IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 14 April 2022, accepted 16 May 2022, date of publication 3 June 2022, date of current version 4 August 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3180165

Efficient Anonymous Authentication for Wireless Body Area Networks

CHUANG LI AND CHUNXIANG XU, (Member, IEEE)
School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

Corresponding author: Chunxiang Xu (chxxu@uestc.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61872060 and Grant 62002050; in part by the National Key Research and Development Program of China under Grant 2017YFB0802000; and in part by the Key Laboratory of Financial Mathematics of Fujian Province University (Putian University), China, under Grant JR201901.

ABSTRACT Advances in integrated circuit and wireless communication technologies are making wireless body area networks (WBANs) an increasingly important medical paradigm. By collecting, uploading, and processing real-time physical parameters, WBANs assist clients in better recognizing and managing their bodies. Besides conveniences it brings, WBANs are facing the risk of clients' privacy leakage during data transmission. Anonymous authentication schemes were proposed to resolve this challenge, and latest schemes ensure that even if a WBAN client's private key is exposed, previous session keys generated by this client cannot be compromised (known as forward security). Unfortunately, previous forward secure schemes need bilinear pairing operations, which is undesirable in computation-resource-bounded WBANs. Furthermore, the property, that once a WBAN client's private key is exposed, previous sessions shouldn't be identified (dubbed forward anonymity), hasn't been considered in existing works. In response to the above challenges, in this paper, we propose an identity-based authenticated encryption method without pairing, and based on this method, we construct an anonymous authentication scheme. Subsequent security and performance analyses demonstrate that our schemes are secure (including forward anonymous) under the random oracle model, and practical in WBANs with limited resources.

INDEX TERMS Identity-based cryptography, free pairing, forward anonymity, wireless body area networks.

I. INTRODUCTION

Wireless body area networks (WBANs), which allow clients to monitor their physical status remotely from medical institutions in real-time, are emerging and promising paradigms in modern medical systems. With the rapid advances in integrated circuit and wireless communication technologies [1], but also with the increase of the world's average age and the advent of population aging in many countries, WBAN plays an increasingly important role in easing the burden on medical institutions.

WBAN consists of three entities, namely sensor nodes, a smart portable device (SPD), and application providers (AP). Among them, sensor nodes are machines placed in, on, and around clients' bodies to monitor their physical parameters (such as body temperature, heart rate, moving speed, etc.) or surrounding environmental parameters (such as ambient temperature, humidity, wind speed, etc.). Sensor nodes send these data to an SPD. The SPD could be a smartphone, a dedicated data sink, or other portable devices. After collecting data from sensor nodes, the SPD transmits these data to an AP. The AP may be a doctor or a medical server in a hospital, a clinic, or a health center. By analyzing data sent by the SPD, the AP returns feedback results to the SPD. In this paper, we take sensor nodes and the SPD as a whole (dubbed WBAN client), and focus on communications between the WBAN client and the AP (dubbed external communication). Fig. 1 presents an typical architecture of WBAN.

As a result of openness, mobility, signal noise and other characteristics of external communication [2], an adversary may intercept, eavesdrop on, modify, or forge these transmitted data. Moreover, the portability of sensor nodes and the SPD limits their power of storage and computation. Authentication for WBANs is one solution to these challenges [3]. An authentication scheme enables a WBAN client and an AP to achieve mutual authentication, and negotiate a session key to secure subsequent conversations [4]–[6]. Unfortunately, since clients' identities are transmitted in plaintexts,

The associate editor coordinating the review of this manuscript and approving it for publication was Diana Gratiela Berbecaru.

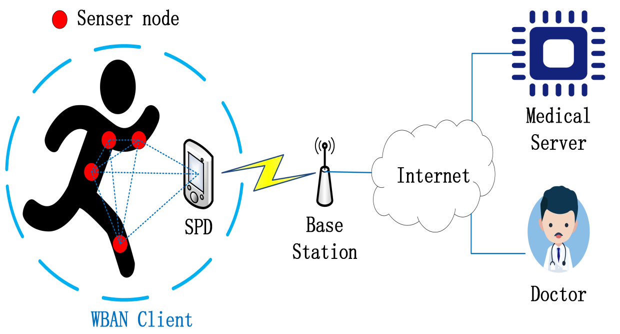

FIGURE 1. Typical WBAN Architecture.

an adversary can distinguish data from different clients. Thus, it's easy for the adversary to obtain time and frequency of data transmission, which may expose clients' privacy. For example, an adversary can easily judge whether a WBAN client has heart disease or fever based on this client's communicating frequencies, for the reason that transmitting frequencies of a client suffering from heart disease (once a minute), and fever (once an hour) are different.

Anonymous authentication (AA) resolves the above challenge by using a certain way to hide clients' identities [7]. In this case, transmitted messages cannot be identified or linked to the same WBAN client. Authentication schemes are generally based on public key infrastructure (PKI), that requires a certificate authority to sign, distribute, and revoke clients' certificates [8]–[11]. Certificates management is cumbersome and undesirable in WBANs. In order to avoid this problem, identity-based AA was proposed [12]. Different from the PKI that binds a client's identity with his/her public key by a certificate, identity-based scheme enables a private key generator (PKG) to generate clients' public keys directly from their identities [8]. Recently, a new security requirement, known as forward security (once the long-term private key of a client is exposed session keys of previous sessions should not be compromised), is proposed [13], [14], and some subsequent works are proved to be forward secure [15]–[18].

However, there are several challenges that need to be paid attention to. First, previous forward secure identity-based AA schemes need bilinear pairing operations, which is time-consuming especially in a resource-limited device of a WBAN client. Second, the property, that once a WBAN client's private key is exposed, previous sessions shouldn't be identified (dubbed forward anonymity), hasn't been considered in existing works. Third, in identity-based AA schemes, it is natural for a PKG to have access to a client's private key. Thus, it is critical for an identity-based AA scheme to prevent the PKG from compromising the security of messages.

In response to the above challenges, we propose identity-based anonymous authentication for WBANs without pairing. To summarize, main contributions of our paper are listed as follows:

1) We propose an identity-based anonymous authenticated encryption scheme without bilinear pairing, dubbed IB-AAE. Our IB-AAE combines the functions of anonymous authentication in [19] and identity-based authentication encryption, and achieves forward security.
2) We propose an identity-based anonymous authentication scheme, dubbed IB-AA. In our IB-AA scheme, once a WBAN client's private key is exposed, or the PKG is compromised by an adversary, the adversary cannot compromise previous sessions. We also demonstrate the security (including forward security and forward anonymity) of our IB-AA scheme under the random oracle model.
3) Experimental performance comparisons indicate that our scheme needs less computation and communication overhead compared with previous anonymous authentication schemes.

Roadmap. Section II shows related works. We present preliminaries, system model, and objectives of this paper in section III. Corresponding constructions are given in section IV. We prove the security of our proposed schemes in section V, and show the performance of IB-AA in section VI. Section VII concludes our works.

II. RELATED WORKS

Based on our contribution, we describe related works from two aspects, namely identity-based anonymous authenticated encryption, and anonymous authentication for WBAN.

Signcryption (i.e., authenticated encryption) was first proposed by [20]. In a signcryption scheme, a sender sends an encrypted message along with his/her identity information to a specific receiver [21]–[24]. It requires that only the receiver can decrypt the message, and it enable the receiver to authenticate the sender through the decryption. Identity-based signcryption was then proposed [25], [26] thereafter. Recently, researchers bring security concerns of identity concealment and x-security into signcryption, and introduced a new cryptographic primitive, named higncryption [19], [27]. By identity concealment, we denote that participants' identities of this scheme should not be leaked to any third party. In the last few years, identity-based higncryption was proposed [28]–[30]. Unfortunately, these schemes are based on bilinear pairing operations.

In the research of anonymous authentication for WBAN, a few studies were conducted to meet forward security. [31] proposed a revocable and forward secure anonymous authentication scheme to revoke expired or exposed clients' keys, while the sender of this scheme can be impersonated according to [32]. [12] conducted an identity-based anonymous authentication scheme that is forward secure and provable secure. Later, [16] described an key replacement attack, and showed that [12] fails to achieve anonymity. [17] presented an online/offline signature, and constructed a anonymous authentication scheme based on this signature, while [33] demonstrated that there is a forgery attack in this scheme. [33] also gave an improved version of the signature algorithm. [34] proposed a anonymous authentication with location privacy using bilinear pairings.

III. PRELIMINARIES

A. REVIEW OF HARD PROBLEMS

Let $\mathbb{G}$ be an additive cyclic group, and G is a generator of $\mathbb{G}$ with a prime order q. We select two random numbers $a, b \in_R Z_q^*$, where $\in_R$ denotes that one randomly selects elements from a set. According to previous works [35], [36], hard problems are defined as follows.

CDH Problem. Given $(G, aG, bG) \in \mathbb{G}^3$, the CDH problem is to compute abG.

DDH Oracle. Given $(G, aG, bG, T) \in \mathbb{G}^4$, this oracle returns *True* if $T = abG$, and *False* otherwise.

GDH Problem. Given $(G, aG, bG) \in \mathbb{G}^3$, the GDH problem is to compute abG with the aid of DDH oracle. There is a public GDH assumption that the probability of solving GDH problem within polynomial time is negligible [37], [38].

B. PUBLIC-KEY SIGNATURE

We review the definitions of Sign and Verify algorithms in a public-key signature scheme. This scheme can be any secure public-key signature scheme.

$Sign(s, M) \rightarrow \sigma$: This is a probabilistic algorithm that takes as input a signer's private key s and a message M, and outputs the corresponding signature σ.

$Verify(P, M, \sigma) \rightarrow$ *False* or *True*: This is a deterministic algorithm that takes signer's public key P, message M, and signature σ, and outputs *True* if σ is a valid signature of M, and *False* otherwise.

C. AUTHENTICATED ENCRYPTION WITH ASSOCIATED DATA

In this paper, authenticated encryption with associated data (AEAD) is used to protect the confidentiality of transmitted data. Based on works in [39], [40], and [41], we provide the definition and security game of AEAD as follows.

Definition 1: (AEAD) An AEAD scheme consists of the following three algorithms.

Initialization. This is a probabilistic algorithm that takes a security parameter κ as input, and outputs random-number space $\mathcal{N}$, plaintext space $\mathcal{M}$, ciphertext space $\mathcal{C}$, public-header space $\mathcal{H}$, symmetric-key space $\mathcal{K}$, and a symmetric key $K \in \mathcal{K}$.

Enc. This is a probabilistic algorithm that takes a random number $N \in \mathcal{N}$, a plaintext $M \in \mathcal{M}$, corresponding public header information $H \in \mathcal{H}$, and K as input, and outputs a ciphertext $C \in \mathcal{C}$.

Dec. This is a deterministic algorithm that takes K, C, and H as input, and outputs corresponding plaintext M or an error symbol "$\perp$".

Definition 2: (AEAD security) Security game of AEAD is defined as follows:

Initialization: A simulator $\mathcal{S}$ generates all the value spaces and the symmetric key K, and keeps K privately. $\mathcal{S}$ also selects $\sigma \in_R \{0, 1\}$.

Enc: Upon receiving query (H, N, M), $\mathcal{S}$ returns $C = Enc_K(H, N, M)$ if $\sigma = 1$, and outputs a random string $Str \in_R \mathcal{C}$ otherwise. $\mathcal{S}$ stores tuple (H, M, C) or (H, M, Str) in list $\mathcal{L}_M$ that is initialized empty.

Dec: Upon receiving query (H, C), $\mathcal{S}$ returns the corresponding M if H, C are stored in list $\mathcal{L}_M$. Otherwise, $\mathcal{S}$ returns "$\perp$" if $\sigma = 0$, and returns $Dec_K(H, C)$ if $\sigma = 1$.

Guess: Adversary $\mathcal{A}$ outputs a guess σ', and wins the game if $\sigma' = \sigma$.

Definition: We say that a scheme is AEAD secure if the advantage for any probabilistic polynomial-time (PPT) adversary to win the game is negligible. The advantage is $Adv_{\mathcal{A}}^{AEAD} = \left|2 \cdot Pr[\sigma' = \sigma] - 1\right|$.

According to the above definitions, we can conclude that in an AEAD secure scheme any PPT adversary cannot generate the same ciphertext with different plaintexts. We can also learn from [40] that any PPT adversary cannot generate the same ciphertext with different symmetric keys. In this paper, the encryption algorithm is denoted by $C = Enc_K(M)$ that takes a symmetric key K and a message M as input, and outputs a ciphertext C; the decryption algorithm is denoted by $M = Dec_K(C)$ that takes a symmetric key K and a ciphertext C, and outputs the corresponding message M. Here, we treat the encryption and decryption algorithms as subroutines, and omit random number N and public header information H for simplicity.

D. SYSTEM MODEL

We consider the generalized system model which consists of three entities, namely private key generator (PKG), application provider (AP), and WBAN client, as shown in Fig. 2.

In general, the PKG is assumed to be an honest but curious third party. An application provider (AP) could be a remote server or a remote system at a health center, a clinic, or a hospital that could provide diagnosis and treatment measures. WBAN client is a general term for SPDs and sensor nodes. First, the PKG chooses its master key, and generates public-private key pairs of APs. Second, WBAN clients' private keys are generated by the PKG. Third, a WBAN client and an AP authenticate each other, and the WBAN client could get services of the AP after authentication.

E. OBJECTIVES

An identity-based anonymous authentication scheme for WBANs is considered to achieve the following objectives.

- **Unforgeability**: An adversary cannot impersonate a WBAN client or an AP to establish a valid session without the corresponding private key.
- **Anonymity**: Any message sent by a targeted WBAN client cannot be identified by a PPT adversary.

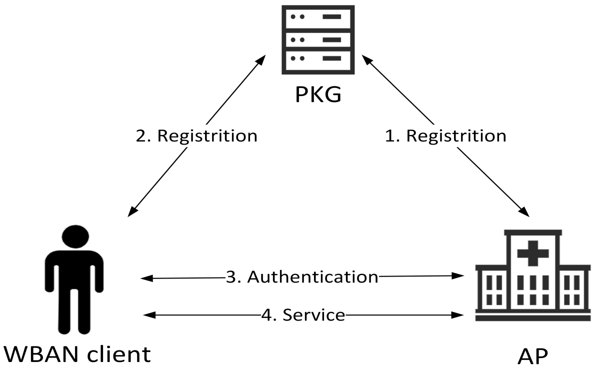

FIGURE 2. System model.

- **Forward security**: An adversary cannot compromise a session key of any previous session sent by a WBAN client with the client's private key.
- **Forward anonymity**: An adversary cannot identify any previous session sent by a WBAN client with the client's private key.
- **Scalability**: It is not necessary for an AP to store any WBAN client's credential.

IV. CONSTRUCTIONS

A. IDENTITY-BASED ANONYMOUS AUTHENTICATED ENCRYPTION (IB-AAE)

Initialization. Let $E : y^3 = x^2 + ax + b \mod p$ be an elliptic curve. In this elliptic curve, a, b are two coefficients, and p is a big prime. Let $\mathbb{G}$ be a cyclic additive group with order q and generator G. Let $H_1 : \{0, 1\}^* \to Z_q^*$ and $H_2 : \{0, 1\}^* \to \mathcal{K}$ be two secure hash functions where $\mathcal{K}$ is the key space of an AEAD algorithm. Let $Enc(.)$ and $Dec(.)$ be the encryption algorithm and decryption algorithm in an AEAD secure scheme. Denote by $Enc_K(M)$ that encrypt message M using key K, and output the corresponding ciphertext. Denote by $Dec_K(C)$ that decrypt ciphertext C with key K, and output the corresponding plaintext. The PKG also runs as follows to set up this system.

1) Select its master key $s \in_R Z_q^*$, and compute the corresponding public key $P_{pub} = sG$.
2) Publish system parameters $params = \{a, b, p, q, G, P_{pub}, H_1, H_2\}$.

Registration. The registration phase is executed through secure channels that cannot be compromised by an adversary. In this scheme, the generation method of the AP's private key can be arbitrary. Finally, the AP generates its public and private key pair ($P_{AP} = s_{AP}G, s_{AP}$) in its own way, and publishes P_{AP}. The client runs the following steps.

Client A sends its identity id_A to the PKG in a secure channel for registration, and the PKG runs as follows.

1) Select $r_A \in_R Z_q^*$, and compute $R_A = r_A G$, $h_A = H_1(id_A, R_A, P_{pub})$, and $d_A = (r_A + h_A s)$.
2) Return (R_A, d_A) to client A in a secure channel.

Client A then sets d_A as his/her private key, and publishes R_A as his/her public key.

IB-AAE. When a client A transmits a message M to the AP, client A executes the following steps.

1) Select $k' \in_R Z_q^*$, and compute $k = H_1(M, k')$, $T = (d_A + k)G$, $V = (d_A + k)P_{AP}$.
2) Compute session key $K = H_2(V, T, id_{AP}, P_{AP})$, and ciphertext $C = Enc_K\ (id_A, R_A, k', M)$.
3) Return (T, C) to the AP.

UnIB-AAE. On receiving (T, C), the AP runs as follows.

1) Compute $V = s_{AP}T$ and session key $K = H_2(V, T, id_{AP}, P_{AP})$.
2) Decrypt the ciphertext $(id_A, R_A, k', M) = Dec_K(C)$.
3) Compute $h_A = H_1(id_A, R_A, P_{pub})$, $P_A = R_A + h_A P_{pub}$, and $k = H_1(M, k')$.
4) Check whether equation $T = P_A + kG$ holds. If so, the AP accept M, id_A, R_A, and aborts otherwise.

B. IDENTITY-BASED ANONYMOUS AUTHENTICATION (IB-AA)

Now, we extend the above IB-AAE to identity-based anonymous authentication (IB-AA).

Initialization. Besides the execution of Initialization phase in section IV-A, the PKG does as follows.

1) Select a secure hash function $H_3 : \{0, 1\}^* \to \mathcal{K}^2$.
2) Publish system parameters $params = \{a, b, p, q, G, P_{pub}, Enc, Dec, H_1, H_2, H_3\}$, and keep s.

Registration. As described in Registration of section IV-A, client A's private and public key pair is (d_A, R_A), and the AP's private and public key pair is (s_{AP}, P_{AP}).

Authentication. In this phase, client A and the AP authenticates each other through three phases. Here, we assume that client A obtains the AP's public key before authentication.

Phase 1. Client A executes the following steps.

1) Choose $k_A \in_R Z_q^*$, and compute $T_A = (k_A + d_A)G$.
2) Send T_A to the AP.

The AP executes the following steps at the same time.

1) Choose $k_{AP} \in_R Z_q^*$, and compute $T_{AP} = k_{AP}G + P_{AP}$.
2) Send T_{AP} to the AP.

Phase 2. After receiving T_{AP}, client A executes the following steps.

1) Compute $V = (d_A + k_A)T_{AP}$, and derive session key $(K_1, K_2) = H_3(V, T_A, T_{AP})$.
2) Encrypt $C_A = Enc_{K_1}(id_A, R_A, k_A)$, and send C_A to the AP.

After receiving T_A, the AP executes the following steps.

1) Compute $V = (s_{AP} + k_{AP})R_A$, and derive session key $(K_1, K_2) = H_3(V, T_A, T_{AP})$.
2) Encrypt the message $C_{AP} = Enc_{K_1}(k_{AP})$, and send C_{AP} to A.

Phase 3. After receiving C_{AP}, A executes the following steps.

1) Decrypt $k_{AP} = Dec_{K_1}(C_{AP})$.

2) Check whether $T_{AP} = P_{AP} + k_{AP}G$ is valid. If it is valid, A sets session key as K_2; otherwise, A aborts.

After receiving C_A, the AP executes the following steps.

1) Decrypt $(id_A, R_A, k_A) = Dec_{K_1}(C_A)$.
2) Compute $h_A = H_1(id_A, R_A, P_{pub})$, and calculate $P_A = R_A + h_A P_{pub}$.
3) Check whether $T_A = k_A G + P_A$ is valid. If it is valid, the AP sets session key as K_2; otherwise, the AP aborts.

V. ADVERSARIAL MODEL AND SECURITY PROOFS

A. ADVERSARIAL MODEL

Let Γ denote a client or an AP, and Γ^i denote the ith instance of a Γ. We assume that all the clients and APs register at the same PKG. We first prove that our IB-AAE is secure under the random oracle model. Based on it, we prove the security of our IB-AA scheme under the random oracle model. We define the ability of a probabilistic polynomial-time adversary in IB-AAE and IB-AA respectively as follows.

1) ADVERSARIAL MODEL OF ANONYMOUS AUTHENTICATED ENCRYPTION

In this model, the adversary is allowed to access the following oracles.

Create oracle: Upon receiving query id_i (the identity of a client), $\mathcal{S}$ runs algorithms *Registration* of IB-AAE, gets the private key d_i, stores (id_i, d_i) into a list $\mathcal{L}_{Key}$, and stores id_i in list $\mathcal{L}_{honest}$.

Corrupt-SK oracle: Upon receiving query id_i, if id_i is recorded in list $\mathcal{L}_{Key}$, $\mathcal{S}$ returns the private key of client id_i, and removes id_i from $\mathcal{L}_{honest}$; otherwise, $\mathcal{S}$ returns $\perp$.

Create-AP oracle: Upon receiving query id_{AP}, $\mathcal{S}$ selects $s_{AP} \in Z_q^*$ randomly as this AP's private key, returns $P_{AP} = s_{AP}G$, stores (P_{AP}, s_{AP}) in a list $\mathcal{L}_{AP}$, and stores id_{AP} in list $\mathcal{L}_{honest}$.

Corrupt-AP oracle: Upon receiving query id_{AP}, if $id_{AP} \in \mathcal{L}_{AP}$, $\mathcal{S}$ returns the private key of id_{AP}, and removes id_{AP} in $\mathcal{L}_{honest}$; otherwise, $\mathcal{S}$ returns $\perp$.

The above oracles consider the adversary's ability that the adversary can compromise the client's and the AP's private key. The hash functions are also assumed to be random oracles.

H_1 oracle: Upon receiving query *Str* (an arbitrary-length string), $\mathcal{S}$ returns a random element h_1 in H_1's output space, and stores (Str, h_1) into a list $\mathcal{L}_{H_1}$ if *Str* wasn't in $\mathcal{L}_{H_1}$ before; otherwise, $\mathcal{S}$ returns the corresponding element in $\mathcal{L}_{H_1}$.

H_2 oracle: Upon receiving query *Str* (an arbitrary-length string), $\mathcal{S}$ returns a random element h_2 in H_2's output space, and stores (Str, h_2) into a list $\mathcal{L}_{H_2}$ if *Str* wasn't in $\mathcal{L}_{H_2}$ before; otherwise, $\mathcal{S}$ returns the corresponding element in $\mathcal{L}_{H_2}$.

H_3 oracle: Upon receiving query *Str* (an arbitrary-length string), $\mathcal{S}$ returns a random element h_3 in H_3's output space, and stores (Str, h_3) into a list $\mathcal{L}_{H_3}$ if *Str* wasn't in $\mathcal{L}_{H_3}$ before; otherwise, $\mathcal{S}$ returns the corresponding element in $\mathcal{L}_{H_3}$.

Moreover, the adversary is allowed to query **IB-AAE** and **UnIB-AAE** oracle in IB-AAE. To prove that our IB-AAE is secure once the random numbers of this scheme are exposed (dubbed x-security), we also build a **Corrupt-R** oracle.

IB-AAE oracle: Upon receiving query (id_A, id_B, M) (represent the sender, the receiver, and a message respectively), if $id_A, id_B \in \mathcal{L}_{Key}$, $\mathcal{S}$ executes *IB-AAE* phase of IB-AAE, and returns the corresponding output *Cipher*. Otherwise, it outputs $\perp$. $\mathcal{S}$ Stores $(Cipher, id_B, k)$ in list $\mathcal{L}_x$ where k is the random number generated during *IB-AAE* phase.

UnIB-AAE oracle: Upon receiving query $(id_B, Cipher)$, if id_B isn't in the list $\mathcal{L}_{Key}$, $\mathcal{S}$ outputs $\perp$; otherwise, $\mathcal{S}$ runs *UnIB-AAE* phase, and returns the corresponding output.

Corrupt-R oracle: Upon receiving query $(id_B, Cipher)$, if $(id_B, Cipher) \in \mathcal{L}_R$, $\mathcal{S}$ outputs the corresponding k; otherwise, $\mathcal{S}$ outputs $\perp$.

With the help of the above oracles, we construct two security games, dubbed OU and IC, as below. Concretely, OU is built in terms of unforgeability and x-security, and IC is constructed in terms of anonymity, forward security, and forward anonymity.

Definition 3: (OU) The security game of OU is defined as follows:

Initialization: $\mathcal{S}$ runs as *Initialization* phase described in section IV.

Query Phase: The adversary is allowed to query all the above oracles polynomial times adaptively.

Forgery Phase: The adversary chooses (id_s^*, id_r^*, M^*) where $id_s^*, id_r^* \in \mathcal{L}_{honest}$. During the forgery phase, the adversary is unallowed to query **IB-AAE**(id_s^*, id_r^*, M^*), **Corrupt-SK**(id_s^*), or **Corrupt-SK**(id_r^*).

Challenge: The adversary outputs a forgery for the output of **IB-AAE**(id_s^*, id_r^*, M^*). If the forgery is valid, the adversary wins this game.

Definition: An IB-AAE scheme is OU secure if the probability $Pr_{\mathcal{A}}^{OU}$ for any PPT adversary in winning this game is negligible.

Definition 4: (IC) The security game of IC is defined as follows:

Initialization: $\mathcal{S}$ runs as *Initialization* phase described in section IV.

Query Phase 1: The adversary is allowed to query all the above oracles polynomial times adaptively.

Challenge: The adversary chooses two tuples $(M_0^*, id_{s_0}^*, id_r^*)$ and $(M_1^*, id_{s_1}^*, id_r^*)$ where M_0^*, M_1^* are equal in length and $id_{s_0}^*, id_{s_1}^*, id_r^* \in \mathcal{L}_{honest}$. The adversary sends these tuples to $\mathcal{S}$, and $\mathcal{S}$ selects a bit $\sigma \in \{0, 1\}$ randomly. $\mathcal{S}$ then sets $id_s^* = id_{s_\sigma}^*$, and generates the target output $Cipher^*$ with the help of the above oracles. If $Cipher^*$ is output by the **AAE** oracle, $\mathcal{S}$ aborts; otherwise, $\mathcal{S}$ sends $Cipher^*$ to the adversary.

Query Phase 2: The adversary is unallowed to query **UnIB-AAE**$(id_r^*, Cipher^*)$, **Corrupt-R**$(id_r^*, Cipher^*)$, or **Corrupt-SK**(id_r^*).

Guess: The adversary outputs a bit σ'. If $\sigma' = \sigma$, the adversary wins this game.

Definition: An IB-AAE scheme is IC secure if any PPT adversary's advantage in winning this game is negligible.

TABLE 1. Symbols and descriptions.

Symbol	Description
M_{in}	message the oracle receives
M_{out}	message sent by the oracle
sid_Γ^i	this session's session identity
pid_Γ^i	the partner's identity of this instance
ssk_A^i	this session's session key
acc_Γ^i	the oracle's state
$msg_{sid_A^i}^{(1)}$	the instance's first message
$msg_{sid_A^i}^{*}$	the instance's message (not the first)

If an IB-AAE scheme is OU-secure, we can conclude that this scheme satisfies unforgeability, anonymity, and x-security. If an IB-AAE scheme is IC-secure, we can conclude that this scheme achieves confidentiality, forward security, and forward anonymity. Then we have the following definition.

Definition 5: (IB-AAE security) An IB-AAE scheme is IB-AAE secure if it is OU secure and IC secure for any sufficiently large security parameter, and against any PPT adversary.

2) ADVERSARIAL MODEL OF IDENTITY-BASED ANONYMOUS AUTHENTICATION

In this model, the adversary is allowed to access **Create**, **Corrupt-SK**, **Create-AP**, **Corrupt-AP** H_1, H_2, H_3 oracles defined in section V-A1. The adversary can also access the following oracles. By entity Γ we denote a client or an AP.

Corrupt-SSK oracle: Upon receiving query Γ^i, $\mathcal{S}$ returns the session key of Γ^i.

Authentication oracle. For an entity Γ, this oracle takes a receiving message M_{in} along with the PKG's public key P_{pub}, Γ's identity id_Γ and private key d_Γ (if Γ is an AP, there should be s_Γ) as input, and the outputs are shown in equation 1.

$$Authentication(P_{pub}, id_\Gamma, d_\Gamma, M_{in}) \rightarrow (M_{out}, acc_\Gamma^i, sid_\Gamma^i, pid_\Gamma^i, ssk_\Gamma^i) \quad (1)$$

The symbols are explained in table 1. Moreover, in this equation, acc_Γ^i is a state with the following four situations. 00) this algorithm is completed; 01) this algorithm is waiting for the next message; 10) this algorithm encounters something wrong and aborts; 11) this algorithm hasn't received the next message within a specific time and expires.A client A and an AP are partners if there exists i, j that $pid_A^i = id_{AP}$, $sid_{AP}^j = id_A$, and $sid_A^i = sid_{AP}^j$. In this oracle, we set $sid_\Gamma^i = i$ for simplicity.

Specifically, $\mathcal{S}$ replies queries for **Authentication** oracle as follows.

- Upon receiving query $(Start, id_A)$, if $id_A \notin \mathcal{L}_{honest}$, $\mathcal{S}$ returns $\perp$ and aborts; otherwise, $\mathcal{S}$ computes $i = i + 1$, $pid_A^i = \perp$, $acc_A^i = 01$, and $ssk_A^i = \perp$. The output is $(msg_{sid_A^i}^{(1)}, acc_A^i, sid_A^i, pid_A^i, ssk_A^i)$ where $msg_{sid_A^i}^{(1)}$ is the instance's first message.
- Upon receiving query $(id_A, i, msg_{sid_A^i}^*)$, $\mathcal{S}$ sets $M_{in} = msg_{sid_A^i}^*$ and runs under the specification of **Authentication** algorithm. $\mathcal{S}$ sets the output message as M_{out}. If $msg_{sid_A^i}^*$ is the last message, $\mathcal{S}$ sets the session key as ssk_A^i, stores (ssk_A^i, sid_A^i) into a list L_{Key}.

According to the above oracles, we give the definitions below. Note that an unexposed session means that the session key, and the private keys of participants are not acquired by the adversary.

Definition 6: (Label security) An identity-based anonymous authentication scheme is label-secure if the following events' probabilities are negligible:

- At least three sessions has the same session identity.
- If $sid_A^i = sid_{AP}^j$ for a client A and an AP, the following events occurs: 1) both client A and AP are initiators or responders; 2) $ssk_A^i \neq ssk_{AP}^j$; 3) $pid_A \neq \perp \wedge pid_A \neq id_{AP}$, or $pid_{AP} \neq \perp \wedge pid_{AP} \neq id_A$.

Definition 7: (Impersonation security) The security game is defined as follows:

Setup: $\mathcal{S}$ runs as *Initialization* phase described in section IV.

Query Phase: The adversary is allowed to query all the oracles polynomial times adaptively.

Challenge: The adversary chooses (id_s^*, id_{AP}^*) as the targeted client and the targeted AP, where $id_s^*, id_{AP}^* \in \mathcal{L}_{honest}$. The adversary is not allowed to query **Corrupt-SK**(id_s^*) or **Corrupt-AP**(AP^*) during this phase.

Test: The adversary $\mathcal{A}$ wins the game if $\mathcal{A}$ completes the **Challenge** phase without being aborted.

Definition: An IB-AA scheme is impersonation secure if any PPT adversary's advantage $Adv_{\mathcal{A}}^{\mathcal{IMP}}$ in winning this game is negligible.

Definition 8: (Anonymous session-key (ASK) indistinguishability) The ASK indistinguishability is constructed in terms of anonymity, forward security, and forward anonymity. The security game is defined as follows:

Setup: $\mathcal{S}$ runs as *Initialization* phase described in section IV.

Query Phase: The adversary is allowed to query all the oracles polynomial times adaptively.

Challenge: The adversary chooses two tuples $(id_{s_0}, id_{AP}^*, Start)$ and $(id_{s_1}, id_{AP}^*, Start)$ where $id_{s_0}^*, id_{s_1}^*, id_{AP}^* \in \mathcal{L}_{honest}$. The adversary then sends these two tuples to $\mathcal{S}$, and $\mathcal{S}$ selects $\sigma \leftarrow \{0, 1\}$ randomly. $\mathcal{S}$ then sets $id_s^* = id_{s_\sigma}^*$ as the target client, and acts according to the specification of **Authentication** oracle. Upon receiving query **Corrupt-SSK**(sid^*) for the targeted session by the adversary, $\mathcal{S}$ returns the corresponding session key if $\sigma = 1$; otherwise, $\mathcal{S}$ returns a random element in key space.

Guess: The adversary guesses a bit σ', and wins this game if $\sigma' = \sigma$.

Definition: An IB-AA scheme is ASK indistinguishable if any PPT adversary's advantage in winning this game is negligible. The advantage is $Adv_{\mathcal{A}}^{\mathcal{ASK}-\mathcal{IN}} = \left|2 \cdot Pr[\sigma' = \sigma] - 1\right|$.

Definition 9: (Strong IB-AA security) An IB-AA scheme is strongly IB-AA secure, if it is label secure, impersonation secure, and ASK indistinguishable for any sufficiently large security parameter and any PPT adversary defined above.

B. SECURITY PROOF

Assume that the adversary is able to issue at most q_1 queries to H_1 oracle, q_2 queries to H_2 oracle, q_3 queries to H_3 oracle, q_C queries to **Create** oracle, q_{CAP} queries to **Create-AP** oracle, q_R queries to **Corrupt-R** oracle, q_{AP} queries to **Corrupt-AP** oracle, q_{SK} queries to **Corrupt-SK** oracle, q_{AAE} queries to **IB-AAE** oracle, q_{UnAAE} queries to **UnIB-AAE** oracle, q_{SSK} queries to **Corrupt-SSK** oracle, and q_{Auth} queries to **Authentication** oracle. We have three theorems below.

Theorem 1: Our IB-AAE scheme is IB-AAE secure in the random oracle model under the GDH assumption.

Theorem 2: Our IB-AA scheme is strong IB-AA secure in the random oracle model under the GDH assumption.

1) PROOF OF THEOREM 1

Lemma 1: Our IB-AAE scheme is OU secure in the random oracle model under the GDH assumption and AEAD security.

Proof. Given $A = aP, B = bP \in \mathbb{G}$ without a, b, we now demonstrate that the simulator $\mathcal{S}$ can solve $GDH(A, B)$ with a non-negligible probability if the adversary $\mathcal{A}$ breaks OU security with a non-negligible probability ε_1.

Initialization: $\mathcal{S}$ selects system parameters *params* as specification, and keeps PKG's private key s secretely. $\mathcal{S}$ randomly chooses two random numbers $i^* \in \{1, 2, \ldots, q_C\}$ and $j^* \in \{1, 2, \ldots, q_{CAP}\}$.

Query Phase: $\mathcal{S}$ simulates H_1, H_2 oracles as specification, and simulates other oracles as follows:

Create oracle: $\mathcal{S}$ maintains a counter i that is initiated to be zero. Upon receiving a Create query id, $\mathcal{S}$ computes $i = i + 1$, and checks whether $i = i^*$ or $i = j^*$. If $i \neq i^*$, $\mathcal{S}$ runs *Registration* phase as specification, $\mathcal{S}$ stores the tuple (id_i, R_i, d_i) into list $\mathcal{L}_{Key}$; if $i = i^*$, $\mathcal{S}$ sets $R_i = A$ and stores the tuple $(id_i, R_i, \perp)$ into list $\mathcal{L}_{Key}$.

Create-AP oracle: $\mathcal{S}$ maintains a counter j that is initiated to be zero. Upon receiving a Create query id, $\mathcal{S}$ computes $j = j + 1$, and checks whether $j = j^*$. If so, $\mathcal{S}$ sets $P_j = B$, and stores the tuple $(id_i, R_i, \perp)$ into list $\mathcal{L}_{Key}$; otherwise, $\mathcal{S}$ runs *Registration* phase as specification.

Corrupt-SK oracle: Upon receiving query id_i, if $id_i \neq id_i^*$, $\mathcal{S}$ runs as specification; otherwise, $\mathcal{S}$ aborts.

Corrupt-AP oracle: Upon receiving query id_j, if $id_j \neq id_j^*$, $\mathcal{S}$ runs as specification; otherwise, $\mathcal{S}$ aborts.

IB-AAE oracle: Upon receiving query (id_s, id_r, M) where id_s is the sender's identity, id_r is the receiver's identity, and M is the message to be sent, $\mathcal{S}$ executes as below:

1) If $id_s \neq id_i^*$, $\mathcal{S}$ returns $Cipher \leftarrow IB - AAE(id_s, id_r, M)$ as specification. $\mathcal{S}$ stores $(k', id_r, Cipher)$ into a list $\mathcal{ST}_C$ that is initiated empty. $\mathcal{S}$ also stores the tuple $(id_r, Cipher, K)$ into list $\mathcal{L}_{GDH}$ if $id_r = id_j^*$.
2) If $id_r \neq id_j^*$, $\mathcal{S}$ sets $k' \in_R Z_q^*$, and calculates $k = H_1(M, k')$, $h_s = H_1(id_s, R_s, P_{pub})$, $P_s = R_s + h_s P_{pub}$, $T = kG + P_s$, and $V = kd_r G + d_r P_s$. $\mathcal{S}$ computes the session key $K = H_2(V, T, id_r, R_r)$ and $C \leftarrow Enc_K(id_s, R_s, k', M)$. $\mathcal{S}$ then returns $Cipher = (T, C)$. $\mathcal{S}$ stores $(k', id_r, Cipher)$ into list $\mathcal{ST}_C$.
3) If $id_s = id_i^*$ and $id_r = id_j^*$, $\mathcal{S}$ sets $k' \in_R Z_q^*$, and computes $k = H_1(M, k')$, $h_s = H_1(id_s, R_s, P_{pub})$, $P_s = R_s + h_s P_{pub}$, and $T = P_s + kG$. $\mathcal{S}$ selects $K \in_R \mathcal{K}$ ensuring that K is different from previous session keys. $\mathcal{S}$ computes $C \leftarrow Enc_K(id_s, R_s, k', M)$, and sends T, C to the adversary. $\mathcal{S}$ stores $(id_r, Cipher, K)$ into list $\mathcal{L}_{GDH}$, and stores $(k', id_r, Cipher)$ into list $\mathcal{ST}_C$.

UnIB-AAE oracle: Upon receiving query $(id_r, Cipher)$, $\mathcal{S}$ executes as below:

1) If $id_r \neq id_j^*$, $\mathcal{S}$ returns $UnIB - AAE(id_r, Cipher)$ as specification.
2) If $id_r = id_j^*$, we consider two cases. If $id_B, Cipher \in \mathcal{L}_{GDH}$, $\mathcal{S}$ obtains the corresponding session key K, and computes $(id_s, R_s, k', M) = Dec_K(Cipher)$. If $id_B, Cipher \notin \mathcal{L}_{GDH}$, $\mathcal{S}$ goes through all the queries in $\mathcal{L}_{H_2}$, and checks whether there exists a tuple (V, T, id_r, R_r) that satisfies $V = CDH(T, P_r)$ with the help of DDH oracle. If the tuple doesn't exist, $\mathcal{S}$ returns $\perp$; otherwise, $\mathcal{S}$ computes $K = H_2(V, T, id_r, R_r)$. Then $\mathcal{S}$ calculates $(id_s, R_s, k', M) = Dec_K(C)$, $k = h_2(M, k')$, and checks whether equation $T = P_s + kG$ holds. $\mathcal{S}$ returns (M, id_s) if the equation holds, and returns $\perp$ otherwise.

Corrupt-R oracle: On query $(id_r, Cipher)$, $\mathcal{S}$ outputs the relevant random number if $(id_r, Cipher) \in \mathcal{ST}_C$, and returns $\perp$ otherwise.

Forgery Phase: $\mathcal{A}$ chooses the target tuple (id_s^*, id_r^*, M^*) where $id_s^*, id_r^* \in \mathcal{L}_{honest}$. If $id_s^* \neq id_{i^*}$ or $id_r^* \neq id_{j^*}$, $\mathcal{S}$ aborts.

Suppose that $\mathcal{A}$ has successfully forges a valid ciphertext $(id_{j^*}, Cipher^*)$. In this case, $\mathcal{A}$ has issued $H_2(V^*, T^*, id_{j^*}, R_{j^*})$ query with overwhelming probability. Thus, $\mathcal{S}$ could get the contents and the output of this query, and can derive $(id_{i^*}, R_{i^*}, k'^*, M^*) = Dec_{K^*}(C)$. Due to equation 2, $\mathcal{S}$ could solve $GDH(A, B)$ by computing $CDH(A, B) = V^* - h_{j^*} sB - kB$.

$$\begin{aligned} V^* &= (d_{i^*} + k)s_{j^*}G = (a + h_{i^*}s + k)bG \\ &= abG + h_{j^*}sB + kB \end{aligned} \tag{2}$$

The probability of the event that $\mathcal{S}$ fails the simulation is analyzed as follows:

E1: $\mathcal{A}$ breaks the AEAD security. This happens with negligible probability.

E2: The target K^* is the same with other outputs of IB-AAE. Concretely, the targeted temporary value is $V^* = (d_{i^*} + k^*)s_{j^*}G$, and the other temporary value is $V = (d_s + k)s_rG$. We consider two cases below:

1) $id_r \neq id_j^*$. Due to the randomness of H_2 oracle, the probability of $K = K^*$ is at most $q_2^2/2|\mathcal{K}|$.
2) $id_r = id_j^*$. If $V^* \neq V$ or $T^* \neq T$, the probability of $K = K^*$ is at most $q_2^2/2|\mathcal{K}|$. Otherwise, we can easily conclude that $k+d_s = k^*+d_{i^*}$, that is, $k = k^*+d_{i^*}-d_s$. Since k and k^* are output by H_1 oracle, the probability of this event is at most $q_1^2/2q$.

Therefore, $Pr[E2] \leq max\{q_2^2/2|\mathcal{K}|, q_1^2/2q\}$.

E3: The target K^* is generated without H_2 oracle by $\mathcal{A}$. The probability of this event is $Pr[E3] \leq 1/|\mathcal{K}|$.

E4: $\mathcal{A}$ issues Corrupt-SK(id_{i^*}) or Corrupt-AP(id_{j^*}). The probability of this event is $Pr[E4] \leq 1 - (q_C - q_{SK})/q_C \times (q_{CAP} - q_{AP})/q_{CAP}$.

In summary, $\mathcal{S}$ can solve $GDH(A, B)$ with the advantage

$$Adv_{\mathcal{S}}^{\mathcal{OU}} \geq \frac{(q_C - q_{SK})(q_{CAP} - q_{AP})\varepsilon_1}{q_C q_{CAP}} (1 - q_{AAE}P_{E2})(1 - \frac{q_{UnAAE}}{|\mathcal{K}|}), \quad (3)$$

where $P_{E2} = max\{\frac{q_2^2}{2|\mathcal{K}|}, \frac{q_1^2}{2q}\}$. Therefore, lemma 1 is proved.

Lemma 2: Our IB-AAE scheme is IC secure in the random oracle model under the GDH assumption and AEAD security.

Proof. Given $A = aP, B = bP \in E$ without a, b, we now demonstrate that the simulator $\mathcal{S}$ can solve $GDH(A, B)$ with a non-negligible probability if the adversary $\mathcal{A}$ breaks IC security with a non-negligible advantage ε_2.

Setup: $\mathcal{S}$ executes as described in the proof of lemma 1. Moreover, $\mathcal{S}$ randomly chooses three random numbers $i^*, j^* \in \{1, 2, \ldots, q_C\}$ and $k^* \in \{1, 2, \ldots, q_{CAP}\}$, and selects two random numbers $y_0, y_1 \in_R Z_q^*$.

Query Phase 1: $\mathcal{S}$ simulates H_1, H_2, Create-AP oracles as described in the proof of lemma 1, and simulates Create oracles as follows.

Create oracle: $\mathcal{S}$ maintains a counter i that is initiated to be zero. Upon receiving a Create query id, $\mathcal{S}$ computes $i = i+1$, and checks whether $i = i^*$, $i = j^*$. If $i \neq i^*$, and $i \neq j^*$, $\mathcal{S}$ runs *Registration* phase as specification, $\mathcal{S}$ stores the tuple (id_i, R_i, d_i) into list $\mathcal{L}_{Key}$; if $i = i^*$, $\mathcal{S}$ sets $R_i = y_0A$, and stores the tuple $(id_i, R_i, \perp)$ into list $\mathcal{L}_{Key}$; if $i = j^*$, $\mathcal{S}$ sets $R_i = y_1A$, and stores the tuple $(id_i, R_i, \perp)$ into list $\mathcal{L}_{Key}$.

Challenge Phase: $\mathcal{A}$ chooses two tuples $(M_0^*, id_{s_0}^*, id_r^*)$ and $(M_1^*, id_{s_1}^*, id_r^*)$ where M_0^*, M_1^* are equal in length, and $id_{s_0}^*, id_{s_1}^*, id_r^* \in \mathcal{L}_{honest}$. $\mathcal{A}$ sends these tuples to $\mathcal{S}$ and $\mathcal{S}$ selects a bit $\sigma \in \{0, 1\}$ randomly. If $id_{s_0} = id_{i^*}$, $id_{s_1} = id_{j^*}$, and $id_r = id_{k^*}$, $\mathcal{S}$ continues; otherwise, $\mathcal{S}$ aborts. $\mathcal{S}$ sets $id_s^* = id_{s_\sigma}^*$, and runs as follows.

$\mathcal{S}$ selects $k'^* \in_R Z_q^*$, and computes $k^* = h_2(M_\sigma^*, k'^*)$,$h_{s_\sigma}^* = H_1(id_r^*, R_r^*, P_{pub})$,$P_{s_\sigma}^* = R_{s_\sigma}^* + h_{s_\sigma}^* P_{pub}$, and $T^* = P_{s_\sigma}^* + k^*G$. $\mathcal{S}$ goes through all the queries in $\mathcal{L}_{H_2}$, and checks whether there exists a tuple (V, T, id_r^*, R_r^*) that satisfies $V = CDH(T, P_r)$ with the help of DDH oracle. If the tuple exists, $\mathcal{S}$ returns $\perp$; otherwise, $\mathcal{S}$ sets $K^* \in_R \mathcal{K}$ so that K^* is different from previous session keys. Then $\mathcal{S}$ calculates $C^* = Enc_{K^*}(id_{s_\sigma}^*, R_{s_\sigma}^*, k'^*, M_\sigma^*)$, returns $Cipher^* = (T^*, C^*)$, and stores $(id_r^*, Cipher^*, K^*)$ into list $\mathcal{L}_{GDH}$.

Query Phase 2: $\mathcal{A}$ can query all the oracles as in **Query Phase 1**.

Guess: Assume that $\mathcal{A}$ outputs the right bit σ with advantage ε_2. We could conclude that $\mathcal{A}$ decrypts the target ciphertext with non-negligible probability. In this case, $\mathcal{S}$ can obtain V^* from the H_2 query. According to equation 4, $\mathcal{S}$ then solves $GDH(A, B)$ by computing equation 5.

$$V^* = (d_{s_\sigma^*} + k)s_{k^*}G = (y_\sigma a + h_{s_\sigma^*}s + k)bG = y_\sigma abG + (h_{s_\sigma^*}s + k)B \quad (4)$$

$$CDH(A, B) = y_\sigma^{-1}[V^* - (h_{s_\sigma^*}s + k)B] \quad (5)$$

The probability of the event that $\mathcal{S}$ fails the simulation is analyzed as follows:

First, the E1, E2, E3 events in this simulation is the same as those in the simulation of lemma 1. Second, the E4 event is that $\mathcal{A}$ queries Corrupt-SK oracle with id_{i^*}, id_{j^*}, or id_{k^*}. The probability is $Pr[E4] \leq 1 - 2(q_C - q_{SK})^2(q_{CAP} - q_{AP})/q_C^2 q_{CAP}$.

In summary, $\mathcal{S}$ can solve $GDH(A, B)$ with an advantage

$$Adv_{\mathcal{S}}^{\mathcal{IC}} \geq \frac{2(q_C - q_{PSK})^2(q_{CAP} - q_{AP})\varepsilon_2}{q_C^2 q_{CAP}} (1 - q_{AAE}P_{E2})(1 - \frac{q_{UnAAE}}{|\mathcal{K}|}), \quad (6)$$

where $P_{E2} = max\{\frac{q_2^2}{2|\mathcal{K}|}, \frac{q_1^2}{2q}\}$. Therefore, theorem 1 is proved.

2) PROOF OF THEOREM 2

Label security. First, for our IB-AA scheme, we set the session id sid as (T_A, T_{AP}). We first prove that the session ids are unique.

Lemma 3: The probability for an adversary to generate two point T_A and T_A' which satisfy that $T_A = d_AG + k_AG = d_A'G + k_A'G = T_A'$ is negligible.

Proof. We can conclude from lemma 3 that $k_A = d_A' - d_A + k_A'$. Since k_A and k_A' are the output of H_1 oracle, the probability that the equation holds is $1/q$, which is negligible. So is it for T_{AP}. Thus, it is negligible for an adversary to generate the same session id. That is, the probability of the event that at least three sessions have the same session identity is negligible, and both the client and AP can't be initiator or responder in one session.

Lemma 4: The probability that two session keys of two different sessions (T_A, T_{AP}) and (T_A', T_{AP}') are the same is negligible.

Proof. First, both calculation methods of the session key on each side are $(K_1, K_2) \leftarrow H_3(CDH(T_A, T_{AP}), T_A, T_{AP})$. We could judge that the session keys generated on each side are the same. Second, according to lemma 3 the $sid \neq sid'$

with overwhelming probability. In this case, the session keys are the same with negligible probability $1/|\mathcal{K}|$ according to the randomness of hash function H_3.

In summary, our IB-AA scheme satisfies label security.

ASK security. The demonstrations that our protocol enjoys impersonation security and ASK indistinguishability are shown as follows separately.

Impersonation security. Given $A = aG, B = bG \in E$, $\mathcal{S}$ is able to break $GDH(A, B)$ assumption with non-negligible probability if the adversary breaks impersonation security with non-negligible probability ε_3.

Initialization: $\mathcal{S}$ selects system parameters *params* as specification, and keeps PKG's private key s secretely. $\mathcal{S}$ selects two random numbers $i^* \in_R \{1, 2, \ldots, q_C\}$ and $j^* \in_R \{1, 2, \ldots, q_{CAP}\}$.

Query Phase: $\mathcal{S}$ simulates H_1, H_2, Authentication oracles as specification, and simulates other oracles as follows.

Create oracle: $\mathcal{S}$ maintains a counter i that is initiated to be zero. Upon receiving a Create query id, $\mathcal{S}$ computes $i = i+1$, and checks whether $i = i^*$. If $i \neq i^*$, $\mathcal{S}$ runs *Registration* phase as specification, $\mathcal{S}$ stores the tuple (id_i, R_i, d_i) into list $\mathcal{L}_{Key}$; if $i = i^*$, $\mathcal{S}$ sets $R_i = A$, and stores the tuple $(id_i, R_i, \perp)$ into list $\mathcal{L}_{Key}$. $\mathcal{S}$ stores id_i into $\mathcal{L}_{honest}$.

Create-AP oracle: $\mathcal{S}$ maintains a counter j that is initiated to be zero. Upon receiving a Create query id, $\mathcal{S}$ computes $j = j+1$, and checks whether $j = j^*$. If $j \neq j^*$, $\mathcal{S}$ runs *Initialization* phase as specification, $\mathcal{S}$ stores the tuple (id_j, P_j, s_j) into list $\mathcal{L}_{Key}$; if $j = j^*$, $\mathcal{S}$ sets $P_j = B$, and stores the tuple $(id_j, P_j, \perp)$ into list $\mathcal{L}_{Key}$. $\mathcal{S}$ stores id_j into $\mathcal{L}_{honest}$.

Corrupt-SK oracle: Upon receiving query id_i, if $id_i \neq id_i^*$, $\mathcal{S}$ runs as specification; otherwise, $\mathcal{S}$ aborts.

Corrupt-AP oracle: Upon receiving query id_j, if $id_j \neq id_j^*$, $\mathcal{S}$ runs as specification; otherwise, $\mathcal{S}$ aborts.

Challenge: The adversary chooses (id_s^*, id_{AP}^*) as the targeted client and the targeted AP, where $id_s^*, id_{AP}^* \in \mathcal{L}_{honest}$. If $id_s^* \neq id_{i^*}$ or $id_r^* \neq id_{j^*}$, $\mathcal{S}$ aborts. Assume that the adversary aims to simulate the target client, $\mathcal{S}$ executes as follows. Note that, the probability for the adversary to simulate the target AP is the same as that of this event.

Upon receiving query $(Start, id_{AP}^*)$, $\mathcal{S}$ chooses $k'_{AP} \in_R Z_q^*$, and returns $T_{AP} = d_{AP}G + k_{AP}G$. Upon receiving query (id_{AP}^*, T_A), $\mathcal{S}$ goes through all the queries in $\mathcal{L}_{H_3}$, and checks whether there exists a tuple (V^*, T_A, T_{AP}) that satisfies $V^* = CDH(T_A, T_{AP})$ with the help of DDH oracle. If the tuple exists, $\mathcal{S}$ gets the corresponding output from $\mathcal{L}_{H_3}$, and computes $C_{AP} = Enc_{K_1}(k_{AP})$. Otherwise, $\mathcal{S}$ selects a random session key $K_1 \in_R \mathcal{K}$, and returns the ciphertext $C_{AP} = Enc_{K_1}(k_{AP})$. Upon receiving other queries, $\mathcal{S}$ runs as specification.

Test: Assume that the adversary $\mathcal{A}$ has completed the scheme. As a result, $\mathcal{A}$ is able to get the corresponding session key. $\mathcal{A}$ generates the same K_1 as that generated by $\mathcal{S}$ with negligible probability $1/|\mathcal{K}|$. Therefore, $\mathcal{A}$ queries H_3 oracle with the correct $V^* = CDH(T_A, T_{AP})$ with overwhelming probability. According to equation 7, $\mathcal{S}$ is able to solve $GDH(A, B)$ by computing equation 8.

$$\begin{aligned} V^* &= (d_{i^*} + k_A)(s_{j^*} + k_{AP})G \\ &= abG + k_{AP}A + (h_{i^*}s + k_A)B \\ &\quad +(h_{i^*}s + k_A)k_{AP}G \end{aligned} \tag{7}$$

$$\begin{aligned} CDH(A, B) &= V^* - k_{AP}A - (h_A s + k_A)B \\ &\quad -(h_{i^*}s + k_A)k_{AP}G \end{aligned} \tag{8}$$

In the same way as the proof in lemma 1, we can conclude that the probability that if the adversary wins the game with non-negligible probability, $\mathcal{S}$ solves $GDH(A, B)$ assumption will be non-negligible.

ASK indistinguishability. Given $A = aG, B = bG \in E$, $\mathcal{S}$ is able to break $GDH(A, B)$ assumption with non-negligible probability if the adversary breaks ASK indistinguishability with non-negligible advantage ε_4.

Setup: $\mathcal{S}$ selects system parameters *params* as specification, and keeps PKG's private key s secretely. $\mathcal{S}$ randomly chooses three random numbers $i^*, j^* \in \{1, 2, \ldots, q_C\}$ and $k^*\{1, 2, \ldots, q_{CAP}\}$, and two random numbers $y_0, y_1 \in_R Z_q^*$.

Query Phase: $\mathcal{S}$ simulates H_1, H_2, Authentication oracles as specification, and simulates other oracles as follows.

Create oracle: $\mathcal{S}$ maintains a counter i that is initiated to be zero. Upon receiving a Create query id, $\mathcal{S}$ computes $i = i+1$, and checks whether $i = i^*$. If $i \neq i^*$, $\mathcal{S}$ runs *Registration* phase as specification, $\mathcal{S}$ stores the tuple (id_i, R_i, d_i) into list $\mathcal{L}_{Key}$; if $i = i^*$, $\mathcal{S}$ sets $R_i = y_0A$, and stores the tuple $(id_i, R_i, \perp)$ into list $\mathcal{L}_{Key}$; if $i = j^*$, $\mathcal{S}$ sets $R_i = y_1A$, and stores the tuple $(id_i, R_i, \perp)$ into list $\mathcal{L}_{Key}$. $\mathcal{S}$ stores id_i into $\mathcal{L}_{honest}$.

Create-AP oracle: $\mathcal{S}$ maintains a counter j that is initiated to be zero. Upon receiving a Create query id, $\mathcal{S}$ computes $j = j + 1$, and checks whether $j = k^*$. If $j \neq k^*$, $\mathcal{S}$ runs *Initialization* phase as specification, $\mathcal{S}$ stores the tuple (id_j, P_j, s_j) into list $\mathcal{L}_{Key}$; if $j = k^*$, $\mathcal{S}$ sets $P_j = B$, and stores the tuple $(id_j, P_j, \perp)$ into list $\mathcal{L}_{Key}$. $\mathcal{S}$ stores id_j into $\mathcal{L}_{honest}$.

Corrupt-SK oracle: Upon receiving query id_i, if $id_i \neq id_i^*$ or $id_i \neq id_j^*$, $\mathcal{S}$ runs as specification; otherwise, $\mathcal{S}$ aborts.

Corrupt-AP oracle: Upon receiving query id_j, if $id_j \neq id_k^*$, $\mathcal{S}$ runs as specification; otherwise, $\mathcal{S}$ aborts.

Challenge: The adversary chooses two tuples $(id_{s_0}^*, id_{AP}^*, Start)$ and $(id_{s_1}^*, id_{AP}^*, Start)$ where $id_{s_0}^*, id_{s_1}^*, id_{AP}^* \in \mathcal{L}_{honest}$. The adversary then sends these two tuples to $\mathcal{S}$, and $\mathcal{S}$ selects $\sigma \leftarrow \{0, 1\}$ randomly. If $id_{s_0}^* \neq id_{i^*}$, $id_{s_1}^* \neq id_{j^*}$ or $id_r^* \neq id_{k^*}$, $\mathcal{S}$ aborts. $\mathcal{S}$ sets the targeted client as $id_{s_\sigma}^*$. $\mathcal{S}$ acts as specification of Authentication oracle.

Guess: Assume that $\mathcal{A}$ outputs the right bit σ with advantage ε_4. As a result, $\mathcal{A}$ is able to get the corresponding session key. $\mathcal{A}$ generates the same K_1 as that generated by $\mathcal{S}$ with negligible probability $1/|\mathcal{K}|$. Therefore, $\mathcal{A}$ queries H_3 oracle with the correct $V = CDH(T_A, T_{AP})$ with overwhelming probability. In this case, $\mathcal{S}$ can obtain the targeted V^* from the H_3 query. According to equation 9, $\mathcal{S}$ then solves $GDH(A, B)$ by computing $CDH(A, B) = y_\sigma^{-1}[V^* - y_\sigma k_{AP}A - (h_{s_\sigma^*}s + k_A)B - (h_{s_\sigma^*}s + k_A)k_{AP}G]$.

$$V^* = (d_{s_\sigma^*} + k_A)(s_{k^*} + k_{AP})G$$

TABLE 2. Security comparison.

Objectives	PV	YL	DH	MESS	IB-AA
Impersonation Security	√	√	√	×	√
Anonymity	√	√	×	√	√
Forward Anonymity	×	×	×	√	√
Forward Security	√	√	√	√	√
Scalability	√	√	√	×	√

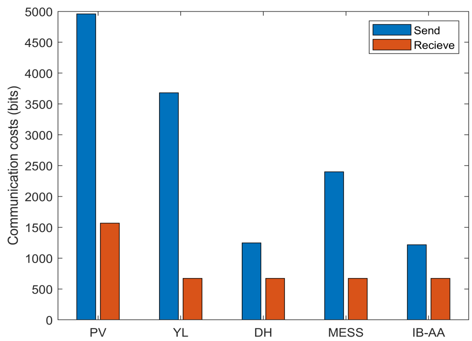

FIGURE 3. Computation costs comparison.

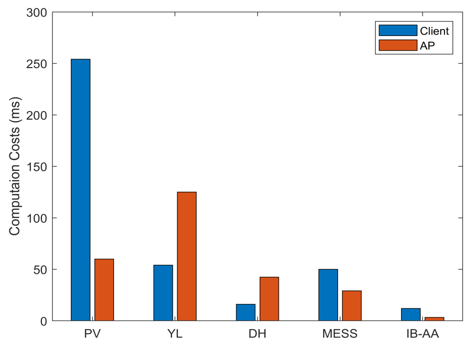

FIGURE 4. Computation costs comparison.

$$= y_\sigma abG + y_\sigma k_{AP}A + (h_{s^*_\sigma} s + k_A)B + (h_{s^*_\sigma} s + k_A)k_{AP}G \quad (9)$$

In the same way as the proof in lemma 2, we can conclude that the simulation is perfect. Thus, if the adversary wins the game with a non-negligible advantage, $\mathcal{S}$ solves $GDH(A, B)$ assumption also with non-negligible probability.

In summary, our IB-AA scheme is label secure, impersonation secure, and ASK indistinguishable. Therefore, our IB-AA scheme is strongly IB-AA secure.

VI. COMPARISON WITH COMPETITIVE SCHEMES

Detailed performance analysis of our proposed scheme is given in this section. Our scheme is also compared with four previous anonymous authentication schemes in terms of security, storage overhead, communication overhead, and computation overhead. We denote "PV", "YL", "DH", and "MESS" by previous schemes of [12], [33], [34], and [17] respectively.

TABLE 3. Computation units.

Operations	Clients(ms)	Application provider(ms)
Double point (G_1)	4.00	0.83
Bilinear Pairing	96.00	20.00
Double point (G_2)	30.00	6.66

First, comparison results of security objectives, which have been described in section III-E, are shown in Table 2. "√" is used to indicate that the corresponding objective is achieved; "×" is used to refer that this objective isn't achieved; "—" indicates this objective isn't considered in this scheme. In the PV scheme, the PKG generates an anonymous identity for each client every period of time, and the client sends this anonymous identity in plaintext. Once the client sends two messages during one period, the forward anonymity of the client cannot be reached. Reference [16] has shown that the DH scheme is vulnerable to key-replacement attacks of the AP. Therefore, once the AP's public key has been replaced, clients in the DH scheme cannot hold anonymity and forward anonymity objectives. The MESS scheme suffers from a forgery attack according to [33], and thereby, it cannot achieve impersonation security. In both MESS scheme and YL scheme, the forward anonymity of the AP hasn't been considered, since once an adversary compromises the AP's private key, it can decrypt and authenticate any message whose receiver is the AP. On the contrary, according to the security proofs in section V, our scheme reaches all objectives in Table 2. Moreover, the MESS scheme needs the AP to build a table for clients' identities and indexes so that the AP could identify an index from a message the AP received. While, in other schemes (including our scheme), there is no need to build or store any index. Therefore, MESS cannot achieve scalability.

Our scheme is implemented with PBC library. Concretely, we simulate the AP through a machine with i5-6500 3.20 GHz 8G RAM; we simulate a client through a machine with Intel PXA270 624-MHz. In this implementation, we set p as a 512-bit-length prime, and we build on F_p an elliptic curve $E: y^2 = x^3 + x \mod p$. We choose an additive group G_1 with 160-bit-length prime order q, and 512-bit-length generator G on curve E. We set bilinear pairing $e : G_1 \times G_1 \rightarrow G_2$, and set the length of elements in G_2 as 1024 bits. According to previous works, the length of a WBAN client's identity, a MAC value, a timestamp, and a "right" value are 32, 160, 32, and 160 bits separately. Execution times of the basic operations, namely double point in G_1, bilinear Pairing, and double point in G_2, are shown in Table 3.

Due to the resource limitation of WBAN clients, storage overhead a client needs is an important factor when comparing WBAN schemes. Storage overhead of an AP determines whether a scheme is scalable or not. Therefore, in this paper, we provide storage comparisons on both client and AP sides in Table 4. As for the computing method of storage overhead, we take PV for example. In PV, a client needs to store his/her public-private key pair, temporary identity, and

TABLE 4. Storage overheads.

Schemes	Clients(bits)	Application provider(bits)
PV	2048	2368
YL	2048	2368
DH	672	672
MESS	1184	672+160n
IB-AA	672	672

tracking parameter all of which are elements in G_1. Thereby, the total storage overhead is $4 \times 512 = 2048$ bits. It is needed for an AP to store its four elements in G_1 (namely its public-private key pair, temporary identity, and tracking parameter) and two elements in Z_q^* (namely two secret keys). Thus, the total storage overhead on the AP side is $4 \times 512 + 2 \times 160 = 2368$ bits. We denote by n the number of clients in Table 4. It is apparent that storage overheads in our scheme is 67.2% and 71.6% less than that in the PV and YL schemes on client and AP sides separately, is 43.2% less than in the MESS scheme on the client side, and is equal to that in the DH scheme. This indicates that our scheme is scalable and practical in terms of storage overheads.

Comparisons of computation and communication overheads are given in Fig. 4 and Fig. 3 respectively. Specifically, we present the computation time required by a client or an AP to execute a scheme in Fig. 4 through "Client" and "AP" items; we provide the length of messages that a client sends and receives in Fig. 3 through "Send" and "Receive" items. Take PV for example. Average computation time of a client to execute a scheme once in PV is $8 \times 4.00 + 30.00 + 2 \times 96.00 = 254.00$ ms, since it requires computing eight double point operations in G_1, one double point operation in G_2, and two bilinear pairing operations. Average computation time of an AP to execute a scheme once in PV is $3 \times 29.00 = 60.00$ ms, since it requires three bilinear pairing operations. In the same way, the computation time of the client and the AP in YL, DH, MESS and our scheme are 54.00 and 125.00 ms, 16.00 and 42.40 ms, 50.00 and 29.15 ms, 12.00 and 3.30ms respectively. After finishing a scheme in PV successfully, a clients sends seven G_1 elements, two elements in Z_q^*, an element in G_2, and a timestamp, which are $7 \times 512 + 2 \times 160 + 1024 + 32 = 4960$ bits; a client receives three elements in G_1 and a timestamp that are $3 \times 512 + 32 = 1568$ bits. In the same way, the communication costs a client sends and receives in YL, DH, MESS and our scheme are 3680 and 672 bits, 1248 and 672 bits, 2400 and 672 bits, 1216 and 672 bits respectively.

According to Fig. 4, we can conclude that our scheme needs 95.3% and 94.5%, 77.8% and 97.4%, 25% and 92.2%, 76% and 88.7% less computation overheads than PV, YL, DH, and MESS on both sides respectively, since our scheme does not require bilinear pairing operations. According to Fig. 3, the message a client needs to send in our proposed scheme is 75.5%, 67.0%, 2.6%, and 49.3% less than that in PV, YL, DH, and MESS, and the message a client needs to receive is 57.1% less than that in PV, and is the same with that in other schemes. This indicates that our scheme is efficient and practical in terms of communication and computation overheads.

VII. CONCLUSION

In this paper, an identity-based pairing-free anonymous authenticated encryption scheme (IB-AAE) has been proposed. An identity-based anonymous authentication scheme (IB-AA) in WBAN have been proposed based on IB-AAE, where forward anonymity can be reached without bilinear pairing. Both IB-AAE and IB-AA have been proved to be secure in the random oracle model. A comprehensive comparison has been conducted to demonstrate that our IB-AA is secure and efficient in terms of computation, communication and storage costs.

REFERENCES

[1] F. Li, Y. Han, and C. Jin, "Cost-effective and anonymous access control for wireless body area networks," *IEEE Syst. J.*, vol. 12, no. 1, pp. 747–758, Mar. 2018.
[2] A. Arfaoui, A. Kribeche, O. R. M. Boudia, A. Ben Letaifa, S. M. Senouci, and M. Hamdi, "Context-aware authorization and anonymous authentication in wireless body area networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–7.
[3] L. Lamport, "Password authentication with insecure communication," *Commun. ACM*, vol. 24, no. 11, pp. 770–772, Nov. 1981.
[4] *IEEE Standard for Local and Metropolitan Area Networks*, Standard 802.15.6-2012, 2012.
[5] M. Toorani, "On vulnerabilities of the security association in the IEEE 802.15.6 standard," in *Proc. Financial Cryptogr. Data Secur. (FC)*, vol. 8976. San Juan, Puerto Rico, Jan. 2015, pp. 245–260.
[6] B. A. Alzahrani, A. Irshad, A. Albeshri, K. Alsubhi, and M. Shafiq, "An improved lightweight authentication protocol for wireless body area networks," *IEEE Access*, vol. 8, pp. 190855–190872, 2020.
[7] J. Liu, Z. Zhang, R. Sun, and K. S. Kwak, "An efficient certificateless remote anonymous authentication scheme for wireless body area networks," in *Proc. IEEE ICC*, Jun. 2012, pp. 3404–3408.
[8] Y. Watanabe, N. Yanai, and J. Shikata, "Anonymous broadcast authentication for securely remote-controlling IoT devices," in *Proc. Adv. Inf. Netw. Appl. (AINA)*, vol. 226. Ottawa, ON, Canada, May 2021, pp. 679–690.
[9] L. Chen, J. Li, Y. Lu, and Y. Zhang, "Adaptively secure certificate-based broadcast encryption and its application to cloud storage service," *Inf. Sci.*, vol. 538, pp. 273–289, Oct. 2020.
[10] D. S. Gupta, S. H. Islam, M. S. Obaidat, P. Vijayakumar, N. Kumar, and Y. Park, "A provably secure and lightweight identity-based two-party authenticated key agreement protocol for IIoT environments," *IEEE Syst. J.*, vol. 15, no. 2, pp. 1732–1741, Jun. 2021.
[11] S. Ji, Z. Gui, T. Zhou, H. Yan, and J. Shen, "An efficient and certificateless conditional privacy-preserving authentication scheme for wireless body area networks big data services," *IEEE Access*, vol. 6, pp. 69603–69611, 2018.
[12] D. He, S. Zeadally, N. Kumar, and J.-H. Lee, "Anonymous authentication for wireless body area networks with provable security," *IEEE Syst. J.*, vol. 11, no. 4, pp. 2590–2601, Dec. 2017.
[13] T. Li, Y. Zheng, and T. Zhou, "Efficient anonymous authenticated key agreement scheme for wireless body area networks," *Secur. Commun. Netw.*, vol. 2017, pp. 1–8, Nov. 2017.
[14] M. Kumar and S. Chand, "A lightweight cloud-assisted identity-based anonymous authentication and key agreement protocol for secure wireless body area network," *IEEE Syst. J.*, vol. 15, no. 2, pp. 2779–2786, Jun. 2021.
[15] M. H. Ibrahim, S. Kumari, A. K. Das, M. Wazid, and V. Odelu, "Secure anonymous mutual authentication for star two-tier wireless body area networks," *Comput. Methods Programs Biomed.*, vol. 135, pp. 37–50, Oct. 2016.
[16] M. Kumar, "Cryptanalysis and improvement of anonymous authentication for wireless body area networks with provable security," *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 936, Jan. 2020.

[17] M. E. S. Saeed, Q.-Y. Liu, G. Tian, B. Gao, and F. Li, ''Remote authentication schemes for wireless body area networks based on the Internet of Things,'' *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4926–4944, Dec. 2018.

[18] E. Lara, L. Aguilar, and J. A. Garcia, ''Lightweight authentication protocol using self-certified public keys for wireless body area networks in health-care applications,'' *IEEE Access*, vol. 9, pp. 79196–79213, 2021.

[19] Y. Zhao, ''Identity-concealed authenticated encryption and key exchange,'' in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 1464–1479.

[20] Y. Zheng, ''Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption),'' in *Proc. Adv. Cryptol. (CRYPTO)*, vol. 1294. Santa Barbara, CA, USA, Aug. 1997, pp. 165–179.

[21] D. Dharminder, M. S. Obaidat, D. Mishra, and A. K. Das, ''SFEEC: Provably secure signcryption-based big data security framework for energy-efficient computing environment,'' *IEEE Syst. J.*, vol. 15, no. 1, pp. 598–606, Mar. 2021.

[22] Y. Cai, H. Zhang, and Y. Fang, ''A conditional privacy protection scheme based on ring signcryption for vehicular ad hoc networks,'' *IEEE Internet Things J.*, vol. 8, no. 1, pp. 647–656, Jan. 2021.

[23] S. Mandal, B. Bera, A. K. Sutrala, A. K. Das, K. R. Choo, and Y. Park, ''Certificateless-signcryption-based three-factor user access control scheme for IoT environment,'' *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3184–3197, Apr. 2020.

[24] C. Peng, J. Chen, M. S. Obaidat, P. Vijayakumar, and D. He, ''Efficient and provably secure multireceiver signcryption scheme for multicast communication in edge computing,'' *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6056–6068, Jul. 2020.

[25] J. Malone-Lee, ''Identity-based signcryption,'' *IACR Cryptol. ePrint Arch.*, vol. 2002, p. 98, Jan. 2002.

[26] X. Boyen, ''Multipurpose identity-based signcryption (a Swiss army knife for identity-based cryptography),'' in *Proc. 23rd Annu. Int. Cryptol. Conf.* in Lecture Notes in Computer Science, vol. 2729, D. Boneh, Ed. Santa Barbara, CA, USA: Springer, 2003, pp. 383–399.

[27] Y. Zhao, ''Identity-concealed authenticated encryption and key exchange,'' *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 1165, Jan. 2018.

[28] H. Wang and Y. Zhao, ''Identity-based higncryption,'' *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 106, Jan. 2019.

[29] Y. Zhao, ''Identity-based authenticated encryption with identity confidentiality,'' in *Computer Security—ESORICS 2020*, vol. 12309, Guildford, U.K.: Springer, Sep. 2020, pp. 633–653.

[30] C. Li, C. Xu, Y. Zhao, K. Chen, and X. Zhang, ''Certificateless identity-concealed authenticated encryption under multi-KGC,'' in *Proc. Int. Conf. Inf. Secur. Cryptol.*, vol. 12020. Nanjing, China, Dec. 2019, pp. 397–415.

[31] H. Xiong and Z. Qin, ''Revocable and scalable certificateless remote authentication protocol with anonymity for wireless body area networks,'' *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 7, pp. 1442–1455, Jul. 2015.

[32] K. Shim, ''Comments on 'revocable and scalable certificateless remote authentication protocol with anonymity for wireless body area networks,''' *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 81–82, 2020.

[33] Y. Liao, Y. Liu, Y. Liang, Y. Wu, and X. Nie, ''Revisit of certificateless signature scheme used to remote authentication schemes for wireless body area networks,'' *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2160–2168, Mar. 2020.

[34] P. Vijayakumar, M. S. Obaidat, M. Azees, S. H. Islam, and N. Kumar, ''Efficient and secure anonymous authentication with location privacy for IoT-based WBANs,'' *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2603–2611, Apr. 2020.

[35] D. Boneh and M. Franklin, ''Identity-based encryption from the Weil pairing,'' *SIAM J. Comput.*, vol. 32, no. 3, pp. 586–615, 2003.

[36] C. Gentry, ''Certificate-based encryption and the certificate revocation problem,'' in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, vol. 2656. Warsaw, Poland, May 2003, pp. 272–293.

[37] T. Okamoto and D. Pointcheval, ''The gap-problems: A new class of problems for the security of cryptographic schemes,'' in *Proc. Int. Workshop Public Key Cryptogr.*, vol. 1992. Cheju Island, South Korea, Feb. 2001, pp. 104–118.

[38] E. Kristin Lauter and A. Mityagin, ''Security analysis of KEA authenticated key exchange protocol,'' in *Public Key Cryptography (PKC '06)*, vol. 3958, pp. 378–394, New York, NY, USA, Apr. 2006.

[39] P. Rogaway, ''Authenticated-encryption with associated-data,'' in *Proc. 9th ACM Conf. Comput. Commun. Secur. (CCS)*, 2002, pp. 98–107.

[40] G. K. Paterson, T. Ristenpart, and T. Shrimpton, ''Tag size does matter: Attacks and proofs for the TLS record protocol,'' in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, vol. 7073, Seoul South Korea, Dec. 2011, pp. 372–389.

[41] M. Bellare, R. Ng, and B. Tackmann, ''Nonces are noticed: AEAD revisited,'' in *Proc. Annu. Int. Cryptol. Conf.*, vol. 11692. Santa Barbara, CA, USA, Aug. 2019, pp. 235–265.

CHUANG LI received the B.Sc. degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), in 2015, where he is currently pursuing the Ph.D. degree. His research interests include cryptographic protocols and network security.

CHUNXIANG XU (Member, IEEE) received the B.Sc. and M.Sc. degrees in applied mathematics and the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 1985, 2004, and 2004, respectively. She is currently a Professor with the School of Computer Science and Engineering, UESTC. Her research interests include information security, cloud computing security, and cryptography.