Int. J. Inf. Secur. (2017) 16:299–311
DOI 10.1007/s10207-016-0320-6

REGULAR CONTRIBUTION

Efficient identity-based online/offline encryption and signcryption with short ciphertext

Jianchang Lai[1] · **Yi Mu**[1] · **Fuchun Guo**[1]

Published online: 18 February 2016

Abstract The technique of online/offline is regarded as a promising approach to speed up the computation of encryption, because the most part of computation, such as pairing over points on elliptic curve and exponentiation in groups, can be pre-computed in the offline phase without knowing the message to be encrypted and/or recipient's identity. The online phase only requires light computation, such as modular multiplication. In this paper, we propose two novel identity-based online/offline schemes: a full secure identity-based online/offline encryption scheme and an identity-based online/offline signcryption scheme. Compared to the other schemes in the literature, our schemes achieve the shortest ciphertext size in both offline and online phases and demonstrate the best performance in offline computation. Our schemes are applicable to devices with limited computation power. They are proven secure in the random oracle model.

Keywords Identity-based · Online/offline · Encryption · Signcryption

1 Introduction

Identity-based encryption (IBE) was introduced by Shamir in 1984 [19] to eliminate the certificate in public key infrastructure. In an IBE system, each user's public key can be an arbitrary string binding the user's identity, such as an email address or a telephone number. If a new user wants to join the network in a network system based on IBE, there is no need for other users in the network to verify its certificate in order to communicate securely.

The merits of identity-based cryptography have drawn a lot of attraction (e.g., [1,2,4,6,10,22,23]). However, all of these works involve computations such as pairings over points on elliptic cure and exponentiations (point multiplications) in groups. These operations are regarded as the most costly computations in cryptography, which might not be applicable for lightweight devices. An elegant solution to reduce the computational overhead of digital signature schemes was proposed by Even et al. [9], where a signing process was split into two phases. The first phase is called offline phase, which is performed prior to obtaining the message to be signed. The second phase is called online phase, which is executed when the message becomes available. All the heavy computations in the signing phase are precomputed in the offline phase. In the online phase, it only performs the light computations such as modular multiplication and hashing.

The first two identity-based online/offline encryption (IBOOE) schemes were proposed by Guo et al. [11] in 2008. In the offline phase, all the heavy computations are conducted without the requirements of the recipient's identity and the message to be encrypted. When the recipient's identity and the message become available, the online phase can be accomplished with great efficiency. However, the ciphertext size of both IBOOE schemes in [11] is large, so might not be desirable for lightweight devices.

The online/offline encryption schemes are usually formulated with the bilinear maps. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative groups with the same prime order p. A sym-

✉ Jianchang Lai
jl967@uowmail.edu.au

Yi Mu
ymu@uow.edu.au

Fuchun Guo
fuchun@uow.edu.au

[1] Centre for Computer and Information Security Research, School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, Australia

metric bilinear map is defined as $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. The first scheme in [11] needs one element in $\mathbb{G}_T$, four elements in $\mathbb{G}$, two elements in $\mathbb{Z}_p^*$ and two one-time signatures. The second scheme requires four elements in $\mathbb{G}_T$, two elements in $\mathbb{G}$ and two elements in $\mathbb{Z}_p^*$.

Subsequently, an IBOOE scheme, which is more efficient than Guo et al.'s [11], was proposed by Liu and Zhou [15]. They removed the elements in $\mathbb{G}_T$ from the final ciphertext and only needed three modular multiplications operations in the online phase. Later, Chow et al. [7] proposed the first identity-based online/offline key encapsulation (IBOOKEM). Based on their IBOOKEM, they proposed a secure IBOOE scheme in the random oracle model. Compared to [15], they saved two elements in $\mathbb{G}$ in the ciphertext. However, Selvi et al. [17,18] showed that the schemes in [15] and [7] are not secure against the chosen ciphertext attack (CCA). A practical IBOOE scheme for wireless sensor network was proposed by Chu et al. [8], which removed the elements in $\mathbb{G}_T$ from the offline storage. However, the security, storage and computation of the proposed scheme are depended on a symmetric encryption scheme.

A provably CCA secure (in the random oracle model) and efficient IBOOE scheme was proposed by Selvi et al. [17,18]. Their IBOOE scheme requires more than one element in group $\mathbb{G}$ in both online and offline phase. Recently, an efficient semi-generic transformation to achieve IBOOE from IBE was proposed by Lai et al. [12] in ACISP 2015. Due to the space limitation, they only proposed a key encapsulation scheme against the chosen plaintext attack.

In this paper, we propose an efficient identity-based online/offline encryption scheme. Our proposed IBOOE scheme achieves the shortest ciphertext, in comparison with other schemes in the literature. Compared to the best-known scheme in the literature, our scheme saves one element in $\mathbb{G}$ in both offline and online phases. We only require one element in $\mathbb{G}$ in both offline storage and ciphertext size in our scheme. In the offline phase, our scheme reduces one group operation in $\mathbb{G}$. Our proposed scheme is proven secure in the random oracle model.

The first identity-based online/offline signcryption (IBOOSC) scheme was proposed by Sun et al. [20] in 2008. They for the first time gave the definition of IBOOSC and its security model. Then, a generic construction of identity-based online/offline signcryption was given by Sun et al. [21]. However, Selvi et al. [18] pointed out that the schemes in [20] and [21] actually are not secure against the existential forgery attack.

To improve the security, Liu et al. [14] proposed a secure IBOOSC scheme. In [14], the final ciphertext still needs four elements in $\mathbb{G}$. To speed up the online computation, Selvi et al. [18] proposed a new secure IBOOSC scheme and reduced one multiplication operation in the online phase compared to [14]. But their scheme requires a larger offline storage and the ciphertext is bigger than [14]. Li et al. [13] proposed an efficient IBOOSC scheme based on [1] for low power devices. Their scheme requires three elements in $\mathbb{G}$ in both the offline storage and the final ciphertext.

Apart from our novel online/offline identity-based encryption scheme, in this paper, we also propose a novel and efficient identity-based online/offline signcryption scheme based on [1]. Our scheme achieves the shortest ciphertext, compared to the existing IBOOSC schemes. Compared to [13], our scheme saves one element in $\mathbb{G}$ in the final ciphertext and saves one element in $\mathbb{G}$ and one element in $\mathbb{Z}_p^*$ in the offline storage. In the offline phase, our signcryption scheme reduces one operation in $\mathbb{G}$. The online computation is comparable and only requires two modular operations.

Organization The remainder of the paper is organized as follows. In Sect. 2, we review some basic preliminaries including bilinear pairing, complexity assumptions, definitions and the corresponding security models. We describe our novel identity-based online/offline encryption scheme with the shortest ciphertext and show its security in Sect. 3. In Sect. 4, we propose an efficient identity-based online/offline signcryption scheme and give its security. In Sect. 5, we give the evaluation and comparison of our schemes and draw our conclusions in Sect. 6.

2 Preliminaries

2.1 Bilinear pairing

Let $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups of the same prime order p. g is a generator of $\mathbb{G}$. A bilinear pairing is a map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties:

1. Bilinear: For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p^*$, we have $e\left(u^a, v^b\right) = e\left(u, v\right)^{ab}$.
2. Non-degeneracy: $e\left(g, g\right) \neq 1$.
3. Computability: It is efficient to compute $e\left(u, v\right)$ for all $u, v \in \mathbb{G}$.

2.2 Complexity assumptions

The computational assumptions for the security of our schemes are based on the following three assumptions. Let $\mathbb{G}$ and $\mathbb{G}_T$ be multiplicative cyclic groups of the same prime order p, g be a generator of $\mathbb{G}$ and e be a pairing map. We review the k-collision attack assumption 1 (k-CAA1) [5], the q-bilinear Diffie–Hellman inversion (q-BDHI) [2] assumption and the q-strong Diffie–Hellman (q-SDH) [3] assumption.

Definition 1 (k-**CAA1**) Given $\left(g, g^a, c_0, \left(c_1, g^{\frac{1}{c_1+a}}\right), \ldots, \left(c_k, g^{\frac{1}{c_k+a}}\right)\right)$ where $c_i \in_R \mathbb{Z}_p^*$ and distinct for $0 \leq i \leq k$, the k-CAA1 problem is to compute $e(g, g)^{\frac{1}{c_0+a}}$.

Definition 2 (q-**BDHI**) Given $(q+1)$-tuple $\left(h, h^a, h^{a^2}, \ldots, h^{a^q}\right) \in \mathbb{G}^{q+1}$ where $a \in_R \mathbb{Z}_p^*$, the q-BDHI problem in $(\mathbb{G}, \mathbb{G}_T)$ is to compute $e(h,h)^{\frac{1}{a}}$.

Definition 3 (q-**SDH**) Given $(q+1)$-tuple $\left(h, h^a, h^{a^2}, \ldots, h^{a^q}\right) \in \mathbb{G}^{q+1}$ where $a \in_R \mathbb{Z}_p^*$, the q-SDH problem in $\mathbb{G}$ is to find a pair $\left(c, h^{\frac{1}{a+c}}\right)$ with $c \in \mathbb{Z}_p^*$.

2.3 Definition of identity-based online/offline encryption

An identity-based online/offline encryption scheme consists of the following five algorithms:

Setup(k): Taking as input a security parameter $k \in \mathbb{Z}^+$, it outputs the system parameters mpk and the master key msk, where the system parameters are publicly known, while the master key is kept secretly and known to generator (PKG) only.

KeyGen(mpk, msk, ID): Taking as input the system parameters mpk, master key msk and an arbitrary $ID \in \{0,1\}^*$, it outputs a private key d_{ID} for ID.

Off-Encrypt(mpk): Taking as input the system parameters mpk, it outputs the offline ciphertext C_{off}.

On-Encrypt(mpk, C_{off}, ID, m): Taking as input the system parameters mpk, offline ciphertext C_{off}, an identity $ID \in \{0,1\}^*$ and a message m, it outputs a ciphertext CT.

Decrypt(mpk, CT, d_{ID}): Taking as input the system parameters mpk, ciphertext CT and the private key d_{ID}, it outputs a plaintext message or a rejected symbol.

The security for IBOOE is the same as that for identity-based encryption. The strongest and commonly accepted notion of security for identity-based encryption system is that of indistinguishability against an adaptive chosen ciphertext attack. The game of IND-ID-CCA is defined under the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Both take the security parameter $k \in \mathbb{Z}^+$ as input.

Setup: $\mathcal{C}$ takes as input a security parameter k and runs the **Setup** algorithm to obtain the system parameters mpk and the master key msk. It gives the adversary $\mathcal{A}$ the system parameters mpk.

Phase 1: $\mathcal{A}$ adaptively makes a polynomial number of following queries:

- *Private key query* (ID_i). $\mathcal{C}$ responds by running key generation algorithm **KeyGen** to generate the private key d_{ID_i} and sending d_{ID_i} to $\mathcal{A}$.
- *Decryption query* (ID_i, CT_i). $\mathcal{C}$ responds by running key generation algorithm **KeyGen** to generate the private key d_{ID_i}, running the algorithm **Decrypt** to decrypt the ciphertext (ID_i, CT_i) and sending the result back to $\mathcal{A}$.

Challenge: $\mathcal{A}$ outputs two messages m_0, m_1 and a challenged identity ID^* under one restriction that $\mathcal{A}$ did not request a private key for ID^* in **Phase 1**. $\mathcal{C}$ picks a random bit $b \in \{0,1\}$, computes a challenge ciphertext CT^* on m_b and then sends CT^* to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ continues to issue private key queries on $ID_i \neq ID^*$ and decryption queries on $(ID_i, CT_i) \neq (ID^*, CT^*)$. $\mathcal{C}$ responds as in **Phase 1**.

Guess: Finally, $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$ for b and wins the game if $b' = b$.

We refer to such an adversary $\mathcal{A}$ as an IND-ID-CCA adversary. We define adversary $\mathcal{A}$'s advantage in attacking the game as

$$\mathsf{Adv}_{\mathcal{A}}(k) = \left|\Pr[b = b'] - \frac{1}{2}\right|.$$

Definition 4 An identity-based online/offline encryption scheme is IND-ID-CCA secure if for any polynomial time IND-ID-CCA adversary the $\mathsf{Adv}_{\mathcal{A}}(k)$ is negligible.

2.4 Definition of identity-based online/offline signcryption

An signcryption scheme should provide confidentiality as well as authentication and non-repudiation. The formal structure of identity-based online/offline signcryption (IBOOS) scheme consists of five algorithms that are modeled as follows:

Setup(k): The same as IBOOE.

KeyGen(mpk, msk, ID): The same as IBOOE.

Off-Signcrypt(mpk, ID_s, d_{ID_s}): Taking as input the system parameters mpk, sender's identity ID_s and the sender's private key d_{ID_s}, it outputs offline ciphertext δ.

On-Signcrypt(mpk, m, ID_s, ID_r, δ): Taking as input a message m, a sender's identity ID_s, a receiver's identity ID_r and the offline output δ, it outputs the final signcryption σ.

Unsigncrypt(mpk, σ, ID_s, ID_r, d_{ID_r}): Taking as input a signcryption σ, a sender's identity ID_s, a receiver's identity ID_r and the receiver's private key d_{ID_r}, it outputs the plaintext message or a rejected symbol.

Signcryption achieves both encryption and digital signature. We first give the security for message confidentiality. The notion of IND-ID-CCA for IBOOSC is similar to that for IBOOE and is defined under the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup: $\mathcal{C}$ takes as input a security parameter k and runs the **Setup** algorithm to obtain the system parameters mpk and the master key msk. It gives the adversary $\mathcal{A}$ the system parameters mpk.

Phase 1: $\mathcal{A}$ adaptively makes a polynomial number of following queries:

– *Private key query* (ID_i). $\mathcal{C}$ responds by running key generation algorithm **KeyGen** to generate the private key d_{ID_i} and sending d_{ID_i} to $\mathcal{A}$.
– *Signcryption query* (ID_s, ID_r, m). $\mathcal{C}$ responds by running key generation algorithm **KeyGen** to generate the sender's private key d_{ID_s}, running the algorithm **Off-Signcrypt** to obtain δ and sending σ by running **On-Signcrypt** to $\mathcal{A}$.
– *Unsigncryption query* (σ, ID_s, ID_r). $\mathcal{C}$ runs **KeyGen** to generate the receiver's private key d_{ID_r} and sends the result of **Unsigncrypt** to $\mathcal{A}$.

Challenge: $\mathcal{A}$ outputs two messages m_0, m_1 and challenged identities $\left(ID_s^*, ID_r^*\right)$ under one restriction that $\mathcal{A}$ did not issue a private key query for ID_r^* in **Phase 1**. $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$ and computes a challenge ciphertext σ^* on m_b, then sends σ^* to $\mathcal{A}$.
Phase 2: $\mathcal{A}$ issues more private key queries on $ID_i \neq ID_r^*$ and unsigncryption queries on $\sigma_i \neq \sigma^*$. $\mathcal{C}$ responds as in **Phase 1**.
Guess: Finally, $\mathcal{A}$ outputs its guess $b' \in \{0, 1\}$ for b and wins the game if $b' = b$.

We refer to such an adversary $\mathcal{A}$ as an IND-ID-CCA adversary. We define adversary $\mathcal{A}$'s advantage in attacking the game as

$$\mathsf{Adv}_{\mathcal{A}}(k) = \left|\Pr[b = b'] - \frac{1}{2}\right|.$$

Definition 5 An identity-based online/offline signcryption scheme is IND-ID-CCA secure if for any polynomial time IND-ID-CCA adversary the $\mathsf{Adv}_{\mathcal{A}}(k)$ is negligible.

The next game defines the security for non-repudiation with regard to signatures embedded in ciphertexts. The commonly accepted notion of security for signature system is existentially unforgeable against adaptive chosen message attacks (EUF-CMA). This notion is defined by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.
Setup: $\mathcal{C}$ takes as input a security parameter k and runs the **Setup** algorithm to obtain the system parameters mpk and the master key msk. It gives the adversary $\mathcal{A}$ the system parameters mpk.
Private key query(ID_i). $\mathcal{C}$ responds as in the IND-ID-CCA game for IBOOSC.
Signcryption query (ID_s, ID_r, m). $\mathcal{C}$ responds as in the IND-ID-CCA game for IBOOSC.
Unsigncryption query (σ, ID_s, ID_r). $\mathcal{C}$ responds as in the IND-ID-CCA game for IBOOSC.
Forge: Finally, $\mathcal{A}$ outputs a triple $\left(\sigma^*, ID_s^*, ID_r^*\right)$ where the private key of ID_r^* has not been queried. $\mathcal{A}$ wins the game if σ^* is a valid ciphertext corresponding to the $\left(ID_s^*, ID_r^*\right)$ and passes the verification. The adversary's advantage is its probability to win the game.

Definition 6 An identity-based online/offline signcryption scheme is EUF-CMA secure if there is no probability polynomial time adversary $\mathcal{A}$ who can win this game with a non-negligible advantage.

3 Identity-based online/offline encryption

In this section, we give our efficient identity-based online/offline encryption scheme and the corresponding security analysis.

3.1 Construction

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic groups of the same prime order p, and g is the generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be the bilinear pairing and n be the length of the message. The scheme is described by the following five algorithms.
Setup: The system parameters are generated as follows. The PKG randomly chooses $\alpha \in \mathbb{Z}_p^*$, computes $v = e(g, g)$ and sets $g_1 = g^\alpha$. Let $H_1 : \{0, 1\}^* \to \mathbb{Z}_p^*$, $H_2 : \mathbb{G}_T \to \{0, 1\}^n$, $H_3 : \mathbb{G}_T \times \{0, 1\}^n \times \mathbb{Z}_p^* \times \mathbb{G} \to \mathbb{Z}_p^*$ be the cryptographic hash functions. The public parameters *mpk* and the master key *msk* are

$$mpk = (\mathbb{G}, \mathbb{G}_T, p, g, g_1, v, n, e, H_1, H_2, H_3), \quad msk = \alpha.$$

KeyGen: A user's private key generation algorithm proceeds as follow. For an identity $ID \in \{0, 1\}^*$, the PKG computes

$$d_{ID} = g^{\frac{1}{\alpha + H_1(ID)}}.$$

Off-Encrypt: The offline storage is computed as follows. The PKG randomly chooses $s, w \in \mathbb{Z}_p^*$ and computes

$$R = v^s, \quad C_1 = \left(g_1 g^w\right)^s, \quad h = H_2(R).$$

The offline storage is $C_{off} = (C_1, R, s, w, h)$.
On-Encrypt: To encrypt a message $m \in \{0, 1\}^n$ for $ID \in \{0, 1\}^*$, the sender computes

$$C_2 = s\left(H_1(ID) - w\right) \mod p,$$
$$C_3 = h \oplus m,$$
$$C_4 = \left(s + H_3(R, m, C_1, C_2)\right) \mod p.$$

Then output the final ciphertext $CT = (C_1, C_2, C_3, C_4)$.
Decrypt: Given a ciphertext $CT = (C_1, C_2, C_3, C_4)$, the recipient decrypts the ciphertext by using its private key d_{ID} to compute

$$R' = e\left(d_{ID}, C_1 g^{C_2}\right),$$
$$m' = C_3 \oplus H_2\left(R'\right),$$

$$s' = \left(C_4 - H_3\left(R', m', C_1, C_2\right)\right) \mod p,$$
$$w' = \left(H_1(ID) - C_2 s'^{-1}\right) \mod p.$$

Then check whether

$$C_1 \stackrel{?}{=} \left(g_1 g^{w'}\right)^{s'}.$$

If it holds, the recipient outputs m' as the result of decryption, and otherwise rejects it.

Correctness: For a valid ciphertext, we have

$$\begin{aligned} R' &= e\left(d_{ID}, C_1 g^{C_2}\right) \\ &= e\left(g^{\frac{1}{\alpha + H_1(ID)}}, (g_1 g^w)^s g^{s(H_1(ID)-w)}\right) \\ &= e\left(g^{\frac{1}{\alpha + H_1(ID)}}, g_1^s g^{s H_1(ID)}\right) \\ &= e(g, g)^s \\ &= R. \end{aligned}$$

The receiver gets the correct R by using its private key. Since it is a valid ciphertext, after getting correct R, receiver can compute the correct m, s and w, and then passes the checking equation. Thus, receiver can get the valid message.

3.2 Security

We now give the formal proof for the security of our scheme. We have the following result states that our scheme is IND-ID-CCA secure under the k-CAA1 assumption.

Theorem 1 *Suppose the hash functions H_1, H_2 and H_3 are random oracles. Then our proposed scheme is indistinguishability against chosen ciphertext attacks (IND-ID-CCA) under k-CAA1 assumption. Specifically, suppose there is an IND-ID-CCA adversary $\mathcal{A}$ that has advantage ϵ against our proposed scheme. $\mathcal{A}$ makes at most q_E private key queries and at most q_{H_1}, q_{H_2}, q_{H_3} queries to the hash functions H_1, H_2, H_3, respectively. Then there is an algorithm $\mathcal{S}$ to solve the k-CAA1 problem with advantage ϵ', where*

$$\epsilon' \geq \frac{\epsilon}{q_{H_1}\left(q_{H_2} + q_{H_3}\right)}.$$

Proof Suppose $\mathcal{A}$ has advantage ϵ in attacking our scheme. We build a simulator $\mathcal{S}$ that solves k-CAA1 problem with advantage ϵ' by running $\mathcal{A}$. Let $\left(g, g^a, c_0, \left(c_1, g^{\frac{1}{c_1+a}}\right), \ldots, \left(c_k, g^{\frac{1}{c_k+a}}\right)\right)$ be a random instance of k-CAA1 problem taken as input by $\mathcal{S}$ and its goal is to compute $e(g, g)^{\frac{1}{a+c_0}}$. In order to use $\mathcal{A}$ to solve the problem, $\mathcal{S}$ needs to simulate a challenger and responses all the queries for $\mathcal{A}$. Without loss of generality, we assume that $k = q_{H_1} - 1$. Simulator $\mathcal{S}$ works by interacting with $\mathcal{A}$ in an IND-ID-CCA game as follows:

Setup: $\mathcal{S}$ sets $g_1 = g^a$ and randomly chooses an index $I \in \{1, 2, \ldots, q_{H_1}\}$ and then sends the system parameters $(\mathbb{G}, \mathbb{G}_T, p, g, g_1, H_1, H_2, H_3)$ to $\mathcal{A}$.

H_1 **queries**: At any time adversary $\mathcal{A}$ may issue queries to the random oracle H_1 on ID_i. $\mathcal{S}$ maintains a list L_1 of tuples (ID_i, c_i). This list is initially empty. If ID_i is already on the list L_1, $\mathcal{S}$ responds with $H_1(ID_i) = c_i$. Otherwise, $\mathcal{S}$ checks the ID_i. If $ID_i = ID_I$, $\mathcal{S}$ sets $H_1(ID_I) = c_0$, returns c_0 and then adds the new tuple (ID_I, c_0) into L_1. If $ID_i \neq ID_I$, $\mathcal{S}$ randomly chooses c_i from the instance of k-CAA1 which has not been chosen by $\mathcal{S}$ before and sets $H_1(ID_i) = c_i$. Then $\mathcal{S}$ adds the new tuple (ID_i, c_i) into L_1 and returns c_i.

H_2 **queries**: At any time, adversary $\mathcal{A}$ may issue queries to the random oracle H_2 on R_i. $\mathcal{S}$ maintains a list L_2 of tuples (R_i, β_i). This list is initially empty. If tuple (R_i, β_i) is in the list L_2, $\mathcal{S}$ responds with $H_2(R_i) = \beta_i$. Otherwise, $\mathcal{S}$ picks a random string $\beta_i \in \{0, 1\}^n$ where β_i has not been used before and sets $H_2(R_i) = \beta_i$. Then $\mathcal{S}$ returns β_i and adds the new tuple (R_i, β_i) into L_2.

H_3 **queries**: At any time, adversary $\mathcal{A}$ may issue queries to the random oracle H_3 on $\left(R_i, m_i, C_1^i, C_2^i\right)$. $\mathcal{S}$ maintains a list L_3 of tuples $\left(R_i, m_i, C_1^i, C_2^i, \gamma_i\right)$. This list is initially empty. If $\left(R_i, m_i, C_1^i, C_2^i\right)$ is on the list, $\mathcal{S}$ responds with γ_i. Otherwise, $\mathcal{S}$ randomly chooses $\gamma_i \in \mathbb{Z}_p^*$ which does not exist in L_3. Then $\mathcal{S}$ sets $H_3\left(R_i, m_i, C_1^i, C_2^i\right) = \gamma_i$, responds to $\mathcal{A}$ with γ_i, and adds the new tuple $\left(R_i, m_i, C_1^i, C_2^i, \gamma_i\right)$ to the list L_3.

Phase 1: At any time, adversary $\mathcal{A}$ may issue the following queries:

- *Private key queries*. Let ID_i be a private key query issued by adversary $\mathcal{A}$. $\mathcal{S}$ responds to this query as follows. If $ID_i = ID_I$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ looks up the list L_1 to get the corresponding (ID_i, c_i). If (ID_i, c_i) is not in the list L_1, $\mathcal{S}$ runs the H_1 queries and gets the corresponding c_i and adds the tuple (ID_i, c_i) into L_1. Then $\mathcal{S}$ sets $d_{ID_i} = g^{\frac{1}{\alpha + c_i}}$ and returns d_{ID_i}.
- *Decryption queries*. Consider a decryption query for a ciphertext (ID_i, CT_i) where $CT_i = \left(C_1^i, C_2^i, C_3^i, C_4^i\right)$. Simulator $\mathcal{S}$ first examines the ID_i. If $ID_i \neq ID_I$; since $\mathcal{S}$ knows the private key of ID_i, he can run the decryption algorithm to decrypt the ciphertext correctly by using d_{ID_i}. If $ID_i = ID_I$, $\mathcal{S}$ decrypts the ciphertext as follows:
  1. Search the tuple $\left(R_i, m_i, C_1^i, C_2^i, \gamma_i\right)$ from the list L_3 such that $H_2(R_i) \oplus m_i = C_3^i$. If no match, output "invalid."
  2. Otherwise, compute $s_i = \left(C_4^i - \gamma_i\right) \mod p$ and check whether

$$C_1^i g^{C_2^i} \stackrel{?}{=} \left(g_1 g^{H_1(ID_i)}\right)^{s_i}.$$

If it does not hold, go back to step 1 and test next tuple in L_3.

3. Otherwise, compute $w_i = \left(H_1\left(ID_i\right) - C_2^i s_i^{-1}\right) \bmod q$ and check whether

$$C_1^i \stackrel{?}{=} \left(g_1 g^{w_i}\right)^{s_i}.$$

If it holds, output m_i as the decryption result and return m_i to $\mathcal{A}$, otherwise go back to step 1.

Challenge: Once adversary $\mathcal{A}$ decides **Phase 1** is over, it outputs two messages $m_0, m_1 \in \{0,1\}^n$ and a challenged identity ID^* under one restriction that ID^* has not been queried in the private key queries phase. If $ID^* \neq ID_I$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ picks a random bit $b \in \{0,1\}$ and computes the challenge ciphertext as follows:

1. Randomly choose $\lambda \in \mathbb{Z}_p^*$, $C_2^* \in \mathbb{Z}_p^*$, $C_3^* \in \{0,1\}^n$, $C_4^* \in \mathbb{Z}_p^*$.
2. Compute $C_1^* = g^{\lambda - C_2^*}$ and set $CT^* = \left(C_1^*, C_2^*, C_3^*, C_4^*\right)$.

Phase 2: $\mathcal{A}$ continues issuing more private key queries on $ID_i \neq ID^*$ and decryption queries on $(ID_i, CT_i) \neq (ID^*, CT^*)$. $\mathcal{S}$ responds as in **Phase 1**.
Guess: Finally, $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$ for b.

From the $\mathcal{A}$'s point of view, if it knows that CT^* is a bad challenge ciphertext (invalid), it must have queried the H_2 or H_3 oracle with R^* as input. This means that R^* is one of input of H_2 or H_3 oracle and in the list L_2 or L_3. Otherwise, according to the assumption, $\mathcal{A}$ will with ϵ advantage output the correct b'. This also means that A has queried the value of H_2 or H_3 taken R^* as input. From the decryption algorithm, we have

$$\begin{aligned} R^* &= e\left(d_{ID^*}, C_1^* g^{C_2^*}\right) = e\left(g^{\frac{1}{\alpha + H_1(ID^*)}}, g^{\lambda - C_2^*} g^{C_2^*}\right) \\ &= e(g,g)^{\frac{\lambda}{\alpha + c_0}}. \end{aligned}$$

$\mathcal{S}$ ignores the guess of $\mathcal{A}$ and randomly picks a R from the list L_2 or L_3 and computes $R^{\lambda^{-1}}$ as the solution to the given instance of k-CAA1 problem.

The above completes the description of simulation algorithm $\mathcal{S}$. To complete the security proof, it remains to show that $\mathcal{S}$ correctly outputs $e(g,g)^{\frac{1}{c_0 + a}}$ with probability at least ϵ'. We first analyze the probability that $\mathcal{S}$ does not abort during the simulation. We consider two events below:

1. A_1 : $\mathcal{A}$ did not query the private key on ID^* in private key query phase.
2. A_2 : The challenged identity ID^* is equal to ID_I.

We have

$$\begin{aligned} \Pr[A_1] &= 1 - \frac{q_E}{q_{H_1}}, \\ \Pr[A_2] &= \frac{1}{q_{H_1} - q_E}, \\ \Pr\left[\neg abort\right] &= \Pr[A_1 \wedge A_2] \\ &= \Pr[A_2] \Pr[A_1 | A_2] \\ &= \left(\frac{1}{q_{H_1} - q_E}\right)\left(1 - \frac{q_E}{q_{H_1}}\right) \\ &= \frac{1}{q_{H_1}}, \end{aligned}$$

The probability of R randomly chosen from the list L_2 or L_3 to be the solution of k-CAA1 is at least $\frac{1}{q_{H_2} + q_{H_3}}$. It follows that $\mathcal{S}$ computes a correct answer for k-CAA1 problem by the above simulation with advantage

$$\epsilon' \geq \frac{\epsilon}{q_{H_1}\left(q_{H_2} + q_{H_3}\right)}.$$

This completes the proof.

4 Identity-based online/offline signcryption

In this section, we propose an efficient IBOOSC scheme based on Barreto et al.'s [1] identity-based signcryption scheme and give the security analysis.

4.1 Construction

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic groups of the same prime order p, and g is the generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be the bilinear pairing and n be the length of the message.
Setup: The system parameters are generated as follows. Given a system security parameter k, the PKG randomly chooses $\alpha \in \mathbb{Z}_p^*$, computes $v = e(g,g)$ and sets $g_1 = g^\alpha$. Let $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$, $H_2 : \{0,1\}^n \times \mathbb{G}_T \times \mathbb{G} \times \mathbb{G} \to \mathbb{Z}_p^*$, $H_3 : \mathbb{G}_T \to \{0,1\}^n$ be the cryptographic hash functions. The public parameters *mpk* and the master key *msk* are given by

$$mpk = (\mathbb{G}, \mathbb{G}_T, p, g, g_1, v, n, e, H_1, H_2, H_3), \qquad msk = \alpha.$$

KeyGen: A user's private key generation algorithm proceeds as follow. For identity $ID \in \{0,1\}^*$, the PKG computes

$$d_{ID} = g^{\frac{1}{\alpha + H_1(ID)}}.$$

Off-Signcrypt: The offline storage is computed as follows. The sender with identity ID_s randomly chooses $s, w, \beta \in \mathbb{Z}_p^*$ and computes

$$R = v^s, \quad C_1 = d_{ID_s}^{\beta}, \quad C_2 = \left(g_1 g^w\right)^s.$$

The offline storage is $\delta = (s, \beta, w, R, C_1, C_2)$.
On-Signcrypt: To encrypt and sign a message $m \in \{0,1\}^n$ for a receiver with identity $ID_r \in \{0,1\}^*$, the sender computes

$$\begin{aligned} C_3 &= m \oplus H_3(R), \\ h &= H_2(m, R, C_1, C_2), \\ C_4 &= \beta^{-1}(s+h) \mod p, \\ C_5 &= s\,(H_1(ID_r) - w) \mod p. \end{aligned}$$

Then output the final ciphertext $\sigma = (C_1, C_2, C_3, C_4, C_5)$.
Unsigncrypt: For a ciphertext $\sigma = (C_1, C_2, C_3, C_4, C_5)$, the recipient decrypts the ciphertext by using its private key d_{ID_r} to compute

$$\begin{aligned} T &= C_2 g^{C_5}, \\ S &= C_1^{C_4}, \\ R &= e(T, d_{ID_r}), \\ m &= C_3 \oplus H_3(R), \\ h &= H_2(m, R, C_1, C_2). \end{aligned}$$

Then check whether

$$R = e\left(S, g_1 g^{H_1(ID_s)}\right) v^{-h}.$$

If it holds, output m as the result of decryption, and believe ID_s has signed on m. Otherwise, reject it.
Correctness: For a valid ciphertext, we have

$$\begin{aligned} T = C_2 g^{C_5} &= \left(g_1 g^w\right)^s g^{s(H_1(ID_r)-w)} = \left(g_1 g^{H_1(ID_r)}\right)^s, \\ S = C_1^{C_4} &= \left(d_{ID_s}^{\beta}\right)^{\beta^{-1}(s+h)} = d_{ID_s}^{s+h}, \\ e\left(T, d_{ID_r}\right) &= e\left(\left(g_1 g^{H_1(ID_r)}\right)^s, g^{\frac{1}{\alpha + H_1(ID_r)}}\right) \\ &= e\left(\left(g^{\alpha + H_1(ID_r)}\right)^s, g^{\frac{1}{\alpha + H_1(ID_r)}}\right) \\ &= e(g,g)^s \\ &= R. \end{aligned}$$

Receiver gets the correct R by using its private key. After obtaining the correct R, receiver can compute the correct m and h and pass the checking equation.

$$\begin{aligned} e\left(S, g_1 g^{H_1(ID_s)}\right) v^{-h} &= e\left(d_{ID_s}^{(s+h)}, g^{\alpha + H_1(ID_s)}\right) e(g,g)^{-h} \\ &= e\left(g^{\frac{s+h}{\alpha + H_1(ID_s)}}, g^{\alpha + H_1(ID_s)}\right) e(g,g)^{-h} \\ &= e(g,g)^{s+h} e(g,g)^{-h} \\ &= e(g,g)^s \\ &= R. \end{aligned}$$

4.2 Security

We now give the formal proof for the security of our signcryption scheme. The security follows immediately from the following two theorems. The first theorem states that our scheme is IND-ID-CCA secure under the q-BDHI assumption for the message confidentiality. The second theorem ensures that our scheme is EUF-CMA secure under q-SDH assumption.

Theorem 2 *Suppose that an IND-ID-CCA adversary $\mathcal{A}$ has an advantage ϵ against our scheme, when asking q_{H_i} queries to random oracles H_i $(i = 1, 2, 3)$, q_{sc} signcryption queries and q_{us} queries to the unsigncryption queries. Then there is an algorithm $\mathcal{S}$ to solve the q-BDHI problem for $q = q_{H_1}$ with advantage*

$$\epsilon' \geq \frac{\epsilon}{q_{H_1}\left(2q_{H_2} + q_{H_3}\right)} \left(1 - \frac{q_{sc}\left(q_{sc} + q_{H_2}\right)}{2^k}\right)\left(1 - \frac{q_{us}}{2^k}\right).$$

Proof Suppose the adversary $\mathcal{A}$ has advantage ϵ in attacking our scheme. We build a simulator $\mathcal{S}$ that solves q-BDHI problem with advantage ϵ' by running $\mathcal{A}$. Let $\left(h, h^a, h^{a^2}, \ldots, h^{a^q}\right)$ be a random instance of q-BDHI problem taken as input by $\mathcal{S}$. Its goal is to compute $e(h,h)^{\frac{1}{a}}$. In order to use $\mathcal{A}$ to solve the problem, $\mathcal{S}$ needs to simulate a challenger and response all the queries for $\mathcal{A}$. Simulator $\mathcal{S}$ works by interacting with $\mathcal{A}$ in an IND-ID-CCA game as follows:
Setup: $\mathcal{S}$ randomly picks $l \in \{1, 2, \ldots, q_{H_1}\}$, $I_l \in \mathbb{Z}_p^*$ and $w_1, w_2, \ldots, w_{l-1}, w_{l+1}, \ldots, w_q \in \mathbb{Z}_p^*$. For $i = 1, 2, \ldots, l-1, l+1, \ldots, q$, it computes $I_i = I_l - w_i$. $\mathcal{S}$ expands the polynomial

$$f(z) = \prod_{i=1, i \neq l}^{q} (z + w_i) = \sum_{j=0}^{q-1} c_j z^j,$$

and sets $g = h^{\sum_{j=0}^{q-1} c_j a^j} = h^{f(a)}$, $X = h^{\sum_{j=1}^{q} c_{j-1} a^j} = h^{af(a)} = g^a$. For $i \in \{1, \ldots, q\} \setminus \{l\}$, $\mathcal{S}$ expands

$$f_i(z) = \frac{f(z)}{z + w_i} = \sum_{j=0}^{q-2} d_j z^j,$$

and computes

$$V_i = h^{\sum_{j=0}^{q-2} d_j a^j} = h^{f_i(a)} = h^{\frac{f(a)}{a+w_i}} = g^{\frac{1}{a+w_i}}.$$

Thus, g is a generator of $\mathbb{G}$ and $\mathcal{S}$ knows $q-1$ pairs $\left(w_i, V_i = g^{\frac{1}{a+w_i}}\right)$. $\mathcal{S}$ sets $g_1 = X^{-1} g^{-I_l} = g^{-a-I_l}$ and

computes $v = e(g, g)$. The system public key is

$$mpk = (G, G_T, p, g, g_1, v, n, e, H_1, H_2, H_3).$$

$\mathcal{S}$ implicitly sets the master key $msk = \alpha = -a - I_l$. For all $i \in \{1, \ldots, q\} \setminus \{l\}$, we have $\left(I_i, V_i^{-1}\right) = \left(I_i, g^{\frac{1}{\alpha+I_i}}\right)$.

H_1 **queries**: At any time, $\mathcal{A}$ may issue queries to the random oracle H_1 on ID_i. $\mathcal{S}$ maintains a list L_1 of tuples (ID_i, I_i). This list is initially empty. If ID_i is already on the list L_1, $\mathcal{S}$ responds with $H_1(ID_i) = I_i$. Otherwise, $\mathcal{S}$ sets $H_1(ID_i) = I_i$, adds the new tuple (ID_i, I_i) to the L_1 list and returns I_i.

H_2 **queries**: At any time, adversary $\mathcal{A}$ may issue queries to the random oracle H_2 on $\left(m_i, R_i, C_{1,i}, C_{2,i}\right)$. $\mathcal{S}$ maintains a list L_2 of tuples$(m_i, R_i, C_{1,i}, C_{2,i}, h_{2,i}, C_{3,i}, \lambda_i)$. This list is initially empty. If the tuple $\left(m_i, R_i, C_{1,i}, C_{2,i}\right)$ already appears on the list L_2, $\mathcal{S}$ responds with $H_2\left(m_i, R_i, C_{1,i}, C_{2,i}\right) = h_{2,i}$.

Otherwise, $\mathcal{S}$ picks a random $h_{2,i} \in \mathbb{Z}_p^*$, sets $H_2\left(m_i, R_i, C_{1,i}, C_{2,i}\right) = h_{2,i}$ and returns $h_{2,i}$ to $\mathcal{A}$. To anticipate possible subsequent **Unsigncrypt** requests, $\mathcal{S}$ additionally simulates random oracle H_3 on its own to obtain $h_{3,i} = H_3(R_i) \in \{0, 1\}^n$, computes $C_{3,i} = m_i \oplus h_{3,i}$, $\lambda_i = R_i \cdot v^{h_{2,i}}$ and then adds the new tuple $\left(m_i, R_i, C_{1,i}, C_{2,i}, h_{2,i}, C_{3,i}, \lambda_i\right)$ to the list L_2.

H_3 **queries**: At any time, adversary $\mathcal{A}$ may issue queries to the random oracle H_3 on R_i. $\mathcal{S}$ maintains a list L_3 of tuples $\left(R_i, h_{3,i}\right)$. This list is initially empty. If R_i appears on the list, $\mathcal{S}$ responds with $H_3(R_i) = h_{3,i}$. Otherwise, $\mathcal{S}$ randomly chooses $h_{3,i} \in \{0, 1\}^n$, sets $H_3(R_i) = h_{3,i}$, responds to $\mathcal{A}$ with $h_{3,i}$ and then adds the new tuple $\left(R_i, h_{3,i}\right)$ to the list L_3.

Phase 1: At any time, adversary $\mathcal{A}$ may issue the following queries:

- *Private key queries*. Let ID_i be a private key query issued by adversary $\mathcal{A}$. If $i = l$, $\mathcal{S}$ outputs failure and stops. Otherwise, $\mathcal{S}$ knows $H_1(ID_i) = I_i$ and returns $V_i^{-1} = g^{\frac{1}{\alpha+I_i}}$ to $\mathcal{A}$.
- *Signcryption queries*. Once $\mathcal{A}$ issues a signcryption query for a message m and identities $(ID_s, ID_r) = (ID_i, ID_j)$ for $i, j \in \left\{1, \ldots, q_{H_1}\right\}$. If $i \neq l$, $\mathcal{S}$ knows the sender's private key $d_{ID_i} = V_i^{-1}$ and can answer the query according to the specification of **Signcryption**. If $i = l$, we have $j \neq l$ by the irreflexivity assumption. Thus, $\mathcal{S}$ knows the receiver's private key $d_{ID_j} = V_j^{-1}$ and does the following steps:
  1. Randomly choose $x, y, h \in \mathbb{Z}_p^*$.
  2. Compute $C_1 = d_{ID_j}^{xy^{-1}}$, $C_2 = \left(g_1 g^{I_j}\right)^{-h} g_1^x$ and $r = e\left(C_2 g^{I_i x}, d_{ID_j}\right)$, $C_5 = I_i x$, set $C_4 = y$.
  3. Patch the hash value $H_2(m, R, C_1, C_2)$ to h. $\mathcal{S}$ fails if H_2 is already defined, but this only happens with probability $\frac{q_{sc}+q_{H_2}}{2^k}$.
  4. Compute $C_3 = m \oplus H_3(R)$ and return $\sigma = (C_1, C_2, C_3, y, I_j x)$ to $\mathcal{A}$.

To see its correctness, we implicitly set $s = \left(\frac{\alpha+I_i}{\alpha+I_j}\right) x - h$, then we have

$$\begin{aligned} T &= C_2 g^{C_5} \\ &= \left(g_1 g^{I_j}\right)^{-h} g_1^x \cdot g^{I_i x} \\ &= \left(g^{\alpha+I_j}\right)^{-h} \cdot \left(g^{\alpha+I_i}\right)^x \\ &= \left(g^{\alpha+I_j}\right)^{s-\left(\frac{\alpha+I_i}{\alpha+I_j}\right)x} \cdot \left(g^{\alpha+I_i}\right)^x \\ &= \left(g^{\alpha+I_j}\right)^s \left(g^{\alpha+I_i}\right)^{-x} \cdot \left(g^{\alpha+I_i}\right)^x \\ &= \left(g^{\alpha+I_j}\right)^s \\ &= \left(g_1 g^{H_1(ID_j)}\right)^s, \end{aligned}$$

$$\begin{aligned} S = C_1^{C_4} &= \left(d_{ID_j}^{xy^{-1}}\right)^y = d_{ID_j}^x = \left(g^{\frac{1}{\alpha+I_j}}\right)^{(s+h)\cdot\frac{\alpha+I_j}{\alpha+I_i}} \\ &= \left(g^{\frac{1}{\alpha+I_i}}\right)^{(s+h)} = d_{ID_i}^{s+h}. \end{aligned}$$

Thus, σ is a valid ciphertext.

- *Unsigncryption queries*. Once adversary issues the unsigncryption query on $\sigma = (C_1, C_2, C_3, C_4, C_5)$ for identities $(ID_s, ID_r) = (ID_i, ID_j)$, simulator $\mathcal{S}$ first examines the ID_j. If $j \neq l$, sine $\mathcal{S}$ knows receiver's private key $d_{ID_j} = V_j^{-1}$, and it can answer the query using d_{ID_j}. If $j = l$, we have $i \neq l$ by the irreflexivity assumption. Thus, $\mathcal{S}$ knows the sender's private key $d_{ID_i} = V_i^{-1}$. For all valid ciphertexts, we have

$$s = \log_{d_{ID_i}}\left(C_1^{C_4} \cdot d_{ID_i}^{-h}\right) = \log_{g_1 g^{I_j}}\left(C_2 g^{C_5}\right).$$

That is

$$d_{ID_i}^s = C_1^{C_4} \cdot d_{ID_i}^{-h}, \quad \left(g_1 g^{I_j}\right)^s = C_2 g^{C_5},$$

where $h = H_2(m, R, C_1, C_2)$. Therefore, we have the relation

$$e\left(C_2 g^{C_5}, d_{ID_i}\right) = e\left(g_1 g^{I_j}, C_1^{C_4} \cdot d_{ID_i}^{-h}\right).$$

Then $\mathcal{S}$ computes $\lambda = e\left(C_1^{C_4}, g_1 g^{I_i}\right)$ and searches through list L_2 for entries of the form $\left(m_i, R_i, C_{1,i}, C_{2,i}, h_{2,i}, C_{3,i}, \lambda_i\right)$ indexed by $i \in \left\{1, \ldots, q_{H_2}\right\}$. If none is found, reject σ. Otherwise, $\mathcal{S}$ does the further following

checking for the corresponding indexes

$$\frac{e\left(C_2 g^{C_5}, d_{ID_i}\right)}{e\left(g_1 g^{I_j}, C_1^{C_4}\right)} = e\left(g_1 g^{I_j}, d_{ID_i}\right)^{-h_{2,i}}.$$

If the unique $i \in \{1, \ldots, q_{H_2}\}$ satisfies the above equation, the matching message m_i is returned. Otherwise, $\mathcal{S}$ rejects it. It is clear that, for all queries, the probability to reject a valid ciphertext is smaller than $\frac{q_{us}}{2^k}$.

Challenge: $\mathcal{A}$ chooses two messages m_0, m_1 and challenged identities (ID_s, ID_r) under the restriction that ID_r has not been query its private key in **Phase 1**. If $ID_r \neq ID_l$, $\mathcal{B}$ aborts.

Otherwise, $\mathcal{S}$ randomly chooses $C_3^* \in \{0,1\}^n$, $\beta, x, y \in \mathbb{Z}_p^*$, $C_1^* \in \mathbb{G}$, sets $C_4^* = x, C_5^* = y$ and computes $C_2^* = g^{-\beta} g^{-y}$. $\mathcal{S}$ returns the challenge ciphertext $\sigma^* = \left(C_1^*, C_2^*, C_3^*, C_4^*, C_5^*\right)$ to $\mathcal{A}$. Let $\rho = \frac{\beta}{a}$, since $\alpha = -a - I_l$, we have

$$C_2^* g^{C_5^*} = g^{-\beta} g^{-y} \cdot g^{y} = g^{-\beta} = g^{-\rho a} = g^{\rho(\alpha + I_l)} = \left(g_1 g^{I_l}\right)^{\rho}.$$

$\mathcal{A}$ cannot learn that σ^* is an invalid ciphertext unless it has queried H_2 or H_3 on $e(g,g)^{\rho}$.

Phase 2: $\mathcal{A}$ issues more queries as in **Phase 1** under the restrictions that it cannot make a private key query on ID_r and unsigncryption query on σ^*. $\mathcal{S}$ responds as in **Phase 1**.

Guess: Finally, $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$.

$\mathcal{S}$ ignores the guess of $\mathcal{A}$ and randomly picks an entry $\left(R_i, h_{3,i}\right)$ or $(m_i, R_i, C_{1,i}, C_{2,i}, h_{2,i}, C_{3,i}, \lambda_i)$ from the list L_3 or L_2. From the decryption, we have

$$R^* = e(g,g)^{\rho} = e(g,g)^{\frac{\beta}{a}} = e(h,h)^{\frac{f(a)^2\beta}{a}}.$$

As $f(a)$ is the polynomial contained a, by polynomial division, we have

$$\begin{aligned}\frac{f(a)^2\beta}{a} &= \left(\sum_{j=0}^{q-2} c_{j+1} a^j + \frac{c_0}{a}\right) f(a)\,\beta \\ &= f(a)\,\beta \cdot \sum_{j=0}^{q-2} c_{j+1} a^j + \left(c_0 \beta \sum_{j=0}^{q-2} c_{j+1} a^j + \frac{c_0^2 \beta}{a}\right),\end{aligned}$$

$$\begin{aligned}R^* &= e(h,h)^{\frac{f(a)^2\beta}{a}} \\ &= e(h,h)^{f(a)\beta \cdot \sum_{j=0}^{q-2} c_{j+1} a^j + \left(c_0 \beta \sum_{j=0}^{q-2} c_{j+1} a^j + \frac{c_0^2\beta}{a}\right)} \\ &= e\left(g^{\beta}, h^{\sum_{j=0}^{q-2} c_{j+1} a^j}\right) e\left(h^{c_0\beta}, h^{\sum_{j=0}^{q-2} c_{j+1} a^j}\right) e(h,h)^{\frac{c_0^2\beta}{a}} \\ &= e\left((gh^{c_0})^{\beta}, h^{\sum_{j=0}^{q-2} c_{j+1} a^j}\right) e(h,h)^{\frac{c_0^2\beta}{a}}.\end{aligned}$$

Thus, we can solve the q-BDHI problem by computing

$$e(h,h)^{\frac{1}{a}} = \left(R^* \cdot e\left(\left(gh^{c_0}\right)^{-\beta}, h^{\sum_{j=0}^{q-2} c_{j+1} a^j}\right)\right)^{\frac{1}{c_0^2\beta}}.$$

This completes the description of the simulation. It remains to analyze $\mathcal{S}$'s advantage to solve the q-BDHI problem. We define the following events:

E_1 : $\mathcal{A}$ does not choose ID_l as the challenge receiver's identity.

E_2 : $\mathcal{A}$ issues a private key query on ID_l.

E_3 : $\mathcal{S}$ aborts in a signcryption query because of a collision on H_2.

E_4 : $\mathcal{S}$ rejects a valid ciphertext in one unsigncryption query.

From the above analysis, we know that these four events are independent events and the probability of $\mathcal{S}$ which will not abort is

$$\Pr\left[\neg abort\right] = \Pr\left[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4\right].$$

We have $\Pr\left[\neg E_1\right] = \frac{1}{q_{H_1}}$ and know $\neg E_1$ implies $\neg E_2$. We also observe that $\Pr[E_3] \leq \frac{q_{sc}\left(q_{sc}+q_{H_2}\right)}{2^k}$ and $\Pr[E_4] \leq \frac{q_{us}}{2^k}$. Therefore,

$$\begin{aligned}\Pr\left[\neg abort\right] &= \Pr\left[\neg E_1 \wedge \neg E_3 \wedge \neg E_4\right] \\ &\geq \frac{1}{q_{H_1}}\left(1 - \frac{q_{sc}\left(q_{sc}+q_{H_2}\right)}{2^k}\right)\left(1 - \frac{q_{us}}{2^k}\right).\end{aligned}$$

The chosen entry contains the right element $R^* = e(g,g)^{\rho}$ with probability larger than $\frac{1}{2q_{H_2}+q_{H_3}}$, since L_3 contains no more than $q_{H_2} + q_{H_3}$ records by construction. Thus, the advantage to solve q-BDHI problem is

$$\epsilon' \geq \frac{\epsilon}{q_{H_1}\left(2q_{H_2}+q_{H_3}\right)}\left(1 - \frac{q_{sc}\left(q_{sc}+q_{H_2}\right)}{2^k}\right)\left(1 - \frac{q_{us}}{2^k}\right).$$

□

Theorem 3 *Suppose that an EUF-CMA adversary $\mathcal{A}$ has an advantage $\epsilon \geq \frac{10(q_{sc}+1)\left(q_{sc}+q_{H_2}\right)}{2^k}$ to produce a forgery within a time t, when asking q_{H_i} queries to random oracles H_i $(i = 1, 2, 3)$, q_{sc} signcryption queries and q_{us} queries to the unsigncryption queries. Then there is an algorithm $\mathcal{S}$ to solve the q-SDH problem for $q = q_{H_1}$ in expected time*

$$\begin{aligned}t' \leq\ & 120686 q_{H_1} q_{H_2} \frac{t + \mathrm{O}\left(\left(q_{sc}+q_{us}\right)t_p\right) + q_{us} q_{H_2} t_e}{\epsilon\left(1 - 1/2^k\right)\left(1 - q/2^k\right)} \\ & + \mathrm{O}\left(q^2 t_m\right)\end{aligned}$$

where t_p is the complexity of a pairing computation, t_e is the cost of exponentiation in $\mathbb{G}_T$ and t_m is the cost of scalar multiplication in $\mathbb{G}$.

Proof Suppose adversary $\mathcal{A}$ can produce a forgery in our scheme with advantage ϵ. We build a simulator $\mathcal{S}$ that solves q-SDH problem by running $\mathcal{A}$. Let $\left(h, h^a, h^{a^2}, \ldots, h^{a^q}\right)$ be a random instance of q-SDH problem taken as input by $\mathcal{S}$ and its goal is to find a pair $\left(c, h^{\frac{1}{a+c}}\right)$, $c \in \mathbb{Z}_p^*$. In order to use $\mathcal{A}$ to solve the problem, $\mathcal{S}$ needs to simulate a challenger and responses all the queries for $\mathcal{A}$. Simulator $\mathcal{S}$ works by interacting with $\mathcal{A}$ in an EUF-CMA game as follows:

Setup: $\mathcal{S}$ randomly chooses $w^*, w_1, w_2, \ldots, w_{q-1} \in \mathbb{Z}_p^*$ and expands the polynomial

$$f(z) = \prod_{i=1}^{q-1}(z + w_i) = \sum_{j=0}^{q-1} c_j z^j.$$

Then set $g = h^{\sum_{j=0}^{q-1} c_j a^j} = h^{f(a)}$, $g_1 = h^{\sum_{j=1}^{q} c_{j-1} a^j} = h^{af(a)} = g^a$ and

$$f_i(z) = \frac{f(z)}{z + w_i} = \sum_{j=0}^{q-2} d_j z^j,$$

and compute

$$V_i = h^{\sum_{j=0}^{q-2} d_j a^j} = h^{f_i(a)} = h^{\frac{f(a)}{a+w_i}} = g^{\frac{1}{a+w_i}}.$$

Thus, g is a generator of $\mathbb{G}$ and $\mathcal{S}$ knows pairs $\left(w_i, V_i = g^{\frac{1}{a+w_i}}\right)$. Then $\mathcal{S}$ computes $v = e(g, g)$. The system public key is

$$mpk = (G, G_T, p, g, g_1, v, n, e, H_1, H_2, H_3).$$

$\mathcal{S}$ implicitly sets the master key $msk = a$ and randomly chooses a challenge identity $ID_s^* \in \{0, 1\}^*$ to adversary $\mathcal{A}$.

H_1 **queries**: At any time, adversary $\mathcal{A}$ can query the random oracle H_1 on ID_i. $\mathcal{S}$ maintains a list L_1 of tuples (ID_i, w_i). This list is initially empty. If ID_i is already on the list L_1, $\mathcal{S}$ responds with $H_1(ID_i) = w_i$. Otherwise, if $ID_i = ID_s^*$, $\mathcal{S}$ sets $H_1(ID_i) = w^*$. If $ID_i \neq ID_s^*$, $\mathcal{S}$ sets $H_1(ID_i) = w_i$. $\mathcal{S}$ adds the new tuple (ID_i, w_i) to the list L_1 and returns w_i.

H_2 **queries**: At any time, adversary $\mathcal{A}$ can query the random oracle H_2 on $(m_i, R_i, C_{1,i}, C_{2,i})$. $\mathcal{S}$ maintains a list L_2 of tuples $(m_i, R_i, C_{1,i}, C_{2,i}, h_{2,i}, C_{3,i}, \lambda_i)$. This list is initially empty. If tuple $(m_i, R_i, C_{1,i}, C_{2,i})$ already appears on the list L_2, $\mathcal{S}$ responds with $H_2\left(m_i, R_i, C_{1,i}, C_{2,i}\right) = h_{2,i}$.

Otherwise, $\mathcal{S}$ picks a random $h_{2,i} \in \mathbb{Z}_p^*$, sets $H_2\left(m_i, R_i, C_{1,i}, C_{2,i}\right) = h_{2,i}$ and returns $h_{2,i}$ to $\mathcal{A}$. To anticipate possible subsequent **Unsigncrypt** requests, $\mathcal{S}$ additionally simulates random oracle H_3 on its own to obtain $h_{3,i} = H_3(R_i)$and computes $C_{3,i} = m_i \oplus h_{3,i}, \lambda_i = R_i \cdot v^{h_{2,i}}$ and adds the new tuple $\left(m_i, R_i, C_{1,i}, C_{2,i}, h_{2,i}, C_{3,i}, \lambda_i\right)$ to the list L_2.

H_3 **queries**: At any time, adversary $\mathcal{A}$ can query the random oracle H_3 on R_i. $\mathcal{S}$ maintains a list L_3 of tuples $\left(R_i, h_{3,i}\right)$. This list is initially empty. If R_i is on the list, $\mathcal{S}$ responds with $H_3(R_i) = h_{3,i}$. Otherwise, $\mathcal{S}$ randomly chooses $h_{3,i} \in \{0, 1\}^n$, sets $H_3(R_i) = h_{3,i}$ and responds to $\mathcal{A}$ with $h_{3,i}$ and then adds the new tuple $\left(R_i, h_{3,i}\right)$ to the list L_3.

Private key queries: Let ID_i be a private key query issued by $\mathcal{A}$. If $ID_i = ID_s^*$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ returns $V_i = g^{\frac{1}{a+w_i}}$ to $\mathcal{A}$.

Signcryption queries Once $\mathcal{A}$ issues a signcryption query for a message m and identities $(ID_s, ID_r) = \left(ID_i, ID_j\right)$ for $i, j \in \left\{1, \ldots, q_{H_1}\right\}$. If $ID_i \neq ID_s^*$, $\mathcal{S}$ knows the sender's private key $d_{ID_i} = V_i$ and can answer the query according to the specification of **Signcryption**. If $ID_i = ID_s^*$, we have $ID_j \neq ID_s^*$ by the irreflexivity assumption. Thus, $\mathcal{S}$ knows the receiver's private key $d_{ID_j} = V_j$ and does the following steps:

1. Randomly choose $x, y, h \in \mathbb{Z}_p^*$.
2. Compute $C_1 = d_{ID_j}^{xy^{-1}}$, $C_2 = \left(g_1 g^{I_j}\right)^{-h} g_1^x$, $R = e\left(C_2 g^{I_i x}, d_{ID_j}\right)$ and $C_5 = I_i x$, set $C_4 = y$.
3. Patch the hash value $H_2(m, R, C_1, C_2)$ to h. $\mathcal{S}$ fails if H_2 is already defined but this only happens with probability $\frac{q_{sc}+q_{H_2}}{2^k}$.
4. Compute $C_3 = m \oplus H_3(R)$ and return $\sigma = (C_1, C_2, C_3, y, I_i x)$ to $\mathcal{A}$.

To see it correctness, we implicitly set $s = \left(\frac{\alpha+I_i}{\alpha+I_j}\right) x - h$, then we have

$$\begin{aligned}
S = C_1^{C_4} &= \left(d_{ID_j}^{xy^{-1}}\right)^y = d_{ID_j}^x = \left(g^{\frac{1}{\alpha+I_j}}\right)^{(s+h)\cdot\frac{\alpha+I_j}{\alpha+I_i}} \\
&= \left(g^{\frac{1}{\alpha+I_i}}\right)^{(s+h)} = d_{ID_i}^{s+h}, \\
T = C_2 g^{C_5} &= \left(g_1 g^{I_j}\right)^{-h} g_1^x \cdot g^{I_i x} \\
&= \left(g^{\alpha+I_j}\right)^{-h} \cdot \left(g^{\alpha+I_i}\right)^x \\
&= \left(g^{\alpha+I_j}\right)^{s-\left(\frac{\alpha+I_i}{\alpha+I_j}\right)x} \cdot \left(g^{\alpha+I_i}\right)^x \\
&= \left(g^{\alpha+I_j}\right)^s \left(g^{\alpha+I_i}\right)^{-x} \cdot \left(g^{\alpha+I_i}\right)^x \\
&= \left(g^{\alpha+I_j}\right)^s \\
&= \left(g_1 g^{H_1(ID_j)}\right)^s.
\end{aligned}$$

Thus, σ is a valid ciphertext.

Unsigncryption queries: Once adversary issues an unsigncryption query on $\sigma = (C_1, C_2, C_3, C_4, C_5)$ for identities $(ID_s, ID_r) = (ID_i, ID_j)$, simulator $\mathcal{S}$ first examines the ID_j. If $ID_j \neq ID_s^*$, since $\mathcal{S}$ knows receiver's private key $d_{ID_j} = V_j$, it can answer the query by using d_{ID_j} to decrypt the ciphertext. If $ID_j = ID_s^*$, we have $ID_i \neq ID_s^*$ by the irreflexivity assumption. Thus, $\mathcal{S}$ knows the sender's private key $d_{ID_i} = V_i$. For all valid ciphertexts, we have

$$s = \log_{d_{ID_i}} \left(C_1^{C_4} \cdot d_{ID_i}^{-h} \right) = \log_{g_1 g^{w_j}} \left(C_2 g^{C_5} \right).$$

That is

$$d_{ID_i}^s = C_1^{C_4} \cdot d_{ID_i}^{-h},$$
$$\left(g_1 g^{w_j}\right)^s = C_2 g^{C_5},$$

where $h = H_2(m, R, C_1, C_2)$. Therefore, we have the relation

$$e\left(C_2 g^{C_5}, d_{ID_i}\right) = e\left(g_1 g^{w_j}, C_1^{C_4} \cdot d_{ID_i}^{-h}\right).$$

Then $\mathcal{S}$ computes $\lambda = e\left(C_1^{C_4}, g_1 g^{w_i}\right)$ and searches through list L_2 for entries of the form $\left(m_i, R_i, C_{1,i}, C_{2,i}, h_{2,i}, C_{3,i}, \lambda_i\right)$ indexed by $i \in \{1, \ldots, q_{H_2}\}$. If none is found, reject σ. Otherwise, $\mathcal{S}$ does the further following checking for the corresponding indexes

$$\frac{e\left(C_2 g^{C_5}, d_{ID_i}\right)}{e\left(g_1 g^{w_j}, C_1^{C_4}\right)} = e\left(g_1 g^{w_j}, d_{ID_i}\right)^{-h_{2,i}}.$$

If the unique $i \in \{1, \ldots, q_{H_2}\}$ satisfies the above checking equation, the matching message m_i is returned. Otherwise, $\mathcal{S}$ rejects it. It is clear that, for all queries, the probability to reject a valid ciphertext is smaller than $\frac{q_{us}}{2^k}$.

Forge: From the forking Lemma [16], after a polynomial replays of the adversary $\mathcal{A}$, $\mathcal{S}$ can obtain two valid forge signatures $(C_1, C_2, C_3, C_4, C_5, s, h, m^*)$ and $(C_1, C_2, C_3, C_4', C_5, s, h', m^*)$ for ID^*, where $h = H_2(m^*, R, C_1, C_2)$ and $h' = H_2(m^*, R, C_1, C_2)$, but $h \neq h'$. Since both signatures are valid and satisfy the verification equation, we have

$$e\left(C_1^{C_4}, g_1 g^{H_1(ID_s^*)}\right) e(g, g)^{-h} = e\left(C_1{}^{C_4'}, g_1 g^{H_1(ID_s^*)}\right) e(g, g)^{-h'}.$$

That is

$$e\left(C_1^{C_4 - C_4'}, g_1 g^{H_1(ID_s^*)}\right) = e(g, g)^{h - h'},$$
$$e\left(\left(C_1^{C_4 - C_4'}\right)^{(h-h')^{-1}}, g_1 g^{H_1(ID_s^*)}\right) = e(g, g).$$

Since

$$g_1 g^{H_1(ID_s^*)} = g^{\alpha + w^*},$$

we have

$$V^* = \left(C_1^{C_4 - C_4'}\right)^{(h-h')^{-1}} = g^{\frac{1}{a+w^*}} = h^{\frac{f(a)}{a+w^*}},$$
$$\frac{f(a)}{a + w^*} = \frac{\gamma}{a + w^*} + \sum_{i=0}^{q-2} \gamma_i a^i.$$

Therefore,

$$h^{\frac{1}{a+w^*}} = \left(V^* \cdot h^{-\sum_{i=0}^{q-2} \gamma_i a^i} \right)^{\frac{1}{\gamma}}.$$

We find a pair $\left(w^*, h^{\frac{1}{a+w^*}}\right)$ to the solution of q-SDH problem.

Finally, by the forking Lemma, it comes that, if $\mathcal{A}$ forges a signature in a time t with probability $\epsilon \geq \frac{10(q_{sc}+1)(q_{sc}+q_{H_2})}{2^k}$, simulator $\mathcal{S}$ can solve the q-SDH problem in expected time

$$t' \leq 120686 q_{H_1} q_{H_2} \frac{t + \mathrm{O}\left((q_{sc} + q_{us}) t_p\right) + q_{us} q_{H_2} t_e}{\epsilon \left(1 - 1/2^k\right)\left(1 - q/2^k\right)} + \mathrm{O}\left(q^2 t_m\right),$$

where t_p is the complexity of a pairing computation, t_e is the cost of exponentiation in $\mathbb{G}_T$ and t_m is the cost of scalar multiplication in $\mathbb{G}$. □

5 Evaluation and comparison

In this section, we draw comparisons between our schemes and the other identity-based online/offline encryption schemes and signcryption schemes in the literature. We denote by m_c the modular computation in $\mathbb{Z}_p^*$, $\mathcal{S}$ a one-time strong signature in [11]. We set k the block size of a symmetric key encryption $\mathcal{E}$ in [8], $\tilde{k}$ the security system parameter in [7], n the message size. We first compare our encryption scheme with the existing schemes. Then we provide an evaluation for our signcryption scheme. Without loss of generality, we assume

Table 1 Comparison of computation cost

	Off comp.	On comp.	Security model	Assumption
[11]^{BB}	$\mathbb{G}_T+6\mathbb{G}+\mathcal{S}$	M$v m_c+\mathcal{S}$	IND-sID-CCA	DBDH
[11]^{G}	$5\mathbb{G}_T+3\mathbb{G}$	$\mathrm{M}+2m_c$	IND-ID-CCA	q-DABDHE
[15]	$\mathbb{G}_T+6\mathbb{G}$	$3m_c$	Adaptive CPA	ℓ-BDHI
[8]	$\mathbb{G}_T+6\mathbb{G}$	$\mathcal{E}+2m_c$	IND-sID-CCA	DBDH
[7]	$\mathbb{G}_T+3\mathbb{G}$	m_c	IND-ID-CCA	ℓ-BDHI
[17]	$\mathbb{G}_T+3\mathbb{G}$	m_c	IND-ID-CCA	k-BDHP
[18]	$\mathbb{G}_T+4\mathbb{G}$	$2m_c$	IND-ID-CCA	k-mBDHIP
Ours	$\mathbb{G}_T+2\mathbb{G}$	$2m_c$	IND-ID-CCA	k-CAA1

Table 2 Comparison of storage

	Offline storage	Ciphertext size	Analysis model
[11]^{BB}	$\lvert\mathbb{G}_T\rvert+4\lvert\mathbb{G}\rvert+\lvert\mathcal{S}\rvert+4\lvert\mathbb{Z}_p^*\rvert$	$\lvert\mathbb{G}_T\rvert+4\lvert\mathbb{G}\rvert+2\lvert\mathcal{S}\rvert+2\lvert\mathbb{Z}_p^*\rvert$	Standard
[11]^{G}	$4\lvert\mathbb{G}_T\rvert+2\lvert\mathbb{G}\rvert+4\lvert\mathbb{Z}_p^*\rvert$	$4\lvert\mathbb{G}_T\rvert+\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert$	Standard
[15]	$\lvert\mathbb{G}_T\rvert+4\lvert\mathbb{G}\rvert+6\lvert\mathbb{Z}_p^*\rvert$	$4\lvert\mathbb{G}\rvert+3\lvert\mathbb{Z}_p^*\rvert+n$	Random Oracle
[8]	$4\lvert\mathbb{G}\rvert+3\lvert\mathbb{Z}_p^*\rvert+\mathrm{k}$	$4\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+\lvert\mathcal{E}\rvert$	Standard
[7]	$\lvert\mathbb{G}_T\rvert+2\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+\tilde{k}$	$2\lvert\mathbb{G}\rvert+\lvert\mathbb{Z}_p^*\rvert+\tilde{k}+\mathrm{n}$	Random Oracle
[17]	$\lvert\mathbb{G}_T\rvert+2\lvert\mathbb{G}\rvert+4\lvert\mathbb{Z}_p^*\rvert+2\mathrm{n}$	$2\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+2\mathrm{n}$	Random Oracle
[18]	$\lvert\mathbb{G}_T\rvert+3\lvert\mathbb{G}\rvert+4\lvert\mathbb{Z}_p^*\rvert+\mathrm{n}$	$3\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+\mathrm{n}$	Random Oracle
Ours	$\lvert\mathbb{G}_T\rvert+\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+\mathrm{n}$	$\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+\mathrm{n}$	Random Oracle

Table 3 Comparison of computation cost and size

	Off comp.	On comp.	Off stor.	Ciphertext size
[18]	$\mathbb{G}_T+7\mathbb{G}$	$2m_c$	$2\lvert\mathbb{G}_T\rvert+4\lvert\mathbb{G}\rvert+4\lvert\mathbb{Z}_p^*\rvert+n$	$\lvert\mathbb{G}_T\rvert+5\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+n$
[14]	$\mathbb{G}_T+7\mathbb{G}$	$3m_c$	$\lvert\mathbb{G}_T\rvert+4\lvert\mathbb{G}\rvert+6\lvert\mathbb{Z}_p^*\rvert$	$4\lvert\mathbb{G}\rvert+3\lvert\mathbb{Z}_p\rvert+n$
[13]	$\mathbb{G}_T+4\mathbb{G}$	$2m_c$	$\lvert\mathbb{G}_T\rvert+3\lvert\mathbb{G}\rvert+4\lvert\mathbb{Z}_p^*\rvert$	$3\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+n$
Ours	$\mathbb{G}_T+3\mathbb{G}$	$2m_c$	$\lvert\mathbb{G}_T\rvert+2\lvert\mathbb{G}\rvert+3\lvert\mathbb{Z}_p^*\rvert$	$2\lvert\mathbb{G}\rvert+2\lvert\mathbb{Z}_p^*\rvert+n$

one multi-exponentiation is equal to two exponentiations in group $\mathbb{G}$.

Table 1 provides a comparison of computation cost and shows that our scheme has the lightest computation in the offline phase. It reduces at least one element in $\mathbb{G}$. In the online phase, it needs two modular computations in $\mathbb{Z}_p^*$ which still has a good computational efficiency. The proposed scheme can achieve IND-ID-CCA under the k-CAA1 assumption.

From Table 2, we achieve shorter ciphertext size than the best-known IBOOE instantiation. We only have one element in $\mathbb{G}$ in the final ciphertext, which greatly reduces the bandwidth when transmitting the encrypted massage. In the offline storage, our scheme also achieves a comparable result, although the protocol in [8] is slightly better in the offline storage compared to our scheme. However, [8] depends on one symmetric key encryption. It leads to the key management problem and the security also depends on the security of the symmetric key encryption algorithm. Thus, in some way, our scheme has the smallest size and saves one element in $\mathbb{G}$ in both the offline and online phases.

Table 3 shows that compared to the existing schemes, our identity-based online/offline signcryption scheme achieves the least computation and the smallest storage in both offline and online stages. In the offline computation, we reduce at least one operation in $\mathbb{G}$. Compared to [13], our proposed scheme saves one element in $\mathbb{G}$ and one element in $\mathbb{Z}_p^*$ in the offline storage, which leads to reduce one element in $\mathbb{G}$ in the final ciphertext. The online computation is still comparable and requires two modular multiplications. These improvements in our two schemes are very important when the communication bandwidth is limited.

We remark that the security proofs of our schemes are under the random oracle model. Random oracle model has usually been regarded as a heuristic method since no hash functions can be used as the random oracle. However, the proof analysis in the random oracle model is more efficient than that without random oracle and achieves an acceptable security level, which has been commonly accepted. In some situations, the weight of efficiency is over security level. Therefore, in this case, the random oracle model provides a better choice.

6 Conclusion

We proposed two novel and efficient identity-based online/offline schemes for encryption and signcryption, respectively. Both our schemes are proven secure in the random oracle model and achieve the shortest ciphertext. The proposed schemes can reduce one operation in $\mathbb{G}$ in offline computation and save at least one element in $\mathbb{G}$ in both offline storage and online ciphertext. These improvements are significant for the devices with limited storage and computation power. Therefore, our schemes are desirable candidates for lightweight devices.

References

1. Paulo, S., Barreto, L.M., Libert, B.: Noel McCullagh, and Jean-Jacques Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In: Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, pp. 515–532 (2005)
2. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Advances in Cryptology-EUROCRYPT 2004, Lecture Notes in Computer Science, vol. 3027, pp. 223–238. (2004)
3. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Proceedings of Advances in Cryptology-Eurocrypt 2004, Lecture Notes in Computer Science, vol. 3027, pp. 56–73. (2004)
4. Boyen, X.: Multipurpose identity-based signcryption. In: Advances in Cryptology-CRYPTO 2003, Lecture Notes in Computer Science, vol. 2729, pp. 383–399. (2003)
5. Chen, L., Cheng, Z.: Security proof of sakai-kasahara's identity-based encryption scheme. In: Cryptography and Coding 2005, Lecture Notes in Computer Science, vol. 3796, pp. 442–459. (2005)
6. Chen, L., Malone-Lee, J.: Improved identity-based signcryption. In: Public Key Cryptography-PKC 2005, Lecture Notes in Computer Science, vol. 3386, pp. 362–379. (2005)
7. Chow, S.S.M., Liu, J.K., Zhou, J.: Identity-based online/offline key encapsulation and encryption. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS'11, pp. 52–60. (2011)
8. Chu, C-K., Liu, J.K., Zhou, J., Bao, F., Deng, R.H.: Practical ID-based encryption for wireless sensor network. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS'10, pp. 337–340. (2010)
9. Even, S., Goldreich, O., Micali, S.: On-line/off-line digital signatures. J. Cryptol. **9**(1), 35–67 (1996)
10. Gentry, C.: Practical identity-based encryption without random oracles. In: Advances in Cryptology-EUROCRYPT 2006, Lecture Notes in Computer Science, vol. 4004, pp. 445–464. (2006)
11. Guo, F., Mu, Y., Chen, Z.: Identity-based online/offline encryption. In: Financial Cryptography and Data Security, Lecture Notes in Computer Science, vol. 5143, pp. 247–261. (2008)
12. Lai, J., Mu, Y., Guo, F., Susilo, W.: Improved identity-based online/offline encryption. In: ACISP 2015, Lecture Notes in Computer Science, vol. 9144, pp. 160–173. (2015)
13. Li, F., Khan, M.K., Alghathbar, K., Takagi, T.: Identity-based online/offline signcryption for low power devices. J. Netw. Comput. Appl. **35**(1), 340–347 (2012)
14. Liu, J.K., Baek, J., Zhou, J.: Online/offline identity-based signcryption revisited. In: Proceedings of the Information Security and Cryptology, Inscrypt 2010, Lecture Notes in Computer Science, vol. 6584, pp. 36–51. (2011)
15. Liu, J.K., Zhou, J.: An efficient identity-based online/offline encryption scheme. In: Applied Cryptography and Network Security, Lecture Notes in Computer Science ACNS'09, vol. 5536, pp. 156–167. (2009)
16. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**(3), 361–396 (2000)
17. Selvi, S.S.D., Vivek, S.S, Rangan, C.P: Identity based online/offline encryption scheme. Cryptology ePrint archive, report 2010/178, (2010)
18. Selvi, S.S.D, Vivek, S.S, Rangan, C.P.: Identity based online/offline encryption and signcryption schemes revisited. In: Security Aspects in Information Technology, Lecture Notes in Computer Science, vol. 7011, pp. 111–127. (2011)
19. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Advances in Cryptology-CRYPTO 1984, Lecture Notes in Computer Science, vol. 196, pp. 47–53. (1985)
20. Sun, D., Huang, X., Mu, Y., Susilo, W.: Identity based online/offline signcryption. In: Network and Parallel Computing Workshops, Lecture Notes in Computer Science, vol. 5245, pp. 34–41. (2008)
21. Sun, D., Mu, Y., Susilo, W.: A generic construction of identity-based online/offline signcryption. In: IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA'08, pp. 707–712. (2008)
22. Waters, B.: Efficient identity-based encryption without random oracles. In: Advances in Cryptology-EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, pp. 114–127 (2005)
23. Waters, B.: Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In: Advances in Cryptology-CRYPTO 2009, Lecture Notes in Computer Science, vol. 5677, pp. 619–636. (2009)