Wireless Netw (2017) 23:145–158
DOI 10.1007/s11276-015-1145-3

Certificateless online/offline signcryption for the Internet of Things

Fagen Li[1] · **Yanan Han**[1] · **Chunhua Jin**[1]

Published online: 12 December 2015

Abstract The Internet of Things (IoT) is an emerging network paradigm that aims to obtain the interactions among pervasive things through heterogeneous networks. Security is an important task in the IoT. Luo et al. (Secur Commun Netw 7(10): 1560–1569, 2014) proposed a certificateless online/offline signcryption (COOSC) scheme for the IoT (hereafter called LTX). Unfortunately, Shi et al. showed that LTX is not secure. An adversary can easily obtain the private key of a user by a ciphertext. Recently, Li et al. proposed a new COOSC scheme (hereafter called LZZ). However, both LTX and LZZ need a point multiplication operation in the online phase, which is not suitable for resource-constrained devices. To overcome this weakness, we propose a new COOSC scheme and prove its security in the random oracle model. In addition, we analyze the performance of our scheme and show its application in the IoT.

Keywords Internet of Things · Security · Signcryption · Certificateless cryptosystem

1 Introduction

The Internet of Things (IoT) is an emerging network paradigm that aims to get the interactions among pervasive things through heterogeneous networks [1, 2]. The pervasive things (e.g. human beings, computers, appliances and cars) can communicate with each other at any time, any place, and in any way. Many information technologies serve as the building blocks of the IoT, such as radio frequency identification (RFID), wireless sensor networks (WSNs), machine-to-machine interfaces (M2M), cloud computing, and so on [3]. The IoT has been widely applied in the smart grid, intelligent transportation, and smart city. The security task to the IoT is challenging because of the scalability, heterogeneity, open nature of wireless communication and limited resources of WSNs and RFID [4]. Luo et al. [5] proposed a certificateless online/offline signcryption (COOSC) scheme (hereafter called LTX) and designed a secure communication model using the COOSC scheme. The COOSC has the following two advantages: (1) it simultaneously achieves confidentiality and authentication at a low cost; (2) it has neither public key certificates nor key escrow problem. Unfortunately, Shi et al. [6] showed that LTX is not secure. An adversary can easily obtain the private key of a user by a ciphertext. Recently, Li et al. [7] gave a new COOSC scheme (hereafter called LZZ). However, both LTX and LZZ need a point multiplication operation in the online phase, which is not suitable for resource-constrained devices.

1.1 Motivation and contribution

To overcome the weakness that needs a point multiplication operation in the online phase of LTX and LZZ, we propose a new COOSC scheme. Using the random oracle model, we prove that our scheme has the indistinguishability against adaptive chosen ciphertext attack (IND-CCA2) under q-bilinear Diffie–Hellman inversion (q-BDHI) and modified bilinear inverse Diffie–Hellman (mBIDH) problems and has the existential unforgeability against adaptive chosen messages attack (EUF-CMA)

✉ Fagen Li
fagenli@uestc.edu.cn

[1] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

under q-strong Diffie–Hellman (q-SDH) and modified inverse computational Diffie–Hellman (mICDH) problems. Compared with LTX and LZZ, our scheme has no point multiplication operation in the online phase. In the unsigncryption phase, our scheme has less computational cost than LTX and LZZ. For the ciphertext size and private key size, our scheme is also shorter than LTX and LZZ. We analyze the performance of our scheme and show its application in the IoT.

1.2 Related work

Signcryption [8] is a cryptographic primitive that performs both the functions of digital signature and public key encryption in a logical single step, at a cost significantly lower than that required by the traditional signature-then-encryption method. Signcryption is very suitable for resource-constrained devices since it simultaneously achieves confidentiality, authentication, integrity and non-repudiation at a lower cost.

In a public key cryptosystem, there exist three methods for the authenticity of a public key, public key infrastructure (PKI), identity-based cryptosystem (IBC) and certificateless cryptosystem (CLC). According to the three public key authentication methods, signcryption can be divided into three types: PKI-based signcryption, identity-based signcryption (IBSC) and certificateless signcryption (CLSC). In the PKI, a certificate authority (CA) issues a certificate that binds a public key and the identity of a user by the signature of the CA. The expired certificates are issued by a certificate revocation list (CRL). The PKI has been widely used in the Internet security. Some famous signcryption schemes in the PKI have been proposed [8, 9]. However, the PKI may not be a good choice for resource-constrained devices since the certificates management is heavy, including distribution, verification, storage and revocation. To reduce the burden of the certificates management, some IBSC schemes were proposed [10–13]. Compared with the PKI, the main advantage of the IBC is the elimination of public key certificates. In the IBC, a user's public key is derived directly from its identity information, such as telephone numbers, email addresses and IP addresses. There is a trusted third party called private key generator (PKG) who takes charge of generating a private key for each user using a master secret key. Authenticity of a public key is explicitly verified without requiring a public key certificate. However, the IBC has a weakness called key escrow problem since the PKG holds all the users' private keys. To overcome this problem, some CLSC schemes were proposed [14–16]. The CLC uses a trusted third party called the key generating center (KGC) who takes charge of generating a partial private key for each user using a master secret key. Then the user generates a secret value and combines the secret value with the partial private key to form a full private key. Note that the KGC does not know the full private key since it does not know the secret value. Therefore, the CLC has neither public key certificates nor key escrow problem.

In 2002, An et al. introduced a new notion called online/offline signcryption (OOSC) by combining the concepts of online/offline signature and signcryption together [17]. A OOSC scheme splits the signcryption into two phases: offline phase and online phase. In the offline phase, most heavy operations are done without the knowledge of a message. In the online phase, only light operations are done when the message is available. OOSC is very suitable to supply the security solution for resource-constrained devices such as sensor nodes, RFID, smart cards and mobile phones. A resource-constrained device is characterized by low computational power and limited battery lifetime and capacity. It can be loaded with the precomputed result of the offline phase from a more powerful device. The entire signcryption process can be finished quickly using the precomputed result. Some PKI-based OOSC schemes are proposed [18–20]. Sun et al. [21] proposed an identity-based online/offline signcryption (IBOOSC) scheme. However, this scheme needs a receiver's identity in the offline phase. To overcome this weakness, Liu et al. [22] proposed a new IBOOSC scheme that does not need a receiver's identity in the offline stage. Li et al. [23] gave a new IBOOSC that has the great advantage in the offline storage and ciphertext length. Li and Xiong [24] proposed a heterogeneous OOSC to secure the communication of the IoT. In the heterogeneous OOSC, the sender belongs to the IBC and the receiver belongs to the PKI. Senthil kumaran and Ilango [25] used the heterogeneous OOSC to design a secure routing in the WSNs.

Recently, the COOSC is considered in [5–7]. However, these schemes need a point multiplication operation in the online phase. We know that the aim of online/offline technique is to shift the heavy operations to the offline phase. Therefore, [5–7] violate this object. In this paper, we give a new COOSC scheme that removes all heavy operations in the online phase.

1.3 Organization

The rest of the paper is organized as follows. The bilinear pairings and security assumptions are introduced in Sect. 2. The formal model of COOSC is given in Sect. 3. We describe a new COOSC scheme in Sect. 4. We give the security and performance of our scheme in Sect. 5. The application of our scheme in the IoT is described in Sect. 6. Finally, the conclusions are given in Sect. 7.

2 Preliminaries

In this section, we describe the bilinear pairings and security assumptions.

Let G_1 and G_2 be two cyclic groups with same prime order p. G_1 is an additive group and G_2 is a multiplicative group. Let P be a generator of G_1. A bilinear pairing is a map $\hat{e}: G_1 \times G_1 \rightarrow G_2$ that satisfies the following properties:

1. *Bilinearity* $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in G_1$, $a, b \in \mathbb{Z}_p^*$.
2. *Non-degeneracy* there are $P, Q \in G_1$ such that $\hat{e}(P, Q) \neq 1$, where 1 is the identity element of group G_2.
3. *Computability* $\hat{e}(P, Q)$ can be efficiently computed for all P,$Q \in G_1$.

The modified Weil pairing and Tate pairing provide admissible maps of this kind. Please refer to [26] for details. The security of our scheme depends on the hardness of the following assumptions.

Definition 1 Given groups G_1 and G_2 of the same prime order p, a generator P of G_1 and a bilinear map $\hat{e}: G_1 \times G_1 \rightarrow G_2$, q-bilinear Diffie–Hellman inversion (q-BDHI) problem in $(G_1, G_2, \hat{e})$ is to compute $\hat{e}(P, P)^{1/\alpha}$ given $(P, \alpha P, \alpha^2 P, \ldots, \alpha^q P)$. Here $\alpha \in \mathbb{Z}_p^*$.

Definition 2 Given groups G_1 and G_2 of the same prime order p, a generator P of G_1 and a bilinear map $\hat{e}: G_1 \times G_1 \rightarrow G_2$, the modified bilinear inverse Diffie–Hellman (mBIDH) problem in $(G_1, G_2, \hat{e})$ is to compute $\hat{e}(P, P)^{1/(\alpha+\gamma)}$ given $(P, \alpha P, \gamma)$. Here $\alpha, \gamma \in \mathbb{Z}_p^*$.

Definition 3 Given groups G_1 and G_2 of the same prime order p, a generator P of G_1 and a bilinear map $\hat{e}: G_1 \times G_1 \rightarrow G_2$, the q-strong Diffie–Hellman (q-SDH) problem in $(G_1, G_2, \hat{e})$ is to find a pair $(w, \frac{1}{\alpha+w} P) \in \mathbb{Z}_p^* \times G_1$ given $(P, \alpha P, \alpha^2 P, \ldots, \alpha^q P)$. Here $\alpha \in \mathbb{Z}_p^*$.

Definition 4 Given a group G_1 of prime order p and a generator P of G_1, the modified inverse computational Diffie–Hellman (mICDH) problem in G_1 is to compute $(\alpha + \gamma)^{-1} P$ given $(P, \alpha P, \gamma)$. Here $\alpha, \gamma \in \mathbb{Z}_p^*$.

3 Certificateless online/offline signcryption

COOSC is an online/offline signcryption scheme in the certificateless cryptosystem. In such a scheme, the signcryption process is split into two phases: offline phase and online phase. In the offline phase, most heavy cryptographic operations are done without the knowledge of a message. In the online phase, only light cryptographic operations are done when the message is available. Now we give the formal definition and security notions of the COOSC.

3.1 Syntax

A generic COOSC scheme consists of the following seven algorithms [5, 7].

Setup is a probabilistic algorithm run by a KGC that takes as input a security parameter k, and outputs a master secret key s and the system parameters *params* that contains a master public key P_{pub}. For simplicity, we omit *params* in the other algorithms in the following content.

PPKE is a partial private key extraction algorithm run by the KGC that takes as input a user's identity *ID* and a master secret key s, and outputs a partial private key D_{ID}.

UKG is a user key generation algorithm run by a user that takes as input an identity *ID*, and outputs a secret value x_{ID} and a public key PK_{ID}. The public key can be published without a certificate.

FPKS is a full private key setup algorithm run by a user that takes as input a partial private key D_{ID} and a secret value x_{ID}, and outputs a full private key S_{ID}.

OffSC is a probabilistic offline signcryption algorithm run by a sender that takes as input a sender's private key S_A and a receiver's identity ID_B and public key PK_B, and outputs an offline signcryption result δ. Note that a message is not required in this phase.

OnSC is an online signcryption algorithm run by a sender that takes as input a message m, an offline signcryption δ and a sender's identity ID_A and public key PK_A, and outputs a ciphertext σ.

USC is a deterministic unsigncryption algorithm run by a receiver that takes as input a ciphertext σ, a sender's identity ID_A and public key PK_A, and a receiver's private key S_B, and outputs a message m or a failure symbol $\perp$ if σ is not a valid ciphertext between the sender and the receiver.

The above algorithms should satisfy the consistency constraint of the COOSC, i.e. if

$$\delta = OffSC(S_A, ID_B, PK_B), \sigma = OnSC(m, \delta, ID_A, PK_A)$$

then we have

$$m = USC(\sigma, ID_A, PK_A, S_B).$$

3.2 Security notions

In the CLC, we need consider two types of adversaries [26], Type I and Type II. A Type I adversary models an attacker that is a common user and does not have the

KGC's master secret key. But it can adaptively replace a user' public key with a selected valid public key. A Type II adversary models an honest-but-curious KGC who knows the master secret key. But it can not replace a user's public key. In addition, a signcryption scheme should satisfy confidentiality [i.e. indistinguishability against adaptive chosen ciphertext attack (IND-CCA2)] and unforgeability [i.e. existential unforgeability against adaptive chosen messages attack (EUF-CMA)] [14]. So, in the CLSC, we should consider four security notions, IND-CCA2-I for a Type I adversary, IND-CCA2-II for a Type II adversary, EUF-CMA-I for a Type I adversary and EUF-CMA-II for a Type II adversary. The four games for the four security notions are described as follows [5, 7].

The first game (Game-I) is a confidentiality game played between a Type I adversary $\mathcal{A}_I$ and a challenger $\mathcal{C}$.

Initial $\mathcal{C}$ runs *Setup* algorithm with a security parameter k and gives the system parameters *params* to $\mathcal{A}_I$.

Phase 1 $\mathcal{A}_I$ performs a polynomially bounded number of queries in an adaptive manner (i.e., each query may depend on the answer to the previous queries).

- *Partial private key extraction queries* $\mathcal{A}_I$ submits an identity ID to $\mathcal{C}$. $\mathcal{C}$ runs *PPKE* algorithm and sends a partial private key D_{ID} to $\mathcal{A}_I$.
- *Private key queries* $\mathcal{A}_I$ submits an identity ID to $\mathcal{C}$. $\mathcal{C}$ runs *FPKS* algorithm and gives a full private key S_{ID} to $\mathcal{A}_I$ ($\mathcal{C}$ may first run *PPKE* and *UKG* algorithms if necessary).
- *Public key queries* $\mathcal{A}_I$ may ask a public key query by submitting an identity ID. $\mathcal{C}$ runs *UKG* algorithm and sends a public key PK_{ID} to $\mathcal{A}_I$.
- *Public key replacement queries* $\mathcal{A}_I$ can replace a public key PK_{ID} with a selected value.
- *Signcryption queries* $\mathcal{A}_I$ may ask a signcryption query by submitting a message m, a sender's identity ID_i and a receiver's identity ID_j. $\mathcal{C}$ first runs *FPKS* algorithm to get the sender's private key S_i and *UKG* algorithm to get the sender's public key PK_i and the receiver's public key PK_j. Then $\mathcal{C}$ runs $OffSC(S_i, ID_j, PK_j)$ to obtain the offline signcryption δ. Finally, $\mathcal{C}$ sends the result of algorithm $OnSC(m, \delta, ID_i, PK_i)$ to $\mathcal{A}_I$. If the public key associated with ID_i has been replaced, $\mathcal{C}$ does not know the sender's secret value. In this case, we require $\mathcal{A}_I$ to supply it.
- *Unsigncryption queries* $\mathcal{A}_I$ may ask an unsigncryption query by submitting a ciphertext σ, a sender's identity ID_i and a receiver's identity ID_j. $\mathcal{C}$ first runs *FPKS* algorithm to get the receiver's private key S_j and *UKG* algorithm to get the sender's public key PK_i. Then $\mathcal{C}$ sends the result of algorithm *USC* $(\sigma, ID_i, PK_i, S_j)$ to $\mathcal{A}_I$. If the public key associated with ID_j has been replaced, $\mathcal{C}$ does not know the receiver's secret value. In this case, we require $\mathcal{A}_I$ to supply it.

Challenge $\mathcal{A}_I$ decides when phase 1 ends. $\mathcal{A}_I$ outputs two equal length messages (m_0, m_1), a sender's identity ID_A and a receiver's identity ID_B on which it wishes to be challenged. Note that ID_B can not be submitted to a private key query in phase 1. ID_B also can not be submitted to both a partial private key extraction query and a public key replacement query. $\mathcal{C}$ chooses a random bit $\beta \in \{0, 1\}$, computes $\delta^* = OffSC(S_A, ID_B, PK_B)$ and the challenge ciphertext $\sigma^* = OnSC(m_\beta, \delta^*, ID_A, PK_A)$ which is sent to $\mathcal{A}_I$. If the public key associated with ID_A has been replaced, $\mathcal{C}$ may not know the sender's secret value. In this case, we require $\mathcal{A}_I$ to supply it.

Phase 2 $\mathcal{A}_I$ may ask a polynomially bounded number of queries adaptively again as in the phase 1. This time, $\mathcal{A}_I$ can not ask a private key query on ID_B. $\mathcal{A}_I$ also can not ask a partial private key extraction query on ID_B if the public key of this identity has been replaced before the challenge phase. In addition, it can not ask an unsigncryption query on (σ^*, ID_A, ID_B) to obtain the corresponding message unless the public key PK_A or PK_B has been replaced after the challenge phase.

Guess $\mathcal{A}_I$ outputs a bit β' and wins the game if $\beta' = \beta$.

The advantage of $\mathcal{A}_I$ is defined as $\mathrm{Adv}(\mathcal{A}) = |2\Pr[\beta' = \beta] - 1|$, where $\Pr[\beta' = \beta]$ is the probability that $\beta' = \beta$.

Definition 5 A COOSC scheme is $(\epsilon, t, q_{ppk}, q_{sk}, q_{pk}, q_{pkr}, q_s, q_u)$-IND-CCA2-I secure if there does not exist a probabilistic t-polynomial time adversary $\mathcal{A}_I$ that has advantage at least ϵ after at most q_{ppk} partial private key extraction queries, q_{sk} private key queries, q_{pk} public key queries, q_{pkr} public key replacement queries, q_s signcryption queries and q_u unsigncryption queries in the Game-I.

The second game (Game-II) is a confidentiality game played between a Type II adversary $\mathcal{A}_{II}$ and a challenger $\mathcal{C}$.

Initial $\mathcal{C}$ runs *Setup* algorithm with a security parameter k and gives a master secret key s and the system parameters *params* to $\mathcal{A}_{II}$.

Phase 1 $\mathcal{A}_{II}$ makes a polynomially bounded number of private key queries, public key queries, signcryption queries and unsigncryption queries just like in the Game-I. Note that the partial private key extraction queries is not needed since $\mathcal{A}_{II}$ can do it by itself.

Challenge $\mathcal{A}_{II}$ decides when phase 1 ends. $\mathcal{A}_{II}$ outputs two equal length messages (m_0, m_1), a sender's identity ID_A and a receiver's identity ID_B on which it wishes to be challenged. Note that ID_B can not be submitted to a private key query in phase 1. $\mathcal{C}$ chooses a random bit $\beta \in \{0, 1\}$, computes $\delta^* = OffSC(S_A, ID_B, PK_B)$ and $\sigma^* = OnSC(m_\beta, \delta^*, ID_A, PK_A)$, and sends σ^* to $\mathcal{A}_{II}$.

Phase 2 $\mathcal{A}_{II}$ may ask a polynomially bounded number of queries adaptively again as in the phase 1. This time, $\mathcal{A}_{II}$ can not ask a private key query on ID_B. In addition, it can

not make an unsigncryption query on (σ^*, ID_A, ID_B) to obtain the corresponding message.

Guess $\mathcal{A}_{II}$ outputs a bit β' and wins the game if $\beta' = \beta$.

The advantage of $\mathcal{A}_{II}$ is defined as $\mathrm{Adv}(\mathcal{A}) = |2\Pr[\beta' = \beta] - 1|$, where $\Pr[\beta' = \beta]$ is the probability that $\beta' = \beta$.

Definition 6 A COOSC scheme is $(\epsilon, t, q_{sk}, q_{pk}, q_s, q_u)$-IND-CCA2-II secure if there does not exist a probabilistic t-polynomial time adversary $\mathcal{A}_{II}$ that has advantage at least ϵ after at most q_{sk} private key queries, q_{pk} public key queries, q_s signcryption queries and q_u unsigncryption queries in the Game-II.

Definition 7 A COOSC scheme is said to be IND-CCA2 secure if it is both IND-CCA2-I secure and IND-CCA2-II secure.

The Game-I and Game-II catch the insider security for confidentiality since the adversary knows all senders' private keys [17]. The insider security ensures the forward security of a signcryption scheme. That is, the confidentiality is still kept if the sender's private key is disclosed.

The third game (Game-III) is an unforgeability game played between a Type I adversary $\mathcal{F}_I$ and a challenger $\mathcal{C}$.

Initial $\mathcal{C}$ runs *Setup* algorithm with a security parameter k and gives the system parameters *params* to $\mathcal{F}_I$.

Attack $\mathcal{F}_I$ performs a polynomially bounded number of queries just like in the Game-I.

Forgery $\mathcal{F}_I$ outputs a ciphertext σ^*, a sender's identity ID_A and a receiver's identity ID_B. $\mathcal{F}_I$ wins this game if the following conditions hold:

1. $USC\ (\sigma^*, ID_A, PK_A, S_B) = m^*$.
2. $\mathcal{F}_I$ has not asked a private key query for ID_A.
3. $\mathcal{F}_I$ has not asked both a public key replacement query for ID_A and a partial private key extraction query for ID_A.
4. $\mathcal{F}_I$ has not asked a signcryption query on (m^*, ID_A, ID_B).

The advantage of $\mathcal{F}_I$ is defined as the probability that it wins.

Definition 8 A COOSC scheme is $(\epsilon, t, q_{ppk}, q_{sk}, q_{pk}, q_{pkr}, q_s, q_u)$-EUF-CMA-I secure if there does not exist a probabilistic t-polynomial time adversary $\mathcal{F}_I$ that has advantage at least ϵ after at most q_{ppk} partial private key extraction queries, q_{sk} private key queries, q_{pk} public key queries, q_{pkr} public key replacement queries, q_s signcryption queries and q_u unsigncryption queries in the Game-III.

The fourth game (Game-IV) is an unforgeability game played between a Type II adversary $\mathcal{F}_{II}$ and a challenger $\mathcal{C}$.

Initial $\mathcal{C}$ runs *Setup* algorithm with a security parameter k and gives a master secret key s and the system parameters *params* to $\mathcal{F}_{II}$.

Attack $\mathcal{F}_{II}$ performs a polynomially bounded number of queries just like in the Game-II.

Forgery $\mathcal{F}_{II}$ outputs a ciphertext σ^*, a sender's identity ID_A and a receiver's identity ID_B. $\mathcal{F}_{II}$ wins this game if the following conditions hold:

1. $USC\ (\sigma^*, ID_A, PK_A, S_B) = m^*$.
2. $\mathcal{F}_{II}$ has not asked a private key query for ID_A.
3. $\mathcal{F}_{II}$ has not asked a signcryption query on (m^*, ID_A, ID_B).

The advantage of $\mathcal{F}_{II}$ is defined as the probability that it succeeds.

Definition 9 A COOSC scheme is $(\epsilon, t, q_{sk}, q_{pk}, q_s, q_u)$-EUF-CMA-II secure if there does not exist a probabilistic t-polynomial time adversary $\mathcal{F}_{II}$ that has advantage at least ϵ after at most q_{sk} private key queries, q_{pk} public key queries, q_s signcryption queries and q_u unsigncryption queries in the Game-IV.

Definition 10 A COOSC scheme is EUF-CMA secure if it is both EUF-CMA-I secure and EUF-CMA-II secure.

In the Game-III and Game-IV, the adversary is allowed to know the receiver's private key S_B. The insider security for unforgeability is obtained [17].

4 An efficient COOSC scheme

In this section, we propose an efficient COOSC scheme. Here we assume that the sender's identity is ID_A and the receiver's identity is ID_B.

Setup given a security parameter k, the KGC chooses an additive group G_1 and a multiplicative G_2 of the same prime order p, a generator P of G_1, a bilinear map $\hat{e} : G_1 \times G_1 \to G_2$, and four hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$, $H_2 : G_1 \to \mathbb{Z}_p^*$, $H_3 : G_2 \to \{0,1\}^n$ and $H_4 : \{0,1\}^n \times \{0,1\}^* \times G_1 \times G_2 \times G_1 \to \mathbb{Z}_p^*$. Here n is the number of bits of a message to be sent. The KGC randomly selects a master secret key $s \in \mathbb{Z}_p^*$ and computes the master public key $P_{pub} = sP$. The KGC publishes the system parameters

$$\{G_1, G_2, p, \hat{e}, n, P, P_{pub}, g, H_1, H_2, H_3, H_4\}$$

and keeps s secret. Here $g = \hat{e}(P, P)$.

PPKE a user sends its identity ID_U to the KGC. The KGC computes a partial private key

$$D_U = \frac{1}{H_1(ID_U) + s} P$$

and returns D_U to the user.

UKG A user with identity ID_U randomly selects $x_U \in \mathbb{Z}_p^*$ as the secret value and sets

$$PK_U = x_U(H_1(ID_U)P + P_{pub})$$

as the public key. The public key can be published without certification.

FPKS Given a partial private key D_U and a secret value x_U, the user sets a full private key

$$S_U = \frac{1}{x_U + H_2(PK_U)} D_U.$$

OffSC Given a sender's private key S_A and a receiver's identity ID_B and public key PK_B, this algorithm works as follows.

1. Choose x, α from $\mathbb{Z}_p^*$ randomly.
2. Compute $r = g^x$.
3. Compute $S' = \alpha S_A$.
4. Compute $T = x(PK_B + H_2(PK_B)(H_1(ID_B)P + P_{pub}))$.
5. Output a offline signcryption $\delta = (x, \alpha^{-1}, r, S', T)$.

OnSC given a message m, a offline signcryption δ and a sender's identity ID_A and public key PK_A, this algorithm works as follows.

1. Compute $c = m \oplus H_3(r)$.
2. Compute $h = H_4(m, ID_A, PK_A, r, S')$.
3. Compute $\theta = (x + h)\alpha^{-1} \bmod p$.
4. Output a ciphertext $\sigma = (c, \theta, S', T)$.

USC given a ciphertext σ, a sender's identity ID_A and public key PK_A, and a receiver's private key S_B, this algorithm works as follows.

1. Compute $r = \hat{e}(T, S_B)$.
2. Recover $m = c \oplus H_3(r)$.
3. Compute $h = H_4(m, ID_A, PK_A, r, S')$.
4. Compute $S = \theta S'$.
5. Accept the message if and only if

$$r = \hat{e}(S, PK_A + H_2(PK_A)(H_1(ID_A)P + P_{pub}))g^{-h},$$

return $\perp$ otherwise.

We summarize the communication process in Fig 1.

Now we check the consistency of our scheme. First, because

$$T = x(PK_B + H_2(PK_B)(H_1(ID_B)P + P_{pub})),$$

we have Eq. (1).

$$\begin{aligned}\hat{e}(T, S_B) &= \hat{e}(x(PK_B + H_2(PK_B)(H_1(ID_B)P + P_{pub})), S_B) \\ &= \hat{e}(x(x_B + H_2(PK_B))(H_1(ID_B) + s) \\ &\quad P, \frac{1}{x_B + H_2(PK_B)} \frac{1}{H_1(ID_B) + s} P) = \hat{e}(P, P)^x \\ &= g^x \\ &= r\end{aligned} \tag{1}$$

Second, since

$$S = \theta S' = (x + h)\alpha^{-1}\alpha S_A = (x + h)S_A,$$

we have Eq. (2).

$$\begin{aligned}&\hat{e}(S, PK_A + H_2(PK_A)(H_1(ID_A)P + P_{pub}))g^{-h} \\ &= \hat{e}((x + h)S_A, (x_A + H_2(PK_A))(H_1(ID_A) + s)P)g^{-h} \\ &= \hat{e}((x + h)\frac{1}{x_A + H_2(PK_A)} \frac{1}{H_1(ID_A) + s} \\ &P, (x_A + H_2(PK_A))(H_1(ID_A) + s)P)g^{-h} \\ &= \hat{e}((x + h)P, P)g^{-h} \\ &= \hat{e}(P, P)^{(x+h)}g^{-h} \\ &= g^{(x+h)}g^{-h} \\ &= g^x \\ &= r\end{aligned} \tag{2}$$

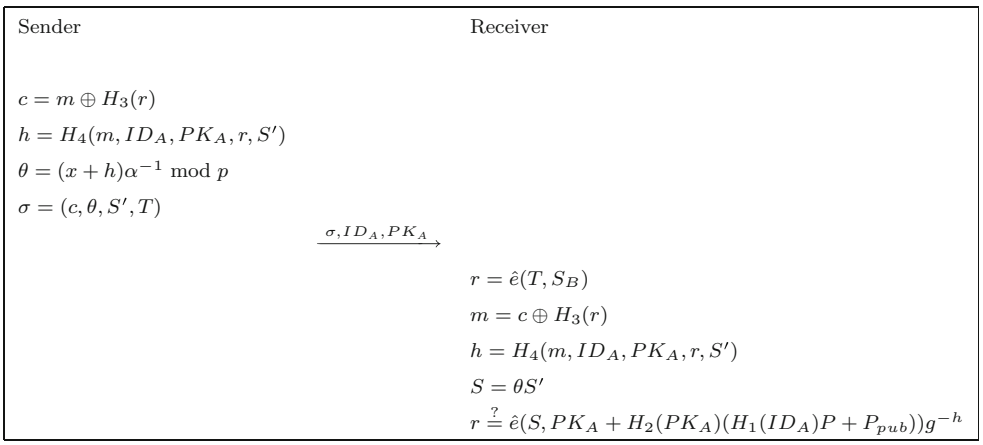


Fig. 1 Certificateless online/offline signcryption communication

5 Analysis of the scheme

In this section, we analyze the security and performance of our scheme.

5.1 Security

Theorem 1 In the random oracle model, our scheme is IND-CCA2 secure under the q-BDHI and mBIDH assumptions.

Proof This theorem follows from the following Lemmas 1 and 2. □

Lemma 1 In the random oracle model, if there is an adversary $\mathcal{A}_I$ that has a non-negligible advantage ϵ against the IND-CCA2-I security of our scheme when running in a time t and performing q_{ppk} partial private key extraction queries, q_{sk} private key queries, q_{pk} public key queries, q_{pkr} public key replacement queries, q_s signcryption queries, q_u unsigncryption queries and q_{H_i} queries to oracles H_i $(i = 1, 2, 3, 4)$, then we can construct an algorithm $\mathcal{C}$ that can solve the q-BDHI problem for $q = q_{H_1}$ with an advantage

$$\epsilon' \geq \frac{\epsilon}{q_{H_1}(q_{H_3} + 2q_{H_4})}\left(1 - \frac{q_s(q_s + q_{H_4})}{2^k}\right)\left(1 - \frac{q_u}{2^k}\right)$$

in a time $t' \leq t + O(q_s + q_u)t_p + O(q_{H_1}^2)t_m + O(q_u q_{H_4})t_e$, where t_p is the cost for one pairing operation, t_m is the cost for a point multiplication operation in G_1 and t_e is the cost for an exponentiation operation in G_2.

Proof We show how $\mathcal{C}$ can use $\mathcal{A}_I$ as a subroutine to solve a random instance $(P, \alpha P, \alpha^2 P, \ldots, \alpha^q P)$ of the q-BDHI problem.

Initial in a preparation phase, $\mathcal{C}$ chooses $\ell \in \{1, \ldots, q_{H_1}\}$, elements $e_\ell \in \mathbb{Z}_p^*$ and $w_1, \ldots, w_{\ell-1}, w_{\ell+1}, w_q \in \mathbb{Z}_p^*$ randomly. For $i = 1, \ldots, \ell-1, \ell+1, \ldots, q$, $\mathcal{C}$ sets $e_i = e_\ell - w_i$. Then $\mathcal{C}$ uses its input to set a generator $Q \in G_1$ and an element $X = \alpha Q \in G_1$ such that it knows $q-1$ pairs $(w_i, V_i = \frac{1}{\alpha + w_i}Q)$ for $i \in \{1, \ldots, q\}\backslash\{\ell\}$ as in [27]. To do so, $\mathcal{C}$ expands the polynomial

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

A generator Q and an element X can be obtained as

$$Q = \sum_{j=0}^{q-1} c_j(\alpha^j P) = f(\alpha)P$$

and

$$X = \sum_{j=1}^{q} c_{j-1}(\alpha^j P) = \alpha f(\alpha)P = \alpha Q.$$

As in [27], the pairs (w_i, V_i) for $i \in \{1, \ldots, q\}\backslash\{\ell\}$ can be gotten by expanding

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

and setting

$$V_i = \sum_{j=0}^{q-2} d_j(\alpha^j P) = f_i(\alpha)P = \frac{f(\alpha)}{\alpha + w_i}P = \frac{1}{\alpha + w_i}Q.$$

The master public key of the KGC is set as $Q_{pub} = -X - e_\ell Q = (-\alpha - e_\ell)Q$ and its corresponding master secret key is implicitly set to $s = -\alpha - e_\ell \in \mathbb{Z}_p^*$. For all $i \in \{1, \ldots, q\}\backslash\{\ell\}$, we have $(e_i, -V_i) = (e_i, \frac{1}{e_i + s}Q)$. $\mathcal{C}$ gives $\mathcal{A}_I$ the system parameters with Q, $Q_{pub} = (-\alpha - e_\ell)Q$ and $g = \hat{e}(Q, Q)$.

Phase 1 $\mathcal{C}$ simulates $\mathcal{A}_I$'s challenger in the Game-I. $\mathcal{C}$ keeps four lists L_1, L_2, L_3 and L_4 to simulate oracles H_1, H_2, H_3 and H_4, respectively. $\mathcal{C}$ should maintain the consistency and avoid collision for these answers. In addition, $\mathcal{C}$ maintains a list L_k that is initially empty to keep the public key information. We assume that H_1 queries are different, that $\mathcal{A}_I$ will ask $H_1(ID)$ before ID is used in the other queries and that the target identity ID_B is submitted to H_1 at some point. In addition, we suppose that the sender's identity is different to the receiver's identity by irreflexivity assumption [10].

- H_1 queries: These queries are indexed by a counter v that is initially set to 1. For a $H_1(ID_v)$ query, $\mathcal{C}$ returns e_v as the answer, inserts (ID_v, e_v) into the list L_1 and increments v.
- H_2 queries: For a $H_2(PK_i)$ query, $\mathcal{C}$ checks if the value of H_2 has been defined for the PK_i. If yes, $\mathcal{C}$ returns previously defined value. Otherwise, $\mathcal{C}$ returns a random $h_{2,i} \in \mathbb{Z}_p^*$ to $\mathcal{A}_I$ and inserts $(PK_i, h_{2,i})$ into the list L_2.
- H_3 queries: For a $H_3(r_i)$ query, $\mathcal{C}$ checks if the value of H_3 has been defined for the same input. If yes, $\mathcal{C}$ returns previously defined value. Otherwise, $\mathcal{C}$ returns a random $h_{3,i} \in \{0,1\}^n$ to $\mathcal{A}_I$ and inserts $(r_i, h_{3,i})$ into the list L_3.
- H_4 queries: For a $H_4(m_i, ID_i, PK_i, r_i, S_i')$ query, $\mathcal{C}$ checks if the value of H_4 has been defined for the same input. If yes, $\mathcal{C}$ returns the previously defined value. Otherwise, $\mathcal{C}$ returns a random $h_{4,i} \in \mathbb{Z}_p^*$ to $\mathcal{A}_I$. In addition, to answer the following queries, $\mathcal{C}$ simulates H_3 oracle to

get $h_{3,i} = H_3(r_i) \in \{0,1\}^n$ and sets $c_i = m_i \oplus h_{3,i}$ and $\xi_i = r_i \cdot \hat{e}(Q,Q)^{h_{4,i}}$. Finally, $\mathcal{C}$ inserts the tuple $(m_i, ID_i, PK_i, r_i, S'_i, h_{4,i}, c_i, \xi_i)$ into the list L_4.

- *Partial private key extraction queries* $\mathcal{A}_I$ can ask a partial private key extraction query by submitting an identity ID_i. If $i = \ell$, then $\mathcal{C}$ fails and stops. Otherwise, $\mathcal{C}$ knows that $H_1(ID_i) = e_i$ and returns $-V_i = \frac{1}{e_i+s}Q$ to $\mathcal{A}_I$.
- *Private key queries* $\mathcal{A}_I$ can ask a private key query by submitting an identity ID_i. If $i = \ell$, then $\mathcal{C}$ fails and stops. Otherwise, $\mathcal{C}$ knows the partial private key $-V_i = \frac{1}{e_i+s}Q$. Then $\mathcal{C}$ searches the list L_k for the entry (ID_i, PK_i, x_i) ($\mathcal{C}$ generates a new key pair information if this entry does not exist) and returns $S_i = -\frac{1}{x_i+h_{2,i}}V_i$.
- *Public key queries* $\mathcal{A}_I$ chooses an identity ID_i and sends it to $\mathcal{C}$. If the list L_k has a tuple (ID_i, PK_i, x_i), then $\mathcal{C}$ gives PK_i to $\mathcal{A}_I$. Otherwise, $\mathcal{C}$ selects a random number $x_i \in \mathbb{Z}_p^*$, sets $PK_i = x_i(e_iQ + Q_{pub})$, inserts (ID_i, PK_i, x_i) into the list L_k, and gives PK_i to $\mathcal{A}_I$.
- *Public key replacement queries* for a public key replacement query for (ID_i, PK_i), $\mathcal{C}$ updates the list L_k with tuple $(ID_i, PK_i, \perp)$. Here $\perp$ denotes an unknown value.
- *Signcryption queries* $\mathcal{A}_I$ can ask a signcryption query by submitting a message m, a sender's identity ID_i and a receiver's identity ID_j. If $i \neq \ell$, $\mathcal{C}$ knows the sender's private key S_i and can answer this query according to the steps of *OffSC* and *OnSC* algorithms. If $i = \ell$ but $j \neq \ell$ by the irreflexivity assumption [10], $\mathcal{C}$ knows the receiver's private key S_j. To answer this query, $\mathcal{C}$ first randomly chooses $\theta, \eta, h \in \mathbb{Z}_p^*$, computes $S' = \theta^{-1}\eta S_j$, $T = \eta(PK_\ell + h_{2,\ell}(e_\ell Q + Q_{pub})) - h(PK_j + h_{2,j}(e_jQ + Q_{pub}))$ and $r = \hat{e}(T, S_j)$. Then $\mathcal{C}$ defines the hash value $H_4(m, ID_\ell, PK_\ell, r, S') = h$. Finally, $\mathcal{C}$ computes $c = m \oplus H_3(r)$ and returns $\sigma = (c, \theta, S', T)$ to $\mathcal{A}_I$. $\mathcal{C}$ fails if H_4 is already defined but this only happens with probability $(q_s + q_{H_4})/2^k$.
- *Unsigncryption queries* $\mathcal{A}_I$ can ask an unsigncryption query by submitting a ciphertext $\sigma = (c, \theta, S', T)$, a sender's identity ID_i and a receiver's identity ID_j. If $j \neq \ell$, $\mathcal{C}$ knows the receiver's private key S_j and can answer this query according to the steps of *USC* algorithm. If $j = \ell$, $\mathcal{C}$ knows the sender's private key S_i since $i \neq \ell$ by the irreflexivity assumption [10]. For all valid ciphertexts, we have

$$\log_{S_i}(\theta S' - hS_i) = \log_{PK_\ell + h_{2,\ell}(e_\ell Q + Q_{pub})} T,$$

where $h = H_4(m, ID_i, PK_i, r, S')$. So the following equation

$$\hat{e}(T, S_i) = \hat{e}(PK_\ell + h_{2,\ell}(e_\ell Q + Q_{pub}), \theta S' - hS_i)$$

holds. $\mathcal{C}$ first computes $\xi = \hat{e}(\theta S', PK_i + h_{2,i}(e_iQ + Q_{pub}))$ and then searches the list L_4 for the entries of the form $(m_i, ID_i, PK_i, r_i, S'_i, h_{4,i}, c, \xi)$ indexed by $i \in \{1, \ldots, q_{H_4}\}$. If there is no such an entry, σ is rejected. Otherwise, $\mathcal{C}$ further checks whether the following equation holds for the corresponding indexes

$$\frac{\hat{e}(T, S_i)}{\hat{e}(PK_\ell + h_{2,\ell}(e_\ell Q + Q_{pub}), \theta S')} = \hat{e}(PK_\ell + h_{2,\ell}(e_\ell Q + Q_{pub}), S_i)^{-h_{4,i}}.$$

If the unique $i \in \{1, \ldots, q_{H_4}\}$ that satisfies this above equation is found, $\mathcal{C}$ returns the matching message m_i. Otherwise, σ is also rejected. For all unsigncryption queries, the probability to reject a valid ciphertext is less than or equal to $\frac{q_u}{2^k}$.

Challenge $\mathcal{A}_I$ generates two equal length messages (m_0, m_1), a sender's identity ID_A and a receiver's identity ID_B on which it hopes to be challenged. If $ID_B \neq ID_\ell$, $\mathcal{C}$ fails. Otherwise, $\mathcal{C}$ chooses $c^* \in \{0,1\}^n$, $\lambda, \theta^* \in \mathbb{Z}_p^*$, $S'^* \in G_1$ randomly and sets $T^* = -\lambda x_B Q - \lambda h_{2,B} Q$. $\mathcal{C}$ returns a ciphertext $\sigma^* = (c^*, \theta^*, S'^*, T^*)$ to $\mathcal{A}_I$. If we define $\rho = \lambda/\alpha$ and since $s = -\alpha - e_\ell$, we have

$$\begin{aligned} T^* &= -\lambda x_B Q - \lambda h_{2,B} Q \\ &= -\rho\alpha x_B Q - \rho\alpha h_{2,B} Q \\ &= (e_B + s)\rho x_B Q + (e_B + s)\rho h_{2,B} Q \\ &= \rho x_B(e_B Q + Q_{pub}) + \rho h_{2,B}(e_B Q + Q_{pub}) \\ &= \rho PK_B + \rho h_{2,B}(e_B Q + Q_{pub}) \\ &= \rho(PK_B + h_{2,B}(e_B Q + Q_{pub})). \end{aligned}$$

$\mathcal{A}_I$ cannot identify that σ^* is not a valid ciphertext unless it asks a H_3 or H_4 query on $\hat{e}(Q,Q)^\rho$.

Phase 2 $\mathcal{A}_I$ can ask a polynomially bounded number of queries adaptively again as in the phase 1 with the following limitation: (1) it can not ask a private key query on ID_B; (2) it can not ask a partial private key extraction query on ID_B if the public key of ID_B has been replaced before the challenge phase; (3) it can not ask an unsigncryption query on (σ^*, ID_A, ID_B) to obtain the corresponding message unless the public key PK_A or PK_B has been replaced after the challenge phase. $\mathcal{C}$ answer $\mathcal{A}_I$'s queries according to the same method as in the phase 1.

Guess $\mathcal{A}_I$ outputs a guess bit β' which is ignored by $\mathcal{C}$.

$\mathcal{C}$ fetches a random entry $(r_i, h_{3,i})$ from the list L_3 or $(m_i, ID_i, PK_i, r_i, S'_i, h_{4,i}, c_i, \xi_i)$ from the list L_4. Since L_3 contains no more than $q_{H_3} + q_{H_4}$ records, the selected entry will contain the correct element $r_i = \hat{e}(Q,Q)^\rho = \hat{e}(P,P)^{f(\alpha)^2\lambda/\alpha}$ with probability $1/(q_{H_3} + 2q_{H_4})$. As in [12], the q-BDHI problem can be solved by noting that, if $\xi^* = \hat{e}(P,P)^{1/\alpha}$, then

$$\hat{e}(Q,Q)^{1/\alpha} = \xi^{*^{(c_0^2)}}\hat{e}(\sum_{j=0}^{q-2} c_{j+1}(\alpha^j P), c_0 P)\hat{e}(Q, \sum_{j=0}^{q-2} c_{j+1}(\alpha^j)P).$$

This finishes the description of the whole simulation. Now we analyze $\mathcal{C}$'s advantage. Define the events E_1, E_2, E_3, E_4 and E_5 as

E_1: $\mathcal{A}_I$ has not chosen ID_ℓ as the receiver's identity in the challenge phase.

E_2: $\mathcal{A}_I$ has asked a private key query on ID_ℓ.

E_3: $\mathcal{A}_I$ has asked a partial private key extraction query on ID_ℓ and the public key of ID_ℓ has been replaced before the challenge phase.

E_4: $\mathcal{C}$ aborts in a signcryption query because of a collision on H_4.

E_5: $\mathcal{C}$ aborts in an unsigncryption query because of rejecting a valid ciphertext.

According to above analysis, we know that the probability of $\mathcal{C}$ not aborting is

$$\Pr[\neg\text{abort}] = \Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4 \wedge \neg E_5].$$

We know that $\Pr[\neg E_1] = 1/q_{H_1}$, $\Pr[E_4] \le q_s(q_s + q_{H_4})/2^k$ and $\Pr[E_5] \le q_u/2^k$. In addition, we know that $\neg E_1$ implies $\neg E_2$ and $\neg E_3$. So we have

$$\Pr[\neg\text{abort}] \ge \frac{1}{q_{H_1}}\left(1 - \frac{q_s(q_s+q_{H_4})}{2^k}\right)\left(1 - \frac{q_u}{2^k}\right).$$

In addition, $\mathcal{C}$ chooses the correct element from the list L_3 or L_4 with probability $1/(q_{H_3} + 2q_{H_4})$. Therefore, we have

$$\epsilon' \ge \frac{\epsilon}{q_{H_1}(q_{H_3}+2q_{H_4})}\left(1 - \frac{q_s(q_s+q_{H_4})}{2^k}\right)\left(1 - \frac{q_u}{2^k}\right).$$

The bound on $\mathcal{C}$'s computation time is obtained from the fact that $\mathcal{C}$ needs $O(q_{H_1}^2)$ point multiplication operations in G_1 in the preparation phase, $O(q_s + q_u)$ pairing operations and $O(q_u q_{H_4})$ exponentiation operations in G_2 in the signcryption and unsigncryption queries. □

Lemma 2 In the random oracle model, if there is an adversary $\mathcal{A}_{II}$ that has a non-negligible advantage ϵ against the IND-CCA2-II security of our scheme when running in a time t and performing q_{sk} private key queries, q_{pk} public key queries, q_s signcryption queries, q_u unsigncryption queries and q_{H_i} queries to oracles H_i $(i = 1, 2, 3, 4)$, then we can construct an algorithm $\mathcal{C}$ that can solve the mBIDH problem with an advantage

$$\epsilon' \ge \frac{\epsilon}{q_{H_1}(q_{H_3}+2q_{H_4})}\left(1 - \frac{q_s(q_s+q_{H_4})}{2^k}\right)\left(1 - \frac{q_u}{2^k}\right)$$

in a time $t' \le t + O(q_s + q_u)t_p + O(q_u q_{H_4})t_e$, where t_p is the cost for one pairing operation and t_e is the cost for an exponentiation operation in G_2.

Proof We show how $\mathcal{C}$ can use $\mathcal{A}_{II}$ as a subroutine to solve a random instance $(P, \alpha P, \gamma)$ of the mBIDH problem.

Initial $\mathcal{C}$ gives $\mathcal{A}_{II}$ a master secret key s and the system parameters *params* with $P_{pub} = sP$. Here s is randomly chosen by $\mathcal{C}$.

Phase 1 $\mathcal{C}$ simulates $\mathcal{A}_{II}$'s challenger in the Game-II. $\mathcal{C}$ maintains four lists L_1, L_2, L_3 and L_4 to simulate oracles H_1, H_2, H_3 and H_4, respectively. $\mathcal{C}$ should keep the consistency and avoid collision for these answers. In addition, $\mathcal{C}$ keeps a list L_k that is initially empty to maintain the public key information. We suppose that H_1 queries are different and that $\mathcal{A}_{II}$ will ask $H_1(ID)$ before ID is used in the other queries. In addition, we suppose that the sender's identity is different to the receiver's identity by irreflexivity assumption [10]. $\mathcal{C}$ chooses a random number $\ell \in \{1, 2, \ldots, q_{H_1}\}$ and answers $\mathcal{A}_{II}$'s queries as follows.

- *H_1queries* For each new ID_i, $\mathcal{C}$ randomly selects $e_i \in \mathbb{Z}_p^*$, inserts (ID_i, e_i) into the list L_1 and answers $H_1(ID_i) = e_i$.
- *H_2queries* For a $H_2(PK_i)$ query, $\mathcal{C}$ checks if the value of H_2 has been defined for the same input. If yes, $\mathcal{C}$ returns previously defined value. Otherwise, $\mathcal{C}$ checks if $PK_i = e_i\alpha P + s\alpha P$ (i.e., $i = \ell$). If yes, $\mathcal{C}$ returns $h_{2,\ell} = \gamma$ and inserts (PK_ℓ, γ) into the list L_2. If no, $\mathcal{C}$ selects a random $h_{2,i}$ from $\mathbb{Z}_p^*$, returns $h_{2,i}$ as an answer and inserts $(PK_i, h_{2,i})$ into the list L_2.
- H_3 queries: For a $H_3(r_i)$ query, $\mathcal{C}$ checks if the value of H_3 has been defined for the same input. If yes, $\mathcal{C}$ returns previously defined value. Otherwise, $\mathcal{C}$ selects a random $h_{3,i}$ from $\{0,1\}^n$, returns $h_{3,i}$ as an answer and inserts $(r_i, h_{3,i})$ into the list L_3.
- H_4 queries: For a $H_4(m_i, ID_i, PK_i, r_i, S_i')$ query, $\mathcal{C}$ checks if the value of H_4 has been defined for the same input. If yes, $\mathcal{C}$ gives the previously defined value. Otherwise, $\mathcal{C}$ returns a random $h_{4,i} \in \mathbb{Z}_p^*$ as the answer. In addition, to answer the following queries, $\mathcal{C}$ simulates H_3 oracle on its own to get $h_{3,i} = H_3(r_i) \in \{0,1\}^n$ and computes $c_i = m_i \oplus h_{3,i}$ and $\xi_i = r_i \cdot \hat{e}(P,P)^{h_{4,i}}$. Lastly, $\mathcal{C}$ inserts the tuple $(m_i, ID_i, PK_i, r_i, S_i', h_{4,i}, c_i, \xi_i)$ into the list L_4.
- *Private key queries* $\mathcal{A}_{II}$ can ask a private key query by submitting an identity ID_i. If $i = \ell$, then $\mathcal{C}$ fails and stops. Otherwise, $\mathcal{C}$ runs H_1 oracle to get (ID_i, e_i). Then $\mathcal{C}$ searches the list L_k for the entry (ID_i, PK_i, x_i) ($\mathcal{C}$ generates a new key pair information if this entry does not exist) and returns

  $$S_i = \frac{1}{x_i + h_{2,i}}\frac{1}{e_i + s}P.$$

  Here $h_{2,i} = H_2(PK_i)$.
- *Public key queries* $\mathcal{A}_{II}$ can ask a public key query by submitting an identity ID_i. If $i \ne \ell$, $\mathcal{C}$ selects a random

$x_i \in \mathbb{Z}_p^*$, sets a public key $PK_i = x_i(e_iP + P_{pub})$, inserts (ID_i, PK_i, x_i) into the list L_k and returns PK_i to $\mathcal{A}_{II}$. Otherwise, $\mathcal{C}$ returns $PK_\ell = e_\ell \alpha P + s\alpha P$ and inserts $(ID_\ell, PK_\ell, \perp)$ into the list L_k.

- *Signcryption queries* $\mathcal{A}_{II}$ can ask a signcryption query by submitting a message m, a sender's identity ID_i and a receiver's identity ID_j. If $i \neq \ell$, $\mathcal{C}$ knows the sender's private key S_i and can answer this query according to the steps of *OffSC* and *OnSC* algorithms. If $i = \ell$ but $j \neq \ell$ by the irreflexivity assumption, $\mathcal{C}$ knows the receiver's private key S_j. To answer this query, $\mathcal{C}$ first randomly chooses $\theta, \eta, h \in \mathbb{Z}_p^*$ and computes $S' = \theta^{-1}\eta S_j$, $T = \eta(PK_\ell + h_{2,\ell}(e_\ell P + P_{pub})) - h(PK_j + h_{2,j}(e_jP + P_{pub}))$ and $r = \hat{e}(T, S_j)$. Then $\mathcal{C}$ defines the hash value $H_4(m, ID_\ell, PK_\ell, r, S')$ to h. Finally, $\mathcal{C}$ computes $c = m \oplus H_3(r)$ and returns $\sigma = (c, \theta, S', T)$ to $\mathcal{A}_{II}$. $\mathcal{C}$ fails if H_4 is already defined but this only happens with probability $(q_s + q_{H_4})/2^k$.
- *Unsigncryption queries* $\mathcal{A}_{II}$ can make an unsigncryption query about a ciphertext $\sigma = (c, \theta, S', T)$, a sender's identity ID_i and a receiver's identity ID_j. If $j \neq \ell$, then $\mathcal{C}$ knows the receiver's private key S_j and can answer this query according to the steps of *USC* algorithm. If $j = \ell$, $\mathcal{C}$ knows the sender's private key S_i since $i \neq \ell$ by the irreflexivity assumption. For all valid ciphertexts, we have

  $$\log_{S_i}(\theta S' - hS_i) = \log_{PK_\ell + h_{2,\ell}(e_\ell P + P_{pub})} T,$$

  where $h = H_4(m, ID_i, PK_i, r, S')$. Therefore, we have

  $$\hat{e}(T, S_i) = \hat{e}(PK_\ell + h_{2,\ell}(e_\ell P + P_{pub}), \theta S' - hS_i).$$

  $\mathcal{C}$ first computes $\xi = \hat{e}(\theta S', PK_i + h_{2,i}(e_i P + P_{pub}))$ and then searches the list L_4 for the entries of the form $(m_i, ID_i, PK_i, r_i, S_i', h_{4,i}, c, \xi)$ indexed by $i \in \{1, \ldots, q_{H_4}\}$. If there is no such an entry, σ is rejected. Otherwise, $\mathcal{C}$ further checks whether the following equation holds for the corresponding indexes

  $$\frac{\hat{e}(T, S_i)}{\hat{e}(PK_\ell + h_{2,\ell}(e_\ell P + P_{pub}), \theta S')} = \hat{e}(PK_\ell + h_{2,\ell}(e_\ell P + P_{pub}), S_i)^{-h_{4,i}}$$

  If the unique $i \in \{1, \ldots, q_{H_4}\}$ that satisfies this above equation is found, then $\mathcal{C}$ returns the matching message m_i. Otherwise, σ is also rejected. For all unsigncryption queries, the probability to reject a valid ciphertext is less than or equal to $\frac{q_u}{2^k}$.

Challenge $\mathcal{A}_{II}$ generates two equal length messages (m_0, m_1), a sender's identity ID_A and a receiver's identity ID_B on which it hopes to be challenged. If $ID_B \neq ID_\ell$, $\mathcal{C}$ fails. Otherwise $\mathcal{C}$ randomly chooses $c^* \in \{0,1\}^n$, $\lambda, \theta^* \in \mathbb{Z}_p^*$, $S'^* \in G_1$ and sets $T^* = \lambda P$. $\mathcal{C}$ returns the ciphertext $\sigma^* = (c^*, \theta^*, S'^*, T^*)$ to $\mathcal{A}_{II}$. $\mathcal{A}_{II}$ cannot identify that σ^* is not a valid ciphertext unless it makes a H_3 or H_4 query on $\hat{e}(T^*, S_B)$.

Phase 2 $\mathcal{A}_{II}$ can ask a polynomially bounded number of queries adaptively again as in the phase 1 with the limitation: (1) it can not ask a private key query on ID_B; (2) it can not ask an unsigncryption query on (σ^*, ID_A, ID_B) to obtain the corresponding message. $\mathcal{C}$ answer $\mathcal{A}_{II}$'s queries according to the same method as in the phase 1.

Guess $\mathcal{A}_{II}$ produces a bit β' which is ignored by $\mathcal{C}$.

$\mathcal{C}$ fetches a random entry $(r_i, h_{3,i})$ from the list L_3 or $(m_i, ID_i, PK_i, r_i, S_i', h_{4,i}, c_i, \xi_i)$ from the list L_4. Since the list L_3 includes no more than $q_{H_3} + q_{H_4}$ records, the chosen entry will contain the right element $r_i = \hat{e}(T^*, S_B)$ with probability $1/(q_{H_3} + 2q_{H_4})$. The mBIDH problem can be solved by noting that, if

$$\hat{e}(T^*, S_B) = \hat{e}(\lambda P, \frac{1}{\alpha + \gamma}\frac{1}{e_i + s}P),$$

we have

$$\hat{e}(P, P)^{\frac{1}{\alpha+\gamma}} = r_i^{\frac{e_i+s}{\lambda}}.$$

This finishes the description of the whole simulation. Now we analyze $\mathcal{C}$'s advantage. Define the events E_1, E_2, E_3 and E_4 as

E_1: $\mathcal{A}_{II}$ does not select ID_ℓ as the receiver's identity in the challenge phase.

E_2: $\mathcal{A}_{II}$ has asked a private key query on the identity ID_ℓ.

E_3: $\mathcal{C}$ aborts in a signcryption query because of a collision on H_4.

E_4: $\mathcal{C}$ aborts in an unsigncryption query because of rejecting a valid ciphertext.

According to above analysis, we know that the probability of $\mathcal{C}$ not aborting is

$$\Pr[\neg\text{abort}] = \Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4].$$

From the above analysis, we know that $\Pr[\neg E_1] = 1/q_{H_1}$, $\Pr[E_3] \leq q_s(q_s + q_{H_4})/2^k$ and $\Pr[E_4] \leq q_u/2^k$. In addition, we know that $\neg E_1$ implies $\neg E_2$. So we have

$$\Pr[\neg\text{abort}] \geq \frac{1}{q_{H_1}}\left(1 - \frac{q_s(q_s + q_{H_4})}{2^k}\right)\left(1 - \frac{q_u}{2^k}\right).$$

In addition, $\mathcal{C}$ chooses the correct element from the list L_3 or L_4 with probability $1/(q_{H_3} + 2q_{H_4})$. Therefore, we have

$$\epsilon' \geq \frac{\epsilon}{q_{H_1}(q_{H_3} + 2q_{H_4})}\left(1 - \frac{q_s(q_s + q_{H_4})}{2^k}\right)\left(1 - \frac{q_u}{2^k}\right).$$

The bound on $\mathcal{C}$'s computation time can be obtained from the fact that $\mathcal{C}$ needs $O(q_s + q_u)$ pairing operations and $O(q_u q_{H_4})$ exponentiation operations in G_2 in the signcryption and unsigncryption queries. □

Table 1 Comparison of existing schemes

Schemes	*OffSC*			*OnSC*			*USC*			Offline storage	Ciphertext size	Private key size	Security
	M	E	P	M	E	P	M	E	P				
LTX [5]	2	1	0	1	0	0	4	0	3	$\|\mathbb{Z}_p^*\| + 2\|G_1\| + \|G_2\|$	$2\|\mathbb{Z}_p^*\| + 2\|G_1\| + \|m\|$	$\|\mathbb{Z}_p^*\| + \|G_1\|$	No
LZZ [7]	5	1	0	1	0	0	5	1	5	$3\|\mathbb{Z}_p^*\| + 4\|G_1\| + \|G_2\|$	$2\|\mathbb{Z}_p^*\| + 4\|G_1\| + \|m\|$	$\|\mathbb{Z}_p^*\| + \|G_1\|$	Yes
Ours	4	1	0	0	0	0	3	1	2	$2\|\mathbb{Z}_p^*\| + 2\|G_1\| + \|G_2\|$	$\|\mathbb{Z}_p^*\| + 2\|G_1\| + \|m\|$	$\|G_1\|$	Yes

Theorem 2 In the random oracle model, our scheme is EUF-CMA secure under the q-SDH and mICDH assumptions.

Proof This proof is similar to the proof of Theorem 1. We can show that a forger in the EUF-CMA game implies a forger in a chosen messages and given identity attacks. By using the forking lemma [28] and the relationship between given identity attack and chosen identity attack [29], we can easily finish this proof. □

5.2 Performance

In this section, we compare the computational cost, offline storage, ciphertext size, private key size and security of our scheme with those of LTX [5] and LZZ [7] in Table 1. We denote by M the point multiplication in G_1, E the exponentiation in G_2 and P the pairing computation. The other operations are ignored in Table 1 since these operations take the most running time of the whole algorithm. $|x|$ denotes the number of bits of x. From Table 1, we know that both LTX and LZZ need one point multiplication in the *OnSC* algorithm. However, our scheme does not need any point multiplication, exponentiation or pairing operation in the *OnSC* algorithm. In addition, our scheme has less computational cost than LTX and LZZ in the *USC* algorithm. For the *OffSC* algorithm, the computational cost of our scheme is slightly higher than LTX and is lower than LZZ. For the offline storage, our scheme is slightly larger than LTX and is smaller than LZZ. For the ciphertext size and private key size, our scheme is shortest among the three schemes. Note that LTX was showed insecure in [6].

We give a quantitative analysis for offline storage, ciphertext size and private key size. We use PBC Type A pairing [30] in this analysis. The Type A pairing is constructed on the curve

$$y^2 \equiv (x^3 + x) \bmod q$$

for some prime $q \equiv 3 \bmod 4$, where the embedding degree is 2 and the order of G_1 is p. In this analysis, we use three kinds of parameters that represents 80-bit, 112-bit and 128-bit AES [31] key size security level, respectively. Table 2 gives the specification for different security level of this analysis.

Table 2 Specification for different security level of this analysis (bits)

Security level	Size of q	Size of p
80-bit	512	160
112-bit	1024	224
128-bit	1536	256

We assume that the size of a message is $|m| = 160$ bits. When we adopt the 80-bit security level, the size of q is 512 bits. So the size of an element in group G_1 is 1024 bits using an elliptic curve with 160 bits p. By standard compression technique [32], the size of an element in group G_1 can be reduced to 65 bytes. The size of an element in G_2 is 1024 bits. So, the offline storage of LTX, LZZ and our scheme are $|\mathbb{Z}_p^*| + 2|G_1| + |G_2|$ bits $= 20 + 2 * 65 +$ 128 bytes $= 278$ bytes, $3|\mathbb{Z}_p^*| + 4|G_1| + |G_2|$ bits $= 3 * 20 + 4 * 65 + 128$ bytes $= 448$ bytes and $2|\mathbb{Z}_p^*| + 2|G_1| + |G_2|$ bits $= 2 * 20 + 2 * 65 + 128$ bytes $= 298$ bytes, respectively. The ciphertext size of LTX, LZZ and our scheme are $2|\mathbb{Z}_p^*| + 2|G_1| + |m|$ bits $= 2 * 20 + 2 * 65 +$ 20 bytes $= 190$ bytes, $2|\mathbb{Z}_p^*| + 4|G_1| + |m|$ bits $= 2 * 20 + 4 * 65 +$ 20 bytes $= 320$ bytes, and $|\mathbb{Z}_p^*| + 2|G_1| + |m|$ bits $= 20 + 2 * 65 + 20$ bytes $= 170$ bytes, respectively. The private key size of LTX, LZZ and our scheme are $|\mathbb{Z}_p^*| + |G_1|$ bits $= 20 + 65$ bytes $= 85$ bytes, $|\mathbb{Z}_p^*| + |G_1|$ bits $= 20 + 65$ bytes $= 85$ bytes, and $|G_1|$ bits $= 65$ bytes, respectively. We can use the same method to compute the offline storage, ciphertext size and private key size at the 112-bit security level and 128-bit security level.

We summarize the offline storage, ciphertext size and private key size of the three schemes at different security level in Figs. 2, 3 and 4, respectively.

6 Application

In this section, we give an application of our scheme in the IoT. Wireless sensor networks (WSNs) are an important part of the IoT since the WSNs takes charge of collecting environmental data for the IoT. The WSNs are composed of a large number of tiny sensor nodes and one or more

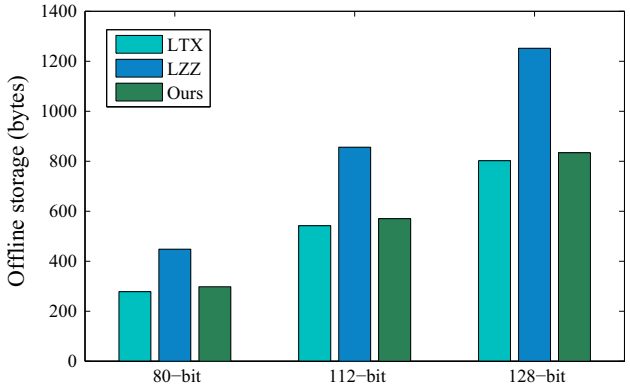


Fig. 2 The offline storage of the three schemes

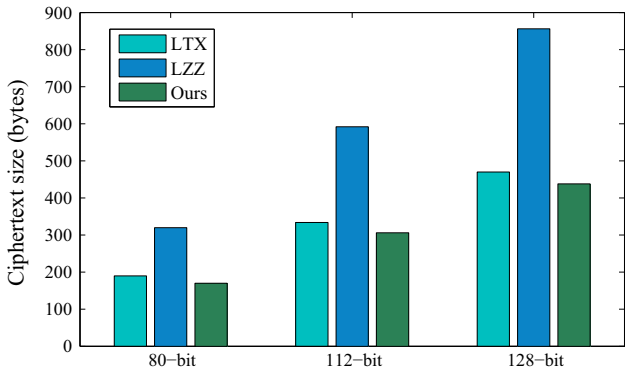


Fig. 3 The ciphertext size of the three schemes

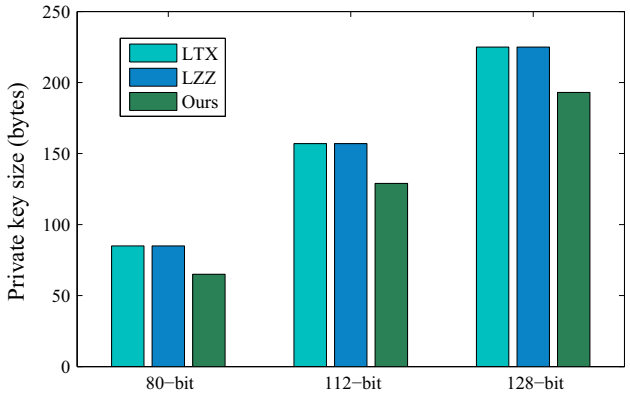


Fig. 4 The private key size of the three schemes

base stations [33, 34]. The base station acts as a gateway between sensor nodes and users since it typically forwards data from the WSNs to an Internet server. This communication from the WSNs to the server should satisfy confidentiality, authentication, integrity, and non-repudiation. Without confidentiality, the data may be disclosed to an adversary. Without authentication, the server can not use the data since the data may be unbelievable. An adversary

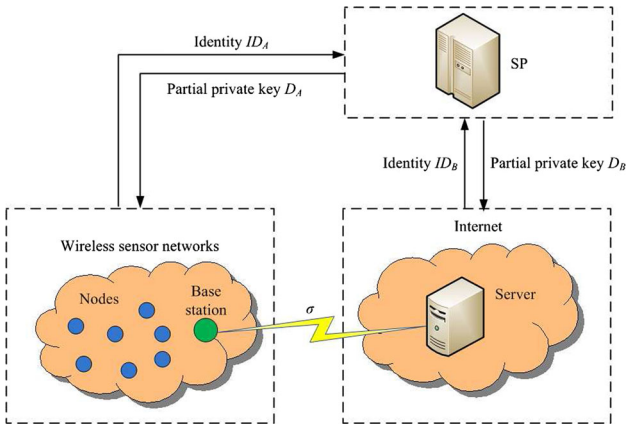


Fig. 5 A secure communication model for the IoT

can send wrong data to the server. Without integrity check, an adversary can modify the transmitted data. Without non-repudiation, the WSNs may deny the transmitted data when a dispute happens. Fig. 5 shows a secure communication model for the IoT using our scheme. This model consists of three main entities, the WSNs, a service provider (SP) and an Internet server. The SP acts as the KGC in the CLC. That is, the SP first runs *Setup* algorithm to setup the system parameters. Then the SP runs *PPKE* algorithm to generate the partial private keys for the base station and the SP. The base station and the server run *UKG* algorithm to generate their secret values and public keys. In addition, the base station and the server run *FPKS* algorithm to obtain their full private keys. The base station is loaded with the precomputed result δ from *OffSC* algorithm. When the WSNs is required to send data to the server, the base station runs *OnSC* algorithm and sends the ciphertext $\sigma = (c, \theta, S', T)$ to the server. When receiving the σ, the server runs *USC* algorithm to recover the data m and verify the validity. In this communication, the confidentiality, authentication, integrity, and non-repudiation are simultaneously achieved. The computational cost of base station is very small since there is no any point multiplication, exponentiation or pairing operation in the *OnSC* algorithm. If the data are large, we also can used hybrid encryption method [16]. That is, we compute $c = E_{H_3(r)}(m)$ instead of $c = m \oplus H_3(r)$. Here E is the encryption algorithm for a symmetric cipher (such as AES [31]) and $H_3(r)$ is the session key. Such modification does not affect the security and efficiency of our scheme.

7 Conclusion

In this paper, we proposed a new certificateless online/offline signcryption scheme and proved its security in the random oracle model. As compared with two existing

certificateless online/offline signcryption schemes, our scheme does not require any point multiplication operation in the online phase. This characteristic makes our scheme very suitable for resource-constrained devices. We gave an application of our scheme in the Internet of Things. A weakness of our scheme is that a receiver's identity is required in the offline phase. An interesting work is to find a certificateless online/offline signcryption scheme that does not need a receiver's identity in the offline phase and does not need any point multiplication operation in the online phase.

Acknowledgments This work is supported by the National Natural Science Foundation of China (Grant Nos. 61073176, 61272525, 61302161 and 61462048) and the Fundamental Research Funds for the Central Universities (Grant No. ZYGX2013J069).

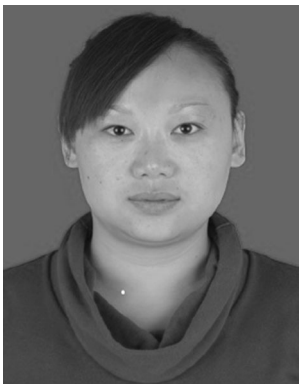
References

1. Tsai, C. W., Lai, C. F., & Vasilakos, A. V. (2014). Future Internet of Things: Open issues and challenges. *Wireless Networks*, *20*(8), 2201–2217.
2. Ning, H. S., & Liu, H. (2015). Cyber-physical-social-thinking space based science and technology framework for the Internet of Things. *Science China Information Sciences*, *58*(3), 031102(19).
3. Roman, R., Zhou, J., & Lopez, J. (2013). On the features and challenges of security and privacy in distributed Internet of Things. *Computer Networks*, *57*(10), 2266–2279.
4. Jing, Q., Vasilakos, A. V., Wan, J., Lu, J., & Qiu, D. (2014). Security of the Internet of Things: Perspectives and challenges. *Wireless Networks*, *20*(8), 2481–2501.
5. Luo, M., Tu, M., & Xu, J. (2014). A security communication model based on certificateless online/offline signcryption for Internet of Things. *Security and Communication Networks*, *7*(10), 1560–1569.
6. Shi, W., Kumar, N., Gong, P., Chilamkurti, N., & Chang, H. (2015). On the security of a certificateless online/offline signcryption for Internet of Things. *Peer-to-Peer Networking and Applications*, *8*(5), 881–885.
7. Li, J., Zhao, J., & Zhang, Y. (2015). Certificateless online/offline signcryption scheme. *Security and Communication Networks*, *8*(11), 1979–1990.
8. Zheng, Y. (1997). Digital signcryption or how to achieve cost (signature & encryption) $\ll$ cost (signature) + cost(encryption). In *Advances in Cryptology-CRYPTO'97, LNCS 1294* (pp. 165–179). Springer.
9. Malone-Lee, J., & Mao, W. (2003). Two birds one stone: Signcryption using RSA. In *Topics in Cryptology-CT-RSA 2003, LNCS 2612* (pp. 211–225). Springer.
10. Boyen, X. (2003). Multipurpose identity-based signcryption: A swiss army knife for identity-based cryptography. In *Advances in Cryptology-CRYPTO 2003, LNCS 2729* (pp. 383–399). Springer.
11. Chen L., & Malone-Lee, J. (2005). Improved identity-based signcryption. In *Public Key Cryptography-PKC 2005, LNCS 3386* (pp. 362–379). Springer.
12. Barreto, P.S.L.M., Libert, B., McCullagh, N., & Quisquater, J.J. (2005). Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In *Advances in Cryptology-ASIACRYPT 2005, LNCS 3788* (pp. 515–532). Springer.
13. Jo, H. J., Paik, J. H., & Lee, D. H. (2014). Efficient privacy-preserving authentication in wireless mobile networks. *IEEE Transactions on Mobile Computing*, *13*(7), 1469–1481.
14. Barbosa, M., & Farshim, P. (2008). Certificateless signcryption. *ACM Symposium on Information, Computer and Communications Security-ASIACCS 2008* (pp. 369–372). Japan: Tokyo.
15. Li, F., Shirase, M., & Takagi, T. (2013). Certificateless hybrid signcryption. *Mathematical and Computer Modelling*, *57*(3–4), 324–343.
16. Yin, A., & Liang, H. (2015). Certificateless hybrid signcryption scheme for secure communication ofwireless sensor networks. *Wireless Personal Communications*, *80*(3), 1049–1062.
17. An, J.H., Dodis, Y., & Rabin, T. (2002). On the security of joint signature and encryption. In *Advances in Cryptology-EUROCRYPT 2002, LNCS 2332* (pp. 83–107). Springer.
18. Zhang, F., Mu, Y., & Susilo, W. (2005). Reducing security overhead for mobile networks. *Advanced Information Networking and Applications-AINA 2005* (pp. 398–403). Taiwan: Taipei.
19. Xu, Z., Dai, G., & Yang, D. (2007). An efficient online/offline signcryption scheme for MANET. *Advanced Information Networking and Applications Workshops-AINAW 2007* (pp. 171–176). Canada: Niagara Falls.
20. Yan, F., Chen, X., & Zhang, Y. (2013). Efficient online/offline signcryption without key exposure. *International Journal of Grid and Utility Computing*, *4*(1), 85–93.
21. Sun, D., Huang, X., Mu, Y., & Susilo, W. (2008). Identity-based on-line/off-line signcryption. *IFIP International Conference on Network and Parallel Computing* (pp. 34–41). China: Shanghai.
22. Liu, J.K., Baek, J., & Zhou, J. (2011). Online/offline identity-based signcryption re-visited. In *Information Security and Cryptology-Inscrypt 2010, LNCS 6584* (pp. 36–51). Springer.
23. Li, F., Khan, M. K., Alghathbar, K., & Takagi, T. (2012). Identity-based online/offline signcryption for low power devices. *Journal of Network and Computer Applications*, *35*(1), 340–347.
24. Li, F., & Xiong, P. (2013). Practical secure communication for integrating wireless sensor networks into the Internet of Things. *IEEE Sensors Journal*, *13*(10), 3677–3684.
25. Senthil kumaran, U., & Ilango, P. (2015). Secure authentication and integrity techniques for randomized secured routing in WSN. *Wireless Networks*, *21*(2), 443–451.
26. Al-Riyami, S.S., & Paterson, K.G. (2003). Certificateless public key cryptography. In *Advances in Cryptology-ASIACRYPT 2003, LNCS 2894* (pp. 452–474). Springer.
27. Boneh, D., & Boyen, X. (2004). Short signatures without random oracles. In *Advances in Cryptology-EUROCRYPT 2004, LNCS 3027* (pp. 56–73). Springer.
28. Pointcheval, D., & Stern, J. (2000). Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, *13*(3), 361–396.
29. Cha, J.C., & Cheon, J.H. (2003). An identity-based signature from gap Diffie–Hellman groups. In *Public Key Cryptography-PKC 2003, LNCS 2567* (pp. 18–30). Springer.
30. PBC Library. http://crypto.stanford.edu/pbc/
31. Daemen, J., & Rijmen, V. (2002). *The design of Rijndael: AES-the advanced encryption standard*. Berlin: Springer.
32. Shim, K. A. (2012). CPAS: An efficient conditional privacy-preserving authentication scheme for vehicular sensor networks. *IEEE Transactions on Vehicular Technology*, *61*(4), 1874–1883.

33. Ferng, H. W., Nurhakim, J., & Horng, S. J. (2014). Key management protocol with end-to-end data security and key revocation for a multi-BS wireless sensor network. *Wireless Networks*, *20*(4), 625–637.
34. Chatterjee, P., Ghosh, U., Sengupta, I., & Ghosh, S. K. (2014). A trust enhanced secure clustering framework for wireless ad hoc networks. *Wireless Networks*, *20*(7), 1669–1684.

Fagen Li is an associate professor in the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, P.R. China. He received his Ph.D. degree in Cryptography from Xidian University, Xi'an, P.R. China in 2007. From 2008 to 2009, he was a postdoctoral fellow in Future University-Hakodate, Hokkaido, Japan, which is supported by the Japan Society for the Promotion of Science (JSPS). He worked as a research fellow in the Institute of Mathematics for Industry, Kyushu University, Fukuoka, Japan from 2010 to 2012. His recent research interests include cryptography and network security. He has published more than 70 papers in the international journals and conferences. He is a member of the IEEE.

Yanan Han received her B.S. degree from Henan Agricultural University, Zhengzhou, P.R. China in 2013. She is now a master student in the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, P.R. China. Her research interests include cryptography and information security.



Chunhua Jin is now a Ph.D. student in the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, P.R. China. Her research interests include cryptography and network security.