \alpha_2)$ where $u \equiv \alpha_1 mod\, n$ and α_1 is regarded as an integer element of $[1, q-1]$. Computes $\beta \equiv (x_s + ku)(mod\, n)$ and computes $Q_p = \beta\mathcal{G}$.
  2. **Proxy delivery**. The signer B sends (β, u) to the proxy Pr in a secure channel, and make Q_p public.
  3. **Proxy verification**. On the receiving the secret key pair (β, u), the proxy signer Pr verifies the correctness of the secret key pair (β, u) by computing the following equation:

$$Q_p = \beta\mathcal{G} = Q_s + u\alpha \tag{8}$$

- **Proxy signing phase**
  1. The proxy signer Pr selects a random integer $t \in [1, n-1]$, and calculates $Z = t.\mathcal{G}$ and sends it to the verifier V

2. On the receiving it, the verifier selects randomly $\omega, \gamma \in [1, n-1]$ and computes the following equations:

$$\alpha' = Z + \omega\mathcal{G} - \gamma Q_p \tag{9}$$

$$g = H(\alpha' || M) \tag{10}$$

$$g' = (g + \gamma) mod\, n \tag{11}$$

   And the verifier V sends g' to the proxy signer node Pr
3. On the receiving of g', Pr calculates the following equation

$$\beta' = (t - \beta.g') mod\, n \tag{12}$$

   and sends it to V
4. V calculates

$$\beta'' = (\beta' + \omega)\, mod\, n \tag{13}$$

   The tuples (M, β'', g) is the proxy blinded signature

- **Verification phase** The verifying node V computes the following equation

$$\delta = H((\beta''\mathcal{G} + g.Q_p)||M) \tag{14}$$

  and finally check whether the proxy blind signature holds with $\delta = g$
- **Correctness**

The computed proxy blind signature is verified because the following equation holds:

$$H((\beta''.\mathcal{G} + g.Q_p)||M) = H(\alpha'||M) \tag{15}$$

That is to verify

$$\begin{aligned}
\beta''.\mathcal{G} + g.Q_p &= \alpha' \\
&= (\beta' + \omega).\mathcal{G} + g.Q_p \\
&= \beta'\mathcal{G} + \omega\mathcal{G} + g.Q_p \\
&= (t - \beta.g')\mathcal{G} + \omega\mathcal{G} + g.Q_p \\
&= t.\mathcal{G} - g'Q_p + \omega.\mathcal{G} + g.Q_p \\
&= t.\mathcal{G} - (g + \delta).Q_p + \omega\mathcal{G} + g.Q_p \\
&= t.\mathcal{G} - g.Q_p - \delta.Q_p + \omega\mathcal{G} + g.Q_p \\
&= t.\mathcal{G} - \delta.Q_p + \omega\mathcal{G} \\
&= Z - \delta.Q_p + \omega\mathcal{G} \\
&= Z + \omega\mathcal{G} - \delta.Q_p \\
&= \alpha'
\end{aligned}$$

4.3 Data encryption

- User A sends a message M to the receiver B by encrypting it using a session key SKas follows:

$$(SK||M)$$

B obtains the encrypted message M and uses the session key SK to recover the message from A. B deletes SK.

5 Security analysis

The paper analysis the designed scheme following formal ROM security analysis, informal discussion of security properties and compares the proposed scheme with existing protocols.

5.1 Formal analysis

The security analysis of the proposed protocol relays on CDH assumption. We followed the security prove analyzed in [20]. The CDH hard problem in group G is stated. Two random oracles H_1 and H_2 follow the same idea as explained in [21] [21]. For security prove, we follow theorems and lemmas given bellow.

Theorem 1. *The proposed protocol is a secure Certificateless AKA.*

Proof: The Certificateless AKA protocol is proved to be secure against two types of adversaries. The proof of **Theorem 1** is discussed using the following Lemmas 1, 2 and 3.

Lemma 1. *. In the presence of benign adversary, two matching oracles $\Phi^n_{i,j}$ and $\Phi^l_{j,i}$ establish the same session key as if there is no adversary. The session key is distributed uniformly at random.*

Proof Suppose that i and j are two users in the protocol and A_{dv} is a benign adversary. In this case, the two oracles gets correctly identical message to the original messages from other oracle; therefore, they consent on the same session key. Since a and b were chosen randomly by users i and j, the common session key is considered as the output of hash function H_2 on a random input. Based on the properties of hash function, the session key is uniformly distributed over $\{0,1\}^J$. As it is detailed in our protocol correctness. The numbers a and b are randomly chosen, two oracles are matching, they are authorized either and the session key is consistently shared.
Thus user A computes

$$H_2(ID_A, ID_B, T_A, T_B, K_A)$$

And application server/User B computes

$$H_2(ID_A, ID_B, T_A, T_B, K_B)$$
$$K_A = K_B = K$$

Finally the matching oracles compute the session key

$$SK{=}H_2(ID_A, ID_B, T_A, T_B, K)$$

Lemma 2. *Assuming that the CDH problem is intractable, the advantage of a Type I adversary in winning game is negligible in the ROM.*

Proof. Assume that A_{dv} can make at most q_{H_2} times H_2 queries and create at most q_c parties. Advantage for A_{dv} to win the game is $A^{\jmath}_{dv}$. Therefore, the challenger can solve the CDH problem with the advantage $\frac{1}{q_c^2 q_s q_{H_2}} A^{\jmath}_{dv}$, q_s is the number of sessions each user can participate in at most.

Assuming that a Type I adversary A_{dv} can win with a non negligible advantage $A^{\jmath}_{dv}$ in polynomial time t. We demonstrate that challenger $\mathcal{C}$ can solve CDH problem with a non negligible probability. We demonstrate how challenger $\mathcal{C}$ uses A_{dv} to compute abP.

All adversary's queries now pass through $\mathcal{C}$. The game is initiated when $\mathcal{C}$ selects a and sets $P_{pub} = aP$; $\mathcal{C}$ selects at random $I, J \in [1, q_{H_1}], T \in [1, q_s], s_I, u_I, h_I \in Z_q^*$ and computes $R_I = s_I P, X_I = u_I P$, and sets P_{pub} as the system public key and sends system $params = \{G, P, P_{pub}, H_1, H_2, \jmath\}$ to A_{dv}.

- **Create**(ID_i): A challenger $\mathcal{C}$ maintains an empty list L_c initially consisting of the tuples
  (ID_i, Q_i, u_i, X_i). If $ID_i = ID_I$, challenger $\mathcal{C}$ lets partial private key, private key and public key to be $Q_i = (s_I, R_I)$, $d_i = (u_I, Q_I)$ and X_I separately. Challenger $\mathcal{C}$ also lets $H_1(ID_I, R_I) \leftarrow h_I$ where R_I, u_I, h_I are mentioned above. Otherwise, challenger $\mathcal{C}$ chooses randomly $u_i, s_i, h_i \in \mathbb{Z}_q^*$ and computes $R_i = s_i P - h_i P_{pub}$, public key is $X_i = u_i P$, then i's partial private key $Q_i = (s_i, R_i)$, private key $d_i = (u_i, Q_i)$ and public key X_i. Finally adds the tuples (ID_i, Q_i, u_i, X_i) and (ID_i, R_i, X_i, h_i) to the list L_c and L_{H_1} separately.
- H_1 *query*: Challenger $\mathcal{C}$ keeps initial empty list L_{H_1} which has tuples of the form
  (ID_i, R_i, X_i, h_i). If (ID_i, R_i, X_i) is on the list L_{H_1}, then h_i is returned. Else, challenger $\mathcal{C}$ executes the query **Create**(ID_i) and returns h_i.
- **Public-Key**(ID_i): Upon obtaining such query, challenger $\mathcal{C}$ looks for a tuple (ID_i, Q_i, u_i, X_i) in the list L_c indexed by ID_i, and outputs X_i as response.
- **Partial-Private-Key**(ID_i): Once a challenger $\mathcal{C}$ is given such query, if $ID_i = ID_I$, $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ looks for a tuple (ID_i, Q_i, u_i, X_i) in a list L_c indexed by ID_i, and outputs Q_i as response.
- **Corrupt**(ID_i): Once a challenger $\mathcal{C}$ is given such query, if $ID_i = ID_I$, $\mathcal{C}$ aborts; else, $\mathcal{C}$ looks for a tuple (ID_i, Q_i, u_i, X_i) in a list L_c indexed by ID_i, if $u_i = \perp$, challenger $\mathcal{C}$ outputs $\perp$. Else challenger $\mathcal{C}$ gives (u_i, Q_i) as response.
- **Public-Key-Replacement**$(ID_i, X_i^{'})$: If $ID_i = ID_I$, $\mathcal{C}$ aborts. Otherwise, challenger $\mathcal{C}$ looks for a tuple (ID_i, Q_i, u_i, X_i) in L_c indexed by ID_i and upgrades X_i to $X_i^{'}$ and sets $u_i = \perp$.

- **Send**$(\Phi_{i,j}^n, \mu)$: Challenger $\mathcal{C}$ keeps empty list L_s consisting of tuples of the form $(\Phi_{i,j}^n, r_{i,j}^n, \mu_{i,j}^n, \mu_{j,i}^n, X_i^n, X_j^n, SK_{i,j}^n)$, where $\mu_{j,i}^n$ is the coming message, X_j^n is the public key of the participant j received by $\Phi_{i,j}^n$, X_i^n is the current public key owned by the user i, $r_{i,j}^n$, $\mu_{i,j}^n$ are described below. Upon receiving such query, if $\mu \neq \alpha$, challenger $\mathcal{C}$ sets $\mu_{j,i}^n = \mu$; else at the end of protocol, a message will be returned. If $\Phi_{i,j}^n$ is accepted, challenger sets message to be $\mu_{j,i}^n$ and similar response from L_s is given once the query has been requested before, if not the challenger does as the following:
  1. If $n = T$, $ID_i = ID_I$, $ID_j = ID_J$, challenger $\mathcal{C}$ sets $SK_{i,j}^n = r_{i,j}^n = \perp$ sets $\mu_{i,j}^n = aP$, return $\mu_{i,j}^n$ as the answer and adds the tuple $(\Phi_{i,j}^n, r_{i,j}^n, \mu_{i,j}^n, \mu_{j,i}^n, X_i^n, X_j^n, SK_{i,j}^n)$ to the list L_s.
  2. Else, if $ID_i \neq ID_J$, selects a random $r_{i,j}^n \in Z_n^*$, computes $\mu_{i,j}^n = r_{i,j}^n P_{pub}$, returns $\mu_{i,j}^n$ as the response, sets $SK_{i,j}^n = \perp$ and adds $(\Phi_{i,j}^n, r_{i,j}^n, \mu_{i,j}^n, \mu_{j,i}^n, X_i^n, X_j^n, SK_{i,j}^n)$ to the list L_s.
  3. Else, selects a random $r_{i,j}^n \in Z_n^*$, computes $\mu_{i,j}^n = r_{i,j}^n P$, returns $\mu_{i,j}^n$ as the response, sets $SK_{i,j}^n = \perp$, and adds $(\Phi_{i,j}^n, r_{i,j}^n, \mu_{i,j}^n, \mu_{j,i}^n, X_i^n, X_j^n, SK_{i,j}^n)$ to the list L_s.
- **Reveal**$(\Phi_{i,j}^n)$: Once receive such query, challenger $\mathcal{C}$ calls L_s for a tuple $(\Phi_{i,j}^n, r_{i,j}^n, \mu_{i,j}^n, \mu_{j,i}^n, X_i^n, X_j^n, SK_{i,j}^n)$, sets $\mu_{i,j}^n = T_i$ and $\mu_{j,i}^n = T_j$ if $SK_{i,j}^n \neq \perp$, then challenger $\mathcal{C}$ returns $SK_{i,j}^n$ as the response. Otherwise, challenger $\mathcal{C}$ looks for the tuple (ID_i, Q_i, u_i, X_i) on the list L_c and does the following:
  - If $n = T$, $ID_i = ID_I$, $ID_j = ID_J$ or $(\Phi_{i,j}^n)$ is oracle which has the matched conversation with $(\Phi_{I,J}^T)$, challenger $\mathcal{C}$ aborts.
  - Else if $ID_i \neq ID_I$, there are two steps:

  1. Challenger $\mathcal{C}$ looks in the list L_{H_2} and L_c for the corresponding tuples $\left(ID_i, ID_j, T_i, T_j, X_i, X_j K_{i,j}^n, h_u\right)$ and (ID_i, Q_i, u_i, X_i), then computes $K_{i,j}^n = (R_i + h_i P_{pub}) + s_i P + r_{i,j}^n X_i + u_i T_{j,i}^n + r_{i,j}^n T_{j,i}^n$,
  2. Otherwise, randomly sample $SK_i \in \{0,1\}^J$ and return $SK_{i;j}^n$ as the answer.
- H_2*query*: Challenger $\mathcal{C}$ maintains a list L_{H_2} of the form $(ID_u^i, ID_u^j, T_u^i, T_u^j, K_u^{i,j}, h_u)$ and responds with H_2 queries $(ID_u^i, ID_u^j, T_u^i, T_u^j, K_u^{i,j})$ in the following ways:
  1. If a tuple indexed by $(ID_u^i, ID_u^j, T_u^i, T_u^j, K_u^{i,j})$ is already in L_{H_2}, challenger responds with the corresponding h_u.
  2. Else challenger $\mathcal{C}$ chooses $h_u \in \{0,1\}^J$. Challenger $\mathcal{C}$ chooses $h_u \in \{0,1\}^J$ and add the tuple $(ID_u^i, ID_u^j, T_u^i, T_u^j, K_u^{i,j}, h_u)$ to the list L_{H_2}
- **Test**$(\Phi_{i,j}^n)$: At certain level, challenger $\mathcal{C}$ will request a test query on some oracles. If challenger $\mathcal{C}$ does not choose one of the oracles $\Phi_{I,J}^T$ to request the **Test** query, then $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ only outputs a random value $b \in \{0,1\}^J$. The probability that $\mathcal{C}$ selects $\Phi_{I,J}^T$ as the **Test** oracle is $\frac{1}{q_c^2 q_s}$. For this case, challenger $\mathcal{C}$ wouldn't have made $Corrupt(\Phi_{I,J}^T)$ or $Reveal(\Phi_{I,J}^T)$ queries, and so challenger $\mathcal{C}$ would not have aborted. If challenger $\mathcal{C}$ can win a such game, then challenger $\mathcal{C}$ must have made the corresponding H_2

query of the form $(ID_T^i, ID_T^j, T_T^i, T_T^j, K_T^{i,j})$. If $\Phi_{I,J}^T$ is the initiator oracle or else $(ID_T^i, ID_T^j, T_T^i, T_T^j, K_T^{i,j})$, with overwhelming probability because H_2 is a random oracle. Thus $\mathcal{C}$ can find the corresponding item in the H_2 list with probability and $\frac{1}{q_{H_2}}$ and outputs $K_T^i - s_I aP - (R_J + h_J P_{pub} - r_{I,J}^T X_i)$ as a solution to the CDH problem. The probability that $\mathcal{C}$ solves the CDH problem is $\frac{\varepsilon}{q_c^2 q_s q_{H_2}}$.

Lemma 3. *Under the assumption that the CDH problem is intractable, the advantage of a Type II adversary A_{dv}^2 against our protocol is negligible in the ROM.*

Proof. Suppose that there is a Type II adversary A_{dv}^2 who can win the game defined in section 4, with a non-negligible advantage $A^{\jmath}$ in polynomial time t. Then, A_{dv}^2 can win the game with no-negligible probability ε. Therefore, We show how to use the ability of A^2dv to construct an algorithm $\mathcal{C}$ to solve the CDH problem. Suppose a challenger $\mathcal{C}$ is given an instance (aP, bP) of the CDH problem, and wants to compute cP with $c = ab \bmod q$. $\mathcal{C}$ first chooses $s \in G$ at random, sets sP as the system public key P_{pub}, selects the system params $\langle \mathbb{F}_p, E/\mathbb{F}_p, G, P, P_{pub}, H_1, H_2 \rangle$, sends params and master key s to A^2dv. Supposed A_{dv}^2 makes at most q_{H_i} times H_i queries and creates at most q_c participants. Let q_s be the maximum number of sessions each participant can compute. Then, $\mathcal{C}$ selects randomly $I, J \in [q_{H_1}]$, $T \in [1, q_s]$, responds to the queries as follows.

- **Create**(ID_i): $\mathcal{C}$ maintains an initially empty list L_c consisting of tuples of the form (ID_i, u_i, X_i). If $ID_i = ID_I$, $\mathcal{C}$ selects a random $r_i, h_i \in \mathbb{Z}_q^*$ and computes $R_i = r_i P$, $s_i = (e_i + h_i s) \bmod q$, public key $X_i = u_i P$ then i 's partial private key, private key and public key are $Q_i = (s_i, R_i)$, $d_i = \{\perp, Q_i\}$ and i's public key is X_i. Otherwise, $\mathcal{C}$ selects randomly $u_i, e_i, h_i \in \mathbb{Z}_n^*$ and computes $s_i = e_i + sh_i$, $R_i = e_i P$ and $X_i = u_i P$ separately. Then i's partial private key, private key and public key are $Q_i = (s_i, R_i)$, $d_i = \{u_i, Q_i\}$ and X_i. Finally, $\mathcal{C}$ adds a tuple (ID_i, R_i, h_i) and (ID_i, Q_i, u_i, X_i) to the list L_{H_1} and L_c, separately. $\mathcal{C}$ answers A^2dv's $H_1(ID_i, R_i)$, $Public - Key(ID_i)$, $Corrupt(ID_i)$, $Send(\Phi_{i,j}^n, \mu)$, $Reveal(\Phi_{i,j}^n)$, H_2 and $Test(\Phi_{I,J}^T)$ queries as it is done in **Lemma 2**. The probability that challenger $\mathcal{C}$ selects $\Phi_{I,J}^T$ as the **Test** oracle is $\frac{1}{q_c^2 q_s}$. In this case, challenger $\mathcal{C}$ would not have made $Corrupt(\Phi_{I,J}^T)$ or $Reveal(\Phi_{I,J}^T)$ queries, and so challenger $\mathcal{C}$ would not have aborted. If challenger $\mathcal{C}$ can win in such game, then challenger $\mathcal{C}$ must have made the corresponding f_2 query of the form $(ID_T^i, ID_T^j, T_T^i, T_T^j, K_T^{i,j})$. If $\Phi_{I,J}^T$ is the initiator oracle. Else $(ID_T^j, ID_T^i, T_T^j, T_T^i, K_T^{i,j})$, with overwhelming probability because H_2 is a random oracle. Thus challenger $\mathcal{C}$ can find the corresponding item in the H_2-list with the probability $\frac{1}{q_{H_2}}$ and outputs $K_T^i - s_I bP - (R_J + h_J P_{pub} - r_{I,J}^T X_j)$ as a solution to the CDH problem. The probability that $\mathcal{C}$ solves the CDH problem is $\frac{\varepsilon}{q_c^2 q_s q_{H_2}}$.

Theorem 2. *The proposed protocol provides the perfect forward security if the CDH assumption in $\mathbb{G}$ is hard.*

Proof: Assuming that user A and B compute the session key SK by applying Certificateless AKA protocol, therefore, the private keys K_A and SK_B get compromised. Assume that a and b are secret values used by user A and user B when they compute a common session key. For an attacker who possesses SK_A, SK_B, $T_A = aP$ and $T_B = bP$ for secrets a and b, must reveal abP. To reveal the value abP without knowing either a or b, the attacker should be able to solve the CDH problem in $\mathbb{G}$. Under the CDH, the probability is negligible. Therefore, the Certificateless AKA proves the perfect forward secrecy feature.

5.2 Informal analysis of security requirements

We present informally the security features provided by our proposed lightweight Certificateless AKA.

- **Unknown key share**: An attacker the session key to encrypt and sign the message M because at each session a new key is established between A and user B, and it is hard to compute the CDH problem $c = ab$.
- **Key compromise impersonation**: If a user A long-term key leaks, the adversary will send a request to the KGC to query users's partial private key; then the Type I attack is met. However in our protocol if an adversary wants to find the master key or a private key of a user, he has to give aP to seek a; from our assumption of it is a hard problem on the elliptic curve, of a group G with generator P.
- **Key control**: None of the users can compute the key agreement because it is derived from a temporary key and computed by two parties A and B.
- **Key escrow**: Since a malicious KGC can calculate a partial private key Q_i. It does not compute $d_i = (u_i, Q_i)$ because the user A and user B select randomly u_i to complete their private keys.
- **Anonymity**: The proposed protocol protects anonymity of nodes during the mediated signature creation. since the content of the message is not revealed due to the proxy signature blindness.
- **Norepudiation**: Other nodes on the chain can not deny the use of data since they can verify the authenticity of user B since the proxy blind signature is verifiable.
- **Immutability**: Since the data broadcast by user B forms a decentralized ledger; no other user/node can modify its content.
- **Verifiability**: Blockchain/Decentralized transaction are publicly known to the chain. Any user can check the transactions and hash along way back to the previous block.
- **Consensus mechanism**: A user A send a consensus message $K||M$ to the blockchain as a permission to use its data. This is important before the use of data.
- **Unlinkability**: When the signer is revealed, the proxy signer Pr can not identify the association between the message and the blind signature he created. This shown in verification phase, the signer checks only whether $\delta = H((\beta''\mathcal{G}+g.Q_p)||M)$ holds. He is not aware of the original signer's private

key and proxy's private key. Thus the signer knows neither the message nor the signature associated with the signature scheme.

- **Indistinguishable**: The proxy's key is not similar to original signer's private key and proxy keys created by different proxy signers are different from each other, any proxy signature is distinguishable from original signer's signature and different proxy signer's signature are different.

5.3 Performance analysis

In this section, we compare our protocol in terms of security features, computation costs and communication costs with other recent related protocols. Five related protocols were compared including our proposed certificateless AKA. The protocols designed in [4], [11], [1], [5], and our proposed protocol achieve different security properties. We take into consideration five security properties such as **key escrow** avoidance, **distinguishability**, **verifiability**,**unlinkability**, **consensus**,**strong undeniable**,**decentralized** architecture. Table 2 presents the comparison about the achieved security properties. We represent point multiplication as T_m, hash function operations as T_h, bilinear pairing operations as T_e, symmetric encryption and decryption as T_{se}. Table 3 illustrates the comparative charts of computation cost and communication cost. Assuming that the size of $|m| = \frac{160}{80}$ bytes, similar to that of $\mathbb{Z}_q^*$, the size of $|ID| = \frac{80}{8}$ bytes, the size of compressed is reduced to $|\mathbb{G}_1| = 65$ bytes, the size of $|t_c| = 2$ bytes [11].

Table 2. Functionality features comparison

Feature	[4]	[11]	[1]	[5]	Ours
Key escrow	✗	✓	✗	✓	✓
Distinguishability	✓	✗	✗	✗	✓
Unlinkability	✗	✗	✗	✗	✓
Strong undeniability	✗	✗	✗	✗	✓
Decentralized	✗	✗	✗	✓	✓

Table 3. The comparison based on computation and communication costs

Schemes	Computation costs		Communication costs
	User A	User B	
[5]	$2T_m + T_e + 2T_h$	$2T_m + T_e + 2T_h$	$\lvert 2ID + 2\mathbb{Z}_q^* \rvert$
[4]	$6T_m$	$6T_m$	$\lvert 4\mathbb{Z}_q^* + 4G_1 + 2t_c + 2ID \rvert$
[11]	$3T_m + 4T_h$	$7T_m + 4T_h 2$	$\lvert \mathbb{Z}_q^* + G_1 + t_c \rvert$
[1]	$2T_m + 1T_h$	$1T_e + 1T_{se} + 1T_h$	$\lvert 1\mathbb{Z}_q^* + 1G_1 + ID + Right + t_c + MAC \rvert$
Ours	$4T_m + 2T_h$	$4T_m + 2T_h$	$\lvert 2ID + 2\mathbb{Z}_q^* \rvert$

The comparison in Table 3 about computation and communication costs of five protocols shows that our proposed protocol has less computation costs with $4T_m + 2T_h$ on user A and B.

6 Conclusion and future work

Authenticated key agreement protocols are important for critical power devices to provide security and privacy of sensitive information. Thus, a certificateless AKA is proposed. A session key is established between user A and decentralized user B to assure a secure communication. A certificate AKA achieves more security features than the existing compared AKA protocols such as key escrow, avoidance, distinguishability, verifiability, consensus, and strong undeniable. In addition to that, a lightweight proxy blind signature between decentralized users/nodes is presented to provide the anonymity of the content of message. The proposed protocol is secure in a random oracle model. It is a lightweight for low capability devices. In the future, we propose and recommend to design lightweight AKA based on proxy re-signature that can work on both cloud and IoT big data.

Acknowledgements

This work is supported by the Pivot Access Ltd, Kigali, Rwanda

References

[1] Li, T., Zheng, Y., Zhou, T.: Efficient anonymous authenticated key agreement scheme for wireless body area networks. Security and Communication Networks 2017 (2017)

[2] Al-Riyami, S.S., Paterson, K.G.: Certificateless public key cryptography. In: Inter- national conference on the theory and application of cryptology and information security. pp. 452–473. Springer (2003)

[3] Swanson, C.M.: Security in key agreement: Two-party certificateless schemes. Mas- ter's thesis, University of Waterloo (2008)

[4] Saeed, M.E.S., Liu, Q.Y., Tian, G., Gao, B., Li, F.: Akaiots: Authenticated key agreement for internet of things. Wireless Networks 25(6), 3081–3101 (2019)

[5] Mwitende, G., Ye, Y., Ali, I., Li, F.: Certificateless authenticated key agreement for blockchain-based wbans. Journal of Systems Architecture 110, 101777 (2020)

[6] Gervais, M., Sun, L., Wang, K., Li, F.: Certificateless authenticated key agreement for decentralized wbans. In: International Conference on Frontiers in Cyber Security. pp. 268–290. Springer (2019)

[7] Shen, J., Chang, S., Shen, J., Liu, Q., Sun, X.: A lightweight multi-layer authentica- tion protocol for wireless body area networks. Future generation computer systems 78, 956–963 (2018)

[8] Li, X., Peng, J., Kumari, S., Wu, F., Karuppiah, M., Choo, K.K.R.: An enhanced 1- round authentication protocol for wireless body area networks with user anonymity. Computers & Electrical Engineering 61, 238–249 (2017)

[9] Jiang, Q., Lian, X., Yang, C., Ma, J., Tian, Y., Yang, Y.: A bilinear pairing based anonymous authentication scheme in wireless body area networks for mhealth. Journal of medical systems 40(11), 1–10 (2016)

[10] Jia, X., He, D., Kumar, N., Choo, K.K.R.: Authenticated key agreement scheme for fog-driven iot healthcare system. Wireless Networks 25(8), 4737–4750(2019)

[11] Omala, A.A., Kibiwott, K.P., Li, F.: An efficient remote authentication scheme for wireless body area network. Journal of medical systems 41(2), 1–9 (2017)
[12] Wazid, M., Das, A.K., Kumar, N., Conti, M., Vasilakos, A.V.: A novel authentication and key agreement scheme for implantable medical devices deployment. IEEE journal of biomedical and health informatics 22(4), 1299–1309 (2017)
[13] Hou, M., Xu, Q.: A two-party certificateless authenticated key agreement protocol without pairing. In: 2009 2nd IEEE International Conference on Computer Science and Information Technology. pp. 412–416. IEEE (2009)
[14] Li, F., Shirase, M., Takagi, T.: Key management using certificateless public key cryptography in ad hoc networks. In: IFIP international conference on network and parallel computing. pp. 116–126. Springer (2008)
[15] Sayid, J., Sayid, I., Kar, J.: Certificateless public key cryptography: A research survey. International Journal of Security and Its Applications 10(7), 103–118 (2016)
[16] Hankerson, D., Menezes, A.J., Vanstone, S.: Guide to elliptic curve cryptography. Springer Science & Business Media (2006)
[17] He, D., Chen, J., Hu, J.: A pairing-free certificateless authenticated key agreement protocol. International Journal of Communication Systems 25(2), 221–230 (2012)
[18] Zhang, L., Zhang, F., Wu, Q., Domingo-Ferrer, J.: Simulatable certificateless two-party authenticated key agreement protocol. Information Sciences 180(6), 1020–1030 (2010)
[19] Alghazzawi, D.M., Salim, T.M., Hasan, S.H.: A secure proxy blind signature scheme using ecc. In: International Conference on Networked Digital Technologies. pp. 47–52. Springer (2011)
[20] He, D., Chen, Y., Chen, J., Zhang, R., Han, W.: A new two-round certificateless authenticated key agreement protocol without bilinear pairings. Mathematical and Computer Modelling 54(11-12), 3143–3152 (2011)
[21] Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security. pp. 62–73 (1993)

BIOGRAPHY

Dr. Mwitende Gervais is Deputy Principal of Academics and Training, Senior Lecturer at Rwanda Polytechnic/IPRC Gishari. He started his career of education from former National University of Rwanda-NUR for four years and moved to Rwanda Polytechnic in 2013 as Lecturer in ICT department. Gervais is a consultant of ICT industry in Cybersecurity R& D and Compliance. During his 16 years in Education many students were supervised and graduated under his responsibilities. He earned his bachelor degree of computer science from National University of Rwanda, Postgraduate from CDAC Mohali India, Masters of computer science NUR, PhD in Cryptography and Cybersecurity from University of Electronic science and Technology of China-UESTC. He published 6 scientific industry oriented papers in well-known journals and He is a member of Telecommunication Systems-Springer, and Blockchain & Cryptocurrency B2C. He is Certified of ISO27003 of Cybersecurity Lead Auditor, Certified as Inclusive Education Trainer, and recently completed the certification of GVV