ISSN 1392–124X (print), ISSN 2335–884X (online) INFORMATION TECHNOLOGY AND CONTROL, 2016, T. 45, Nr. 3

A Short Certificate-based Signature Scheme with Provable Security

Ying-Hao Hung, Sen-Shan Huang, Yuh-Min Tseng*

Department of Mathematics, National Changhua University of Education,
Jin-De Campus, Changhua 500, Taiwan, R.O.C.
e-mail: ymtseng@cc.ncue.edu.tw

crossref http://dx.doi.org/10.5755/j01.itc.45.3.12814

Abstract. Certificate-based signature (CBS) is an attractive paradigm since it simultaneously solves the certificate revocation problem in conventional signatures and the key escrow problem in ID-based signatures. In particular, short certificate-based signatures are useful in bandwidth reduction for communication due to their short signature lengths. However, it is still a challenging and open problem to design a secure short certificate-based signature (SCBS) scheme. Recently, to solve this problem, Li et al. proposed an efficient SCBS scheme. However, in this article, we will show that Li et al.'s scheme is insecure against Type I adversary (i.e. uncertified entity) under an accredited security model. Moreover, we propose a new SCBS scheme with provable security. Based on the computational Diffie–Hellman (CDH) assumption, we demonstrate that our SCBS scheme possesses existential unforgeability against adaptive chosen-message attacks under the same accredited security model. When compared with previous SCBS schemes, our scheme is the first provably secure SCBS scheme while retaining efficiency.

Keywords: Short signature; Certificate-based signature; Existential unforgeability.

1. Introduction

In conventional public-key system (PKS) settings [1, 2], a trusted certificate authority (CA) issues public key certificates to provide un-forgeable and trusted links between the identities and the public keys of the users. Therefore, a public key infrastructure (PKI) is required to manage and maintain certificates of all the users. In such a system, anyone who wants to verify signatures of other entities must verify their authorized public key certificates beforehand to ensure that these public keys are still valid. Hence, the certificate management is generally considered to be costly, when adopted in a PKI.

To simplify the certificate management, Shamir [3] introduced the concept of identity (ID)-based public-key system (ID-PKS) setting. Until 2001, Boneh and Franklin [4] proposed the first practical ID-PKS setting and ID-based encryption (IBE) scheme. In their ID-PKS setting, a user's public key is determined by some identity information such as social security number, e-mail address, and name. This avoids the necessity of certificates, and associates an implicitly verified public key to each user. A trusted private key generator (PKG) with a master secret key is responsible to generate and send each user a private key via a secure channel. However, since the PKG knows each user's private key, it can generate signatures on behalf of any user, or decrypt any ciphertexts sent to any user. Hence, the key escrow property becomes an inherent problem in the ID-PKS setting, so that ID-PKS setting is only suitable for a closed organization where the PKG is fully trusted by everyone in the group.

To solve the key escrow problem in the ID-PKS setting, Al-Riyami and Paterson [5] presented a new paradigm, called certificateless PKS (CL-PKS) setting, which eliminates the usage of certificates in the conventional PKS settings. In the CL-PKS setting, there are two roles, namely, the key generation center (KGC) and users. A user independently generates a public/secret key pair, and the KGC generates a partial private key in accordance with the identity of the user and then sends it to the user via a secure channel. To decrypt a message, a user requires both her/his secret key and the associated partial private key. Note that the KGC does not know any user's secret key so that it is unable to impersonate a user or decrypt the ciphertexts sent to a user. Therefore, the CL-PKS setting solves the key escrow problem in the ID-PKS setting by eliminating the usage of certificates in the conventional PKS setting. However, due to the lack of public key certificates, both the ID-PKS and CL-PKS settings must provide additional revocation

* Corresponding author

mechanisms [6−9]. In 2003, Gentry [10] introduced the notion of certificate-based PKS (CB-PKS) setting, which resolves the inherent key escrow problem in ID-PKS setting and the certificate revocation problem in the conventional PKS. In Gentry's scheme, a certificate acts as a partial private key as well. A user independently generates her/his public/secret key pair and sends the public key to a trusted certificate authority (CA). Then the CA generates a certificate for the user by the user's public key and some additional identity information

Table 1. Comparisons among the conventional PKS, ID-PKS, CL-PKS and CB-PKS settings

	PKS [1, 2]	**ID-PKS [4]**	**CL-PKS [5]**	**CB-PKS [10]**
Averting key escrow problem	Yes	No	Yes	Yes
Required channel for revocation	Public	secure	secure	Public
Level of trust placed on the CA/PKG/KGC	Low	High	Middle	Low
Certificate validation before encrypting and verifying	Required	Not required	Not required	Not required

Since then, the CA updates the certificate periodically. To sign a message or decrypt a ciphertext, the user requires both her/his secret key and an up-to-date certificate. Since the CA is unable to obtain the secret key of any user, the key escrow problem will not take place in the CB-PKS setting. Meanwhile, CB-PKS setting also solves the certificate revocation problem.

Table 1 lists the comparisons among the conventional PKS [1, 2], ID-PKS [4], CL-PKS [5], and CB-PKS [10] settings in terms of averting the key escrow problem, revocable functionality, the level of trust placed on the CA/PKG/KGC and certificate validation before encrypting and verifying. It is obvious that the ID-PKS setting suffers from the key escrow problem. For the level of trust placed on the CA/PKG/KGC, the CB-PKS setting is better than the others. Except for the conventional PKS and the CB-PKS settings, the other settings remove the need of certificates. In the conventional PKS settings, the users can verify illegal or compromised users by referring to the certificate revocation list (CRL). In the ID-PKS and CL-PKS settings, the PKG/KGC adopts a secure channel to transmit the private keys to non-revoked users periodically [4, 5]. In the CB-PKS setting, the CA updates the certificates via a public channel. According to Table 1, the CB-PKS construction possesses the advantages of both ID-PKS (implicit certification) and conventional PKS (no escrow) settings while it does not need a secure channel for revocation. In the conventional PKS setting, before verifying a signature by the signer's public key, one needs to verify the signer's certificate issued by the CA to guarantee the validation of the signer's public key. Due to additional computation time and storage, the certificates in conventional PKS settings are costly to use and manage.

1.1. Motivation

Digital signature is one important cryptographic primitive, which provides the integrity, authentication and non-repudiation of messages. Indeed, authentication (identification) schemes [11−15] may be implemented by employing signature schemes. With the rapid growth of wireless communications, clients (users) often use handheld wireless devices to access remote servers via open network channels. When cryptographic mechanisms are involved in wireless environments, these wireless devices are generally resource-constrained because they possess low-power energy and limited computing capability. In this case, numerous data bits of communication and cryptographic operations with expensive computations would become heavy load for wireless devices. Hence, it is a critical issue to alleviate the communication and computational load of wireless devices. For wireless devices such as smart card, PDA, cell phone, RFID chip and sensor, message communication consumes more time and energy than computation does. To transmit one bit of data requires more energy than to execute one 32-bit instruction [16]. Therefore, reducing the number of communication bits becomes an important issue for cryptographic mechanisms executed on wireless devices.

Since the usage of short signature aims at the reduction of communication bandwidth, it is suitable for wireless environments. In 2001, Boneh *et al*. [17] constructed the first short signature scheme from bilinear pairings in conventional PKS setting. It is half the size of a DSA signature with a similar level of security. Afterwards, several concrete short signature schemes [18, 19] were proposed in the standard model (without random oracles). On the other hand, several researchers [20−23] also proposed short certificateless signature schemes in CL-PKS setting. However, it is still a challenging and open problem to design a secure short certificate-based signature (SCBS). Recently, to solve this problem, Li *et al*. [24] proposed an efficient SCBS scheme. However, in this article, we will show that Li *et al*.'s scheme is insecure against Type I adversary (i.e. uncertified entity). Moreover, we will propose the first provably secure SCBS scheme.

1.2. Related work

Following Gentry's [10] concept, in 2004, Kang *et al*. [25] proposed the first certificate-based signature (CBS) scheme which is derived from bilinear pairings on elliptic curves. In 2007, Li *et al*. [26] introduced the *key replacement attack* on the CB-PKS setting and demonstrated Kang *et al*.'s scheme is insecure against key replacement attacks. Since the key replacement

attacks might occur in the CB-PKS setting, Li *et al*. [26] redefined Kang *et al*.'s security model by additionally addressing key replacement attacks and proposed an improved version. In 2008, Liu *et al*. [27] furthermore proposed two new CBS schemes. One of them did not require any pairing operations in the random oracle model and the other was secure without random oracle. Unfortunately, Zhang *et al*. [28] showed that Liu *et al*.'s first scheme was insecure and proposed an improved scheme. Meanwhile, Wu *et al*. [29] also proposed an improved CBS scheme. However, the signature sizes of all the CBS schemes mentioned above are more than one group element.

In 2011, Liu *et al*. [30] proposed the first short certificate-based signature (SCBS) scheme. Unfortunately, Cheng *et al*. [31] showed that their scheme is insecure against a Type I adversary under an accredited security model defined in [26-28]. Recently, Li *et al*. [24] proposed a new SCBS scheme. However, the security model of Li *et al*.'s scheme is weaker than the accredited security model [26−28] in the sense that a Type I adversary (i.e. uncertified entity) can extract neither singer's secret key nor certificate of a target entity. Hence, it is still a challenging and open problem to design a secure SCBS scheme under the accredited security model.

1.3. Contribution

In this article, we first show that Li *et al*.'s SCBS scheme [24] is insecure against a Type I adversary under the accredited security model in [26−28]. To achieve our goal, we redefine the framework of SCBS schemes, in which the public key of a user is determined by both the user and the CA. In the accredited security model [26−28], an adversary can extract either a singer's secret key or the associated certificate of a target entity. Finally, we propose the first provably secure SCBS signature scheme in the random oracle model [32, 33]. Our scheme has the following features. Firstly, as compared with the previously proposed CBS and SCBS schemes, our scheme enjoys lower communication bandwidth while retaining computation efficiency. Secondly, the proposed scheme possesses existential unforgeability against adaptive chosen-message attacks under the computational Diffie–Hellman (CDH) assumption. Lastly, to the best of our knowledge, our scheme is the first provable secure SCBS scheme under the accredited security model in [26−28].

The remainder of the article is organized as follows. In Section 2, we give a brief introduction for bilinear pairings and security assumption. In Section 3, we redefine the framework and security notions for SCBS schemes. In section 4, we review Li *et al*.'s SCBS scheme and show how a Type I adversary can successfully attack their scheme. Our concrete SCBS scheme is given in Section 5. In Section 6, we analyze the security of our scheme. Comparisons are demonstrated in Section 7. Finally, we draw a conclusion in Section 8.

2. Preliminaries

In this section, we briefly present some properties of bilinear pairings and a relevant security assumption [4].

2.1. Bilinear pairings

Let G and G_T be additive and multiplicative cyclic groups of the same prime order q, respectively. An admissible bilinear pairing is a map $\hat{e}$: $G\times G \rightarrow G_T$ with the following properties:

1. Bilinearity: for $P, Q\in G$ and $a, b\in Z_q^*$, the map $\hat{e}$ satisfies the equality $\hat{e}(aP, bQ)=\hat{e}(P, Q)^{ab}$.
2. Non-degeneracy: there exist $P, Q\in G$ such that $\hat{e}(P, Q)\neq 1$.
3. Computability: for all $P, Q\in G$, $\hat{e}(P, Q)$ can be computed efficiently.

2.2. Security assumption

Here, we present a hard mathematical problem and define its corresponding assumption.

- *Computational Diffie-Hellman* (CDH) *problem*: given P, aP, $bP\in G$ with uniformly random choices of $a, b\in Z_q^*$, the CDH problem is to compute abP.

Definition 1. The CDH assumption in G is defined as follows. Given P, aP, $bP\in G$ with uniformly random choices of a, $b\in Z_q^*$, there exists no probabilistic polynomial-time adversary (PPT) $\mathcal{A}$ with non-negligible probability who can compute abP. The successful probability (advantage) of the adversary $\mathcal{A}$ is presented as

$$\mathrm{Adv}_{\mathcal{A}} = \Pr[\mathcal{A}(P, aP, bP)=abP: P\in G, a, b\in Z_q^*].$$

3. Framework and adversarial model of SCBS

3.1. Framework

We present the framework of short certificate-based signature (SCBS) schemes depicted in Fig.1, which is modified from that of the CBS scheme in [26−28]. Our framework is slightly different from that of the conventional CBS schemes in the sense that the full public key of a user is generated by both the user and the CA. A SCBS scheme is specified by five algorithms, namely, *Setup*, *User key generation*, *Certificate generation*, *Sign* and *Verify* algorithms.

- *Setup* is a probabilistic algorithm run by the CA that takes a security parameter as input, and returns a master secret key and public parameters *PP*. *PP* is made public and available for all the other algorithms.
- *User key generation* is a probabilistic algorithm run by a user that takes as input the user's identity

ID, and outputs the secret key S_{ID} and the partial public key P_{ID}.

- *Certificate generation* is a probabilistic algorithm run by the CA that takes as input the master secret key, the public parameters *PP*, a user's identity *ID* and partial public key P_{ID}, and returns the user's certificate C_{ID} and (full) public key $PK_{ID}=(P_{ID}, R_{ID})$ to the user, and publishes (ID, PK_{ID}) in a public directory.
- *Sign* is a deterministic algorithm run by a user that takes as input the user's secret key S_{ID}, certificate C_{ID} and a message *M*, and returns a signature σ.
- *Verify* is a deterministic algorithm that takes as input a message *M*, a signature σ, a user's identity *ID* with the public key PK_{ID}, and outputs either "accept" or "reject".

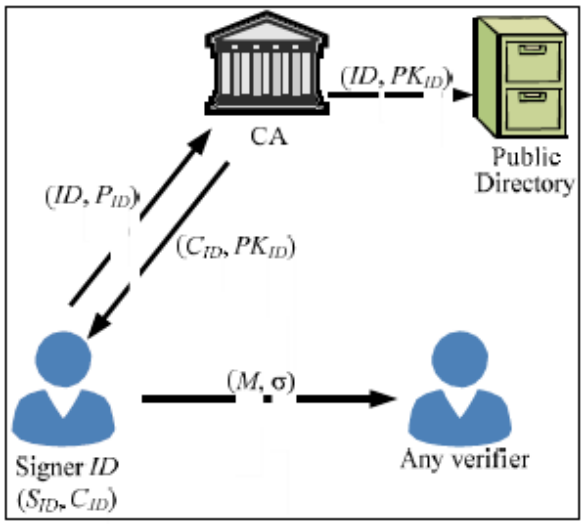

Figure 1. Framework of SCBS schemes

3.2. Security model

Based on the security model of CBS schemes in [26−28], we define "existential unforgeability of short certificate-based signatures against adaptive chosen-message attacks" (UF-SCBS-ACM). In UF-SCBS-ACM attacks, the security notions for SCBS schemes include two types of adversaries, namely, Type I and Type II adversaries with different query capabilities. A Type I adversary A_I acts as an uncertified entity who does not have access to the master secret key *s* so that it cannot obtain the certificate of a target entity. A Type II adversary A_{II} models the malicious CA who owns the master secret key *s*, but cannot obtain the secret key of the target entity. The security notions for SCBS schemes are modeled by the following two games **(Games 1 and 2)** between a challenger *C* and the two types of adversaries.

Definition 2. (UF-SCBS-ACM**).** A SCBS scheme offers existential unforgeability against adaptive chosen-message (UF-SCBS-ACM) attacks if no PPT adversary $\mathcal{A}$ of Type I or Type II has a non-negligible advantage in the following two games (Games 1 and 2) played between a challenger *C* and the adversary $\mathcal{A}$.

Game 1 (for Type I adversary, AI)

- *Setup*. The challenger *C* runs the *Setup* algorithm to produce a master secret key *s* and a list of public parameters *PP*. *PP* is given to A_I and *s* is kept secret by *C*.
- *Queries*. The adversary A_I may issue a number of different queries to the challenger *C* in an adaptive manner as follows.
  - *User key generation* (*ID*). The challenger *C* runs the *User key generation* algorithm to return the user's partial public key P_{ID} to A_I.
  - *Certificate generation* (*ID*, P_{ID}). The challenger *C* runs the *Certificate generation* algorithm to return the user's certificate C_{ID} and public key $PK_{ID}=(P_{ID}, R_{ID})$ to A_I.
  - *Corruption* (*ID*). Upon receiving this query, the challenger *C* returns the user's secret key S_{ID} to A_I.
  - *Public key replacement* (*ID*, PK'_{ID}). The adversary A_I chooses a new public key $PK'_{ID}=(P'_{ID}, R'_{ID})$ for the user with identity *ID*. The challenger *C* records this replacement.
  - *Sign* (*m*, *ID*). Upon receiving this query on (*m*, *ID*) under the public key $PK_{ID}= (P_{ID}, R_{ID})$, the challenger *C* generates a valid signature σ and returns it to A_I even though the challenger *C* does not hold the secret key and the certificate of the user with identity *ID*.
- *Forgery*. The adversary A_I generates a signature tuple $(m^*, \sigma^*, ID^*, PK_{ID^*})$, where PK_{ID^*} is the original public key without replacement. The advantage of the adversary A_I is defined as the probability that A_I wins Game 1. We say that the adversary A_I wins Game 1 if the following conditions are all satisfied.
  1. The response of the *Verification* algorithm on $(m^*, \sigma^*, ID^*, PK_{ID^*})$ is "accept".
  2. (m^*, ID^*) has never been submitted during the *sign* query.
  3. ID^* has never been submitted during the *Certificate generation* query.

Game 2 (for Type II adversary, AII)

- *Setup*. The challenger *C* runs the *setup* algorithm to produce a master secret key *s* and a list of public parameters *PP*. *PP* and *s* are sent to A_{II}.
- *Queries*. Since the adversary A_{II} knows the master secret key, it can compute the certificates and the public keys of all the users. The adversary A_{II} may make a number of queries as in **Game 1**, except the

certificate generation query because it can compute the certificates of all the users.

- *Forgery*. The adversary A_{II} generates a signature tuple (m^*, σ^*, ID^*, PK_{ID^*}), where PK_{ID^*} is the original public key without replacement. The advantage of the adversary A_{II} is defined as the probability that A_{II} wins Game 2. We say that the adversary A_{II} wins Game 2 if the following conditions are all satisfied.
  1. The response of the *Verification* algorithm on (m^*, σ^*, ID^*, PK_{ID^*}) is "accept".
  2. (m^*, ID^*) has never been submitted during the *sign* query.
  3. ID^* has never been submitted during the *Corruption* query.

4. Review and weakness of Li *et al.*'s SCBS scheme

Recently, Li *et al*. [24] presented a SCBS scheme and claimed their scheme is secure against both Type I and Type II adversaries in the random oracle model. However, the security model of Li *et al*.'s scheme is weaker than the accredited security model in [26−28]. In their model, a Type I adversary (i.e. uncertified entity) extracts neither the secret key nor the certificate of a target entity. On the contrary, the accredited security model allows an adversary to obtain either the secret key (Type I adversary) or the certificate (Type II adversary) of a target entity as described in Games 1 and 2 in Section 3. In this section, we show that their scheme is insecure in the presence of Type I adversary under the accredited security model. In the following, we first review Li *et al*.'s SCBS scheme.

4.1. Li *et al.*'s SCBS scheme

Li *et al*.'s SCBS scheme consists of five algorithms:

- *Setup*: Given a security parameter 1^k, the CA generates a list of public parameter $PP=<G, G_T, e, q, P, Q, mpk, H_0, H_1>$ by performing the following steps.
  1. Let (G, G_T) be a bilinear group pair of a prime order q, and let e: $G \times G \rightarrow G_T$ be a bilinear map.
  2. Select a random number $s \in Z_q^*$ and two random elements $P, Q \in G$. Set s as the master secret key msk, and $s \cdot P$ as the system public key mpk.
  3. Choose two cryptographic hash functions H_0:$\{0, 1\}^* \times G \times G \rightarrow G_T$ and H_1 : $\{0, 1\}^* \rightarrow Z_q^*$.
- *User key generation*: Given PP, a user with identity ID randomly selects a secret key $s_{ID} \in Z_q^*$ and computes the corresponding public key $PK_{ID}=(PK^1_{ID}, PK^2_{ID})$, where $PK^1_{ID}=s_{ID} \cdot P$ and $PK^2_{ID}=s_{ID} \cdot Q$.
- *Certificate generation*: Given PP, the master secret key s, a public key PK_{ID} of the user with identity ID, the CA computes $Q_{ID}=H_0(ID, PK_{ID})$ and the certificate $Cert_{ID}=s \cdot Q_{ID}$.
- *Sign*: Taking as input PP, ID, a secret key s_{ID}, a certificate $Cert_{ID}$ and a message $m \in \{0, 1\}^*$, a user calculates the signature $\sigma=(1/(H_1(m)+s_{ID})) \cdot Cert_{ID}$.
- *Verify*: Given a signature pair (m, σ), the public parameter PP, an identity ID and a public key $PK_{ID}=(PK^1_{ID}, PK^2_{ID})$, a verifier calculates $Q_{ID}=H_0(ID, PK_{ID})$. This algorithm outputs "accept" if both equalities $e(\sigma, H_1(m)P+PK^1_{ID})=e(mpk, Q_{ID})$ and $e(PK^1_{ID}, Q)=e(PK^2_{ID}, P)$ hold. Otherwise, it outputs "reject".

4.2. Weakness of Li *et al.*'s SCBS scheme

Here, we point out that Li *et al*.'s scheme is not secure against Type I adversary A_I in the accredited security model of CBS schemes [26−28]. In Game 1, A_I cannot issue the *Certificate generation* query on a target entity with identity ID^*, but can obtain the associated secret key s_{ID^*} by making the *Corruption* query on ID^*. Then, we show that A_I can obtain both the certificate $Cert_{ID^*}$ and the associated secret key s_{ID^*} of the target entity by the following steps.

1. First, A_I makes the *User key generation* query on a target user with identity ID^* to obtain the user's public key $PK^*_{ID}=(s_{ID^*} \cdot P, s_{ID^*} \cdot Q)$.
2. Next, A_I issues the *Corruption* query to obtain the user's secret key s_{ID^*}. Then A_I makes the *Sign* query to obtain a valid signature $\sigma^*=(1/(H_1(m^*)+ s_{ID^*})) \cdot Cert_{ID^*}$ on the message m with respect to the public key PK^*_{ID}.
3. Finally, A_I can obtain the certificate $Cert_{ID^*}=\sigma^*/(1/H_1(m^*)+ s_{ID^*})$.

By above, A_I can generate valid signatures on any messages on behalf of the target user with identity ID^*. Therefore, A_I can win the Game 1 so that Li *et al.*'s SCBS scheme is insecure under the aforementioned security model. Indeed, Cheng *et al*. [31] have presented the same attack on Liu *et al*.'s SCBS scheme [30] and pointed out that it is still a challenging and open problem to design a secure SCBS scheme.

5. Our SCBS scheme

The proposed SCBS scheme consists of five algorithms, namely, *Setup*, *User key generation*, *Certificate generation*, *Sign* and *Verify* algorithms.

- *Setup*: Given a security parameter l, the CA first generates a bilinear group pair (G, G_T) of a prime order $q > 2^l$, an admissible bilinear map $\hat{e}$: $G \times G \rightarrow G_T$, and an arbitrary generator P of G. Next, the CA randomly chooses a master secret key $s \in Z_q^*$ and sets $P_{pub} = s \cdot P$ as the system public key.

  The CA picks three hash functions f: $\{0, 1\}^* \rightarrow Z_q^*$, H_1 and H_2: $\{0, 1\}^* \rightarrow G$. A list of public parameters are presented as $PP = <G, G_T, q, \hat{e}, P, P_{pub}, f, H_1, H_2>$.
- *User key generation*: A user with identity ID randomly selects a secret key $S_{ID} \in Z_q^*$ and sets $P_{ID}=S_{ID} \cdot P$ as her/his partial public key.
- *Certificate generation*: Given a user's identity ID and partial public key P_{ID}, the CA randomly chooses $r_{ID} \in Z_q^*$, and then computes $R_{ID}=r_{ID} \cdot P$, $h=f(ID, P_{ID}, R_{ID}, P_{pub})$ and the corresponding certificate $C_{ID}=r_{ID}+s \cdot h$. The CA returns the certificate C_{ID} and the public key $PK_{ID}=(P_{ID}, R_{ID})$ to the user.
- *Sign*: To sign a message $m \in \{0, 1\}^*$, the signer with a secret key S_{ID} and a certificate C_{ID} computes $U_1=H_1(m, ID, PK_{ID}, P_{pub})$ and $U_2=H_2(m, ID, PK_{ID}, P_{pub})$ and generates the corresponding signature $\sigma=S_{ID} \cdot U_1+C_{ID} \cdot U_2$.
- *Verify*: Given a signature tuple (m, σ, ID, PK_{ID}) with $PK_{ID}=(P_{ID}, R_{ID})$, a verifier performs the following steps to validate the signature tuple:
  1. Compute $h=f(ID, P_{ID}, R_{ID}, P_{pub})$, $U_1=H_1(m, ID, PK_{ID}, P_{pub})$ and $U_2=H_2(m, ID, PK_{ID}, P_{pub})$.
  2. Verify the equality $\hat{e}(P, \sigma)=\hat{e}(P_{ID}, U_1) \cdot \hat{e}(R_{ID}+h \cdot P_{pub}, U_2)$. If it holds, output "accept". Otherwise, output "reject". Here, the correctness of the equality in Step (2) of the *Verify* algorithm follows since

$$\begin{aligned}\hat{e}(P, \sigma) &= \hat{e}(P, S_{ID} \cdot U_1+C_{ID} \cdot U_2)\\ &= \hat{e}(P, S_{ID} \cdot U_1) \cdot \hat{e}(P, C_{ID} \cdot U_2)\\ &= \hat{e}(S_{ID} \cdot P, U_1) \cdot \hat{e}(P, (r_{ID}+s \cdot h) \cdot U_2)\\ &= \hat{e}(S_{ID} \cdot P, U_1) \cdot \hat{e}((r_{ID}+s \cdot h) \cdot P, U_2)\\ &= \hat{e}(P_{ID}, U_1) \cdot \hat{e}(R_{ID}+h \cdot P_{pub}, U_2).\end{aligned}$$

6. Security analysis

In this section, we give the security analysis of the proposed SCBS scheme. Theorems 1 and 2, respectively, show that the proposed scheme is secure against Type I and Type II adversaries in the UF-SCBS-ACM games (Games 1 and 2) presented in Definition 3.

Theorem 1. *Suppose that the hash functions H_1, H_2 and f are random oracles. Assume that a Type I UF-SCBS-ACM adversary $\mathcal{A}$ can break the proposed SCBS scheme with non-negligible advantage ε within running time τ. Then, there exists an algorithm $\mathcal{C}$ to solve the CDH problem with a non-negligible advantage*

$$\varepsilon' \geq (1-1/q_c)(1-1/(1+q_s))^{q_s}(1/(q_c(1+q_s)))\varepsilon$$

within running time $\tau' = \tau + O(q_c+q_s+q_u+q_1+q_2)\tau_1$, where τ_1 is the time to perform a scalar multiplication in G. Also, q_c, q_s, q_u, q_1, and q_2, denote, respectively, the maximal numbers of $\mathcal{A}$'s queries to the *Certificate generation* oracle, the *Sign* oracle, the *User key generation* oracle, the random oracle H_1, and the random oracle H_2. $\mathcal{A}$ can, respectively, ask q_c queries to the *Certificate generation* oracle, q_1 queries to the random oracle H_1, q_2 queries to the random oracle H_2, q_u queries to the *User key generation* oracle, and q_s queries to the *Sign* oracle.

Proof. Assume that an algorithm $\mathcal{C}$ is given a random instance (P, aP, bP) of the CDH problem, where P, aP, $bP \in G$ with unknown a, $b \in Z_q^*$. Let $J=abP$ be the solution of the CDH problem. The algorithm $\mathcal{C}$ finds J by interacting with the adversary $\mathcal{A}$ as follows.

- *Setup*. The challenger $\mathcal{C}$ creates a list of public parameters $PP=\{G, G_T, q, \hat{e}, P, P_{pub}, f, H_1, H_2\}$ by setting $P_{pub}=aP$. The challenger $\mathcal{C}$ gives $\mathcal{A}$ the public parameters PP. Here, the hash functions f, H_1 and H_2 are random oracles controlled by the challenger $\mathcal{C}$. To be consistent and collision-free, the challenger $\mathcal{C}$ maintains four initially empty lists L_f, L_1, L_2, and L_C to record the responses of the query oracles.
- *Queries*: The challenger $\mathcal{C}$ first randomly chooses $t \in [1, q_c]$ and lets ID' denote the target identity of the t-th query to the *Certificate generation* oracle. The challenger $\mathcal{C}$ answers the queries issued by $\mathcal{A}$ as follows.
  - *User key generation* (ID): $\mathcal{C}$ maintains an initially empty list P^{list} of tuples (ID, P_{ID}, S_{ID}). If ID's corresponding tuple (ID, P_{ID}, S_{ID}) exists in P^{list}, $\mathcal{C}$ return P_{ID} as answer. Otherwise, $\mathcal{C}$ randomly chooses a value $S_{ID} \in Z_q^*$, computes $P_{ID}=S_{ID} \cdot P$, returns P_{ID} to the adversary $\mathcal{A}$, and adds (ID, P_{ID}, S_{ID}) to P^{list}. Additionally, if $ID=ID'$, the challenger $\mathcal{C}$ randomly chooses $r_{ID} \in Z_q^*$, sets $R_{ID}=r_{ID} \cdot P$, and stores the tuple $(ID, P_{ID}, R_{ID}, \perp)$ in the initially empty list L_C.
  - $f(ID, P_{ID}, R_{ID})$: $\mathcal{C}$ maintains an initially empty list L_f of tuples (ID, P_{ID}, R_{ID}, h). If there exists a tuple (ID, P_{ID}, R_{ID}, h) in L_f related to the query $f(ID, P_{ID}, R_{ID})$, $\mathcal{C}$ returns h as answer. Otherwise, $\mathcal{C}$ randomly chooses $h \in Z_q^*$ and

returns it to the adversary $\mathcal{A}$. Then, the challenger C adds (ID, P_{ID}, R_{ID}, h) to L_f.

- $H_1(m, ID, PK_{ID})$: C maintains an initially empty list L_1 of tuples (m, ID, PK_{ID}, w, U_1). If there exists a tuple (m, ID, PK_{ID}, w, U_1) in L_1 related to the query $H_1(m, ID, PK_{ID})$, C returns U_1 to $\mathcal{A}$. Otherwise, C randomly chooses $w \in Z_q^*$, sets $U_1 = w \cdot P$, returns U_1 to $\mathcal{A}$, and adds (m, ID, PK_{ID}, w, U_1) to L_1.
- $H_2(m, ID, PK_{ID})$: C maintains an initially empty list L_2 of tuples (m, ID, PK_{ID}, u, U_2, $coin$). If there exists a tuple (m, ID, PK_{ID}, u, U_2, $coin$) in L_2 related to the query $H_2(m, ID, PK_{ID})$, C returns U_2 to $\mathcal{A}$. Otherwise, C randomly chooses $u \in Z_q^*$. If $ID \neq ID'$, C sets $U_2 = u \cdot P$ and $coin$=0. If $ID=ID'$, C first tosses a $coin \in \{0, 1\}$ with $\Pr[coin=1]=\delta$ for some δ that will be determined later. Moreover, C sets $U_2 = u \cdot P$ if $coin$=0, and $U_2 = u \cdot bP$ if $coin$=1. Then the challenger C returns U_2 to the adversary $\mathcal{A}$ and adds (m, ID, PK_{ID}, u, U_2, $coin$) to the list L_2.
- *Certificate generation* (ID, P_{ID}): C maintains an initially empty list L_C of tuples (ID, P_{ID}, R_{ID}, C_{ID}). If there exists (ID, P_{ID}, R_{ID}, C_{ID}) in the list L_C, C returns the certificate C_{ID} and public key PK_{ID}= (P_{ID}, R_{ID}) to $\mathcal{A}$. Otherwise, the challenger C performs the following procedures. If $ID=ID'$, C returns failure and terminates. If $ID \neq ID'$, the challenger C chooses two random elements v, $h \in Z_q^*$, and sets $C_{ID}=v$, $R_{ID} = v \cdot P - h \cdot P_{pub}$, and $h = f(ID, P_{ID}, R_{ID})$. If this particular tuple (ID, P_{ID}, R_{ID}, h) already exists in L_f, C discards it, chooses two new random elements v, $h \in Z_q^*$, and repeats the process until the outcome (ID, P_{ID}, R_{ID}, h) is new to L_f. Finally, C adds (ID, P_{ID}, R_{ID}, C_{ID}) to the list L_C and returns C_{ID} to the adversary $\mathcal{A}$.
- *Corruption* (ID): If ID's related tuple (ID, P_{ID}, S_{ID}) already exists in P^{list}, C returns S_{ID} as answer. Otherwise, C issues *User key generation* query on ID and returns the outcome S_{ID} to $\mathcal{A}$.
- *Public key replace* (ID, PK'_{ID}): The adversary A_I chooses a new public key $PK'_{ID}=(P'_{ID}, R'_{ID})$ of a user with identity ID and sends it to the challenger C. Then C replaces ID's corresponding tuples (ID, P_{ID}, S_{ID}) in P^{list} and (ID, P_{ID}, R_{ID}, C_{ID}) in L_C, respectively, with the new tuples (ID, P'_{ID}, $\perp$) and (ID, P'_{ID}, R'_{ID}, $\perp$).
- *Sign*(m, ID): When receiving a signature query on (m, ID) under the public key $PK_{ID}=(P_{ID}, R_{ID})$, the challenger C first issues the H_2 query with (m, ID, PK_{ID}) and obtains the corresponding tuple (m, ID, PK_{ID}, u, U_2, $coin$) L_2. The challenger C discusses the following two cases.
  1. If $ID=ID'$ and $coin$=1, C returns failure and terminates.
  2. Otherwise, C chooses two random elements h, $w \in Z_q^*$ and computes the signature $\sigma = w \cdot P_{ID} + u \cdot R_{ID} + u \cdot h \cdot P_{pub}$. Then, C adds (ID, P_{ID}, R_{ID}, h) in L_f and (m, ID, PK_{ID}, w, U_1) in L_1, and returns the signature σ. Even though the challenger C does not hold the corresponding secret key and the certificate of the user with ID, the signature σ is still valid because it passes the verification as follows.

$$\begin{aligned}\hat{e}(P, \sigma) &= \hat{e}(P, w \cdot P_{ID} + u \cdot R_{ID} + u \cdot h \cdot P_{pub}) \\ &= \hat{e}(P, w \cdot P_{ID}) \cdot \hat{e}(P, u \cdot R_{ID} + u \cdot h \cdot P_{pub}) \\ &= \hat{e}(w \cdot P, P_{ID}) \cdot \hat{e}(u \cdot P, R_{ID} + h \cdot P_{pub}) \\ &= \hat{e}(U_1, P_{ID}) \cdot \hat{e}(U_2, R_{ID} + h \cdot P_{pub}) \\ &= \hat{e}(P_{ID}, U_1) \cdot \hat{e}(R_{ID} + h \cdot P_{pub}, U_2).\end{aligned}$$

- *Forgery*: Assume that the adversary $\mathcal{A}$ forges a valid signature tuple (m^*, σ^*, ID^*) under the public key PK_{ID^*}. Here PK_{ID^*} is the original public key without replacement. First, the challenge C obtains the corresponding tuple (m^*, ID^*, PK_{ID^*}, u, U_2, $coin$) in the list L_2. We discuss three cases.
  1. If $ID^* \neq ID'$, the challenger C returns failure and terminates.
  2. If $ID^* = ID'$ and $coin$=0, the challenger C returns failure and terminates.
  3. If $ID^* = ID'$ and $coin$=1, the challenger C uses the forgery signature σ^* to solve the CDH problem as follows. Since σ^* is valid,

$$\hat{e}(P, \sigma^*) = \hat{e}(P_{ID^*}, U_1) \cdot \hat{e}(R_{ID^*} + h \cdot P_{pub}, U_2).$$

By properties of bilinear pairings,

$$\begin{aligned}\hat{e}(P, \sigma^*) &= \hat{e}(P_{ID^*}, U_1) \cdot \hat{e}(R_{ID^*} + h \cdot P_{pub}, U_2) \\ &= \hat{e}(P_{ID^*}, w \cdot P) \cdot \hat{e}(r_{ID^*} \cdot P + h \cdot aP, u \cdot bP) \\ &= \hat{e}(w \cdot P_{ID^*}, P) \cdot \hat{e}(r_{ID^*} \cdot P, u \cdot bP) \cdot \hat{e}(h \cdot aP, u \cdot bP) \\ &= \hat{e}(w \cdot P_{ID^*}, P) \cdot \hat{e}(r_{ID^*} \cdot u \cdot bP, P) \cdot \hat{e}(hu \cdot abP, P) \\ &= \hat{e}(w \cdot P_{ID^*} + r_{ID^*} \cdot U_2 + hu \cdot abP, P) \\ &= \hat{e}(P, w \cdot P_{ID^*} + r_{ID^*} \cdot U_2 + hu \cdot abP).\end{aligned}$$

So, we have $\sigma^* = w \cdot P_{ID^*} + r_{ID^*} \cdot U_2 + hu \cdot abP$. Then, the challenger C outputs $(\sigma^* - w \cdot P_{ID^*} - r_{ID^*} U_2)/hu$ as the solution abP of the CDH problem. Finally, we calculate the probability that C does not abort during the simulation. The probability that it does not abort during the *Certificate generation* query and the *Forgery* phases are $(1-1/q_c)$ and δ/q_c, respectively. In

addition, in the *Sign* query phase, the probability that it does not abort is $(1-\delta/q)^{q_s} \geq (1-\delta)^{q_s}$. Therefore, the probability that $\mathcal{C}$ does not abort is $(1-1/q)(1-\delta)^{q_s}\delta/q$. This value can be maximized at $\delta_{opt}=1/(q_s+1)$. Hence, if the adversary $\mathcal{A}$ has a non-negligible advantage ε to break the proposed SCBS scheme, the challenger $\mathcal{C}$ has a non-negligible advantage

$$\varepsilon' \geq \left(1-\frac{1}{q_c}\right)\left(1-\frac{1}{1+q_s}\right)^{q_s}\left(\frac{1}{(q_c\,(1+q_s))}\right)\varepsilon$$

to solve the CDH problem. Finally, to answer queries in the simulation game above, the required computation time is $\tau' \approx \tau + O(q_c+q_s+q_u+q_1+q_2)\tau_1$, where τ_1 is the time to perform a scalar multiplication in G.□

Theorem 2. *Suppose that the hash functions H_1 and H_2 are random oracles. Assume that a Type II UF-SCBS-ACM adversary $\mathcal{A}$ can break the proposed SCBS scheme with non-negligible advantage*

within running time τ. Then, there exists an algorithm $\mathcal{C}$ to solve the CDH problem with a non-negligible advantage

$$\varepsilon' \geq \left(1-\frac{1}{q_u}\right)^{q_{cor}}\left(1-\frac{1}{1+q_s}\right)^{q_s}\left(\frac{1}{q_u(1+q_s)}\right)\varepsilon$$

within running time $\tau' \approx \tau + O(q_s+q_u+q_1+q_2)\tau_1$, where τ_1 is the time to perform a scalar multiplication in G. Also, q_{cor}, q_s, q_u, q_1, and q_2 denote, respectively, the maximal numbers of $\mathcal{A}$'s queries to the *Corruption* oracle, the *Sign* oracle, the *User key generation* oracle, the random oracle H_1, and the random oracle H_2.

Proof. Assume that an algorithm $\mathcal{C}$ is given a random instance (P, aP, bP) of the CDH problem, where P, aP, $bP \in G$ with unknown a, $b \in Z_q^*$. Let $J=abP$ be the solution of the CDH problem. The algorithm $\mathcal{C}$ finds J by interacting with the adversary $\mathcal{A}$ as follows.

- *Setup*. The challenger $\mathcal{C}$ creates a list of public parameters $PP=\{G, G_T, q, \hat{e}, P, P_{pub}, f, H_1, H_2\}$ and $P_{pub}=s\cdot P$, where s is the master secret key. The challenger $\mathcal{C}$ gives $\mathcal{A}$ the public parameters PP and s. Here, the hash functions f, H_1 and H_2 are random oracles controlled by the challenger $\mathcal{C}$. To be consistent and collision-free, the challenger $\mathcal{C}$ maintains three initially empty lists L_f, L_1 and L_2 to record the responses of the hash functions. Since the adversary $\mathcal{A}$ knows the master secret key s, it can compute the certificate C_{ID} and know all users' public keys so that it does not need to issue the *certificate generation* query.
- *Queries*: The challenger $\mathcal{C}$ first randomly chooses $t \in [1, q_u]$ and lets ID' denote the target identity of the t-th query to the *User key generation* oracle. The challenger $\mathcal{C}$ answers the queries issued by $\mathcal{A}$ as follows.
  - *User key generation* (ID): $\mathcal{C}$ maintains an initially empty list P^{list} of tuples (ID, P_{ID}, S_{ID}). If ID's corresponding tuple (ID, P_{ID}, S_{ID}) already exists in P^{list}, $\mathcal{C}$ returns P_{ID} as answer. Otherwise, we split into two cases. If $ID \neq ID'$, $\mathcal{C}$ randomly chooses a value $S_{ID} \in Z_q^*$ and computes $P_{ID}=S_{ID}\cdot P$. If $ID=ID'$, $\mathcal{C}$ sets $P_{ID}=aP$ and $S_{ID}=\perp$. In both cases, $\mathcal{C}$ adds (ID, P_{ID}, S_{ID}) to P^{list}, and returns P_{ID} to the adversary $\mathcal{A}$.
  - $f(ID, P_{ID}, R_{ID})$: As the $f(ID, P_{ID}, R_{ID})$ query in Theorem 1.
  - $H_1(m, ID, PK_{ID})$: $\mathcal{C}$ maintains an initially empty list L_1 of tuples (m, ID, PK_{ID}, w, U_1, *coin*). If there exists a tuple (m, ID, PK_{ID}, w, U_1, *coin*) in L_1 related to the query $H_1(m, ID, PK_{ID})$, $\mathcal{C}$ returns U_1 to the adversary $\mathcal{A}$. Otherwise, $\mathcal{C}$ first randomly chooses $w \in Z_q^*$. Then, if $ID \neq ID'$, $\mathcal{C}$ sets $U_1=w\cdot P$ and *coin*=0. If $ID=ID'$, $\mathcal{C}$ tosses a *coin* $\in \{0, 1\}$ with $\Pr[coin=1]=\delta$ for some δ that will be determined later. Moreover, $\mathcal{C}$ sets $U_1=w\cdot P$ if *coin*=0, and $U_1=w\cdot bP$ if *coin*=1. Finally, $\mathcal{C}$ returns the corresponding U_1 to $\mathcal{A}$ and adds (m, ID, PK_{ID}, w, U_1, *coin*) to L_1.
  - $H_2(m, ID, P_{ID}, R_{ID})$: $\mathcal{C}$ maintains an initially empty list L_2 of tuples (m, ID, PK_{ID}, u, U_2). If there exists a tuple (m, ID, PK_{ID}, u, U_2) in L_2 related to the query $H_2(m, ID, P_{ID}, R_{ID})$, $\mathcal{C}$ returns U_2 to $\mathcal{A}$. Otherwise, $\mathcal{C}$ randomly chooses $u \in Z_q^*$, sets $U_2=u\cdot P$, returns U_2 to $\mathcal{A}$, and adds (m, ID, PK_{ID}, u, U_2) to L_2.
  - *Corruption* (ID): If ID's corresponding tuple (ID, P_{ID}, S_{ID}) already exists in the list P^{list}, $\mathcal{C}$ returns
  - S_{ID} as answer. Otherwise, $\mathcal{C}$ issues *User key generation* query on ID to obtain the associated S_{ID}. Finally, $\mathcal{C}$ returns S_{ID} to the adversary $\mathcal{A}$ if $ID \neq ID'$, and returns failure and terminates, otherwise.
  - *Public key replace* (ID, PK'_{ID}): The adversary $\mathcal{A}$ chooses a new public key $PK'_{ID}=(P'_{ID}, R'_{ID})$ of a user with identity ID and sends it to the challenger $\mathcal{C}$. Then $\mathcal{C}$ replaces the corresponding tuple (ID, P_{ID}, S_{ID}) in P^{list} with the new tuple (ID, P'_{ID}, $\perp$).
  - *Sign* (m, ID): When receiving a signature query on (m, ID) under the public key $PK_{ID}=(P_{ID}, R_{ID})$, the challenger $\mathcal{C}$ first issues the H_2 query with (m, ID, PK_{ID}) and obtains the

corresponding tuple (m, ID, PK_{ID}, w, U_1, $coin$) in L_1. The challenger C discusses two cases.

1. If $ID=ID'$and $coin$=1, C returns failure and terminates.
2. Otherwise, C chooses two random elements h, $u \in Z_q^*$ and computes the signature $\sigma = w \cdot P_{ID}+u \cdot R_{ID}+u \cdot h \cdot P_{pub}$. Then, C adds (ID, P_{ID}, R_{ID}, h) in L_f and (m, ID, PK_{ID}, u, U_2) in L_2, and returns the signature σ. The signature σ is valid because it can pass the verification, as described in the proof of Theorem 1.

- *Forgery*: Assume that the adversary $\mathcal{A}$ forges a valid signature tuple (M^*,σ^*, ID^*) under the public key PK_{ID*}=(P_{ID*}, R_{ID*}). First, the challenger C obtains the corresponding tuple (m^*, ID^*, PK_{ID*}, w, U_1, $coin$) in the list L_1. We discuss three cases.

  1. If $ID^* \neq ID'$, the challenger C returns failure and terminates.
  2. If $ID^*=ID'$ and $coin$=0, the challenger C returns failure and terminates.
  3. If $ID^*=ID'$ and $coin$=1, the challenger C uses the forgery signature σ^* to solve the CDH problem as follows. Since σ^* is valid,

  $\hat{e}(P, \sigma^*)=\hat{e}(P_{ID*}, U_1)\cdot\hat{e}(R_{ID*} + h\cdot P_{pub}, U_2)$.

  By properties of bilinear pairings,

  $\hat{e}(P, \sigma^*)=\hat{e}(P_{ID*}, U_1)\cdot\hat{e}(R_{ID*}+h\cdot P_{pub}, U_2)$

  $= \hat{e}(aP, w\cdot bP)\cdot\hat{e}(R_{ID*}+h\cdot s\cdot P, u\cdot P)$

  $= \hat{e}(w\cdot abP, P)\cdot\hat{e}(u\cdot R_{ID*}+u\cdot h\cdot s\cdot P, P)$

  $= \hat{e}(w\cdot abP+u\cdot R_{ID*}+u\cdot h\cdot s\cdot P, P)$

  $= \hat{e}(P, w\cdot abP+u\cdot R_{ID*}+u\cdot h\cdot s\cdot P)$.

  So, we have $\sigma^*=w\cdot abP+u\cdot R_{ID*}+u\cdot h\cdot s\cdot P$. Then, the challenger C outputs $(\sigma^* - u\cdot R_{ID*}-u\cdot h\cdot s\cdot P)/w$ as the solution abP of the CDH problem. Finally, we calculate the probability that C does not abort during the simulation. The probability that it does not abort during the *corruption* query phase is $(1-1/q_u)^{q}_{cor}$ and the *Forgery* phase is δ/q_u. In addition, in the *Sign* query phase, the probability that it does not abort is $(1-\delta / q)^{q_s} \geq (1-\delta)^{q_s}$. Therefore, the probability that C does not abort is $(1-1/q_u)^{q_{cor}}(1-\delta)^{q_s}\delta/q_u$. This value can be maximized at δ_{opt} =1/(q_s +1). Hence, if the adversary $\mathcal{A}$ has a non-negligible advantage ε to break the proposed SCBS scheme, the challenger C has a non-negligible advantage

$$\varepsilon' \geq \left(1-\frac{1}{q_u}\right)^{q_{cor}}\left(1-\frac{1}{1+q_s}\right)^{q_s}\left(\frac{1}{q_u(1+q_s)}\right)\varepsilon$$

to solve the CDH problem. Finally, for answering queries in the simulation game above, the required computation time is $\tau'=\tau+O(q_s+q_u+q_1+q_2)\tau_1$, where τ_1 is the time to perform a scalar multiplication in G. □

7. Performance comparisons and discussions

In the following, we first define several time-consuming operations.

- T_p: the time of executing a bilinear pairing operation $\hat{e}$: $G \times G \rightarrow G_T$;
- T_m: the time of executing a scalar multiplication in G, an exponentiation in G_T or an exponentiation operation in Z_q^*;
- T_h: the time of executing a map-to-point hash function.

In the following, Table 2 lists the comparisons among Zhang's CBS scheme [28], Liu *et al.*'s SCBS scheme [30], Li *et al.*'s SCBS scheme [24] and our SCBS scheme in terms of signature size,

the computational costs of signing and verification, and security. For the signature size, Zhang's scheme requires two elements in G while Liu *et al.*'s, Li *et al.*'s and our schemes require only one element in G.

For the computational cost of the signing and verification phases, our scheme is better than Zhang's scheme, but requires more operations than Liu *et al.*'s and Li *et al.*'s schemes. However, both Liu *et al.*'s and Li *et al.*'s schemes suffer from the attacks of Type I adversary under an accredited security model. According to Table 2, our scheme is provably secure while retaining efficiency.

Table 2. Comparisons between the previously proposed CBS and SCBS schemes and ours

	Zhang's CBS scheme [28]	**Liu et al.'s SCBS scheme [30]**	**Li et al.'s SCBS scheme [24]**	**Our SCBS scheme**
Signature size	2	1	1	1
Computation cost of signing	5Tm	Tm+Th	Tm+Th	2Tm +2Th
Computation cost of verification	3Tp +3Tm+Th	2Tp +Tm+2Th	4Tp +Tm+2Th	3Tp +Tm+2Th
Type I attack	UF-CBS-ACM	Existing attack [29]	Existing attack (Section 4)	UF-SCBS-ACM
Type II attack	UF-CBS-ACM	UF-SCBS-ACM	UF-SCBS-ACM	UF-SCBS-ACM

8. Conclusions

In this article, we demonstrate that Li *et al*.'s SCBS scheme is insecure against the attacks of Type I adversary under an accredited security model. We also propose the *first* provably secure SCBS scheme in the random oracle model. In the accredited security model, the adversary is allowed to issue the *Sign* query even though the challenger does not hold the corresponding secret key and the certificate of the user. This is the strongest capability that an adversary can possess in SCBS schemes. Moreover, due to the short signature length, our SCBS scheme is well suited for low-bandwidth communication environments with high-level security.

Acknowledgements

The authors would like to appreciate anonymous referees for their valuable comments and constructive suggestions. This research was partially supported by Ministry of Science and Technology, Taiwan, R.O.C., under grant no. MOST103 -2221-E-018-022-MY2.

References

[1] **T. ElGamal.** A public key cryptosystem and a signature scheme based on discrete logarithms. In: *Proc. of Crypto'84*, LNCS, 196, 1984, pp. 10–18.

[2] **R. L. Rivest, A. Shamir, L. Adleman.** A method for obtaining digital signatures and public-key cryptosystems. *Commun. of ACM*, 1978, Vol. 21, No. 2, 120–126.

[3] **A. Shamir.** Identity-based cryptosystems and signature schemes. In: *Proc. of Crypto'84*, LNCS, 196, 1984, pp. 47–53.

[4] **D. Boneh, M. Franklin.** Identity-based encryption from the Weil pairing. In: *Proc. of. Crypto'01*, LNCS, 2139, 2001, pp. 213–229.

[5] **S. S. Al-Riyami, K. G. Paterson.** Certificateless public key cryptography. In: *Proc. of ASIACRYPT'03*, LNCS, 2894, 2003, pp. 452–473.

[6] **Y. M. Tseng, T. T. Tsai.** Efficient revocable ID-based encryption with a public channel.

[7] *Computer Journal*, 2012, Vol. 55, No. 4, 475-486.

[8] **T. T. Tsai, Y. M. Tseng.** Revocable certificateless public key encryption. *IEEE Systems Journal*, 2015, Vol. 9, No. 3, 824-833.

[9] **Y. H. Hung, T. T. Tsai, Y. M. Tseng, S. S. Huang.** Strongly secure revocable ID-based signature without random oracles. *Information Technology and Control*, 2014, Vol. 43, No. 3, 264–276.

[10] **T. T. Tsai, S. S. Huang, Y.M. Tseng.** Secure certificateless signature with revocation in the standard model. *Mathematical Problems in Engineering*, 2014, Vol. 2014, Article ID 728591.

[11] **C. Gentry.** Certificate-based encryption and the certificate revocation problem. In: *Proc. of Eurorypt'03*, LNCS, 2656, 2003, pp. 272–293.

[12] **A. Fiat, A. Shamir.** How to prove yourself: practical solutions to identification and signature Problems. In: *Proc. of Crypto'86, LNCS*, 263, 1987, pp. 186–194.

[13] **K. Kurosawa, S. Heng.** From digital signature to ID-based identification/signature. In: *Proc. of PKC'04, LNCS*, 2947, 2004, pp. 248–261.

[14] [**Y. M. Tseng, T.Y. Wu, J.D. Wu.** A pairing-based user authentication scheme for wireless clients with smart cards. *Informatica*, 2008, Vol. 19, No. 2, 285–302.

[15] **D. He, N. Kumar, N. Chilamkurti.** A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks. *Information Sciences*, 2015, Vol. 321, 263-277.

[16] **D. He, S. Zeadally.** Authentication protocol for an ambient assisted living system. *IEEE Communications Magazine*, 2015, Vol. 53, No. 1, 71-77.

[17] **K. Barr, K. Asanovic.** Energy aware lossless data compression. *ACM Transactions on Computer Systems*, 2006, Vol. 24, No. 3, 250–291.

[18] **D. Boneh, B. Lynn, H. Shacham**. Short signatures from the Weil pairing. In: *Proc. of ASIACRYPT'01*, LNCS, 2248, 2001, pp. 514–532.

[19] **D. Boneh, X. Boyen.** Short signatures without random oracles. In: *Proc. of EUROCRYPT'04*, LNCS, 3027, 2004, pp. 56–73.

[20] **F. Zhang, X. Chen, W. Susilo, Y. Mu.** A new short signature scheme without random oracles from bilinear pairings. *IACR Cryptology ePrint Archive*, 2005, Article 386.

[21] **K. Y. Choi, J.H. Park, D.H. Lee**. A new provably secure certificateless short signature scheme. Computers and Mathematics with Applications, 2011, Vol. 61, No. 7, 1760–1768.

[22] **R. Tso, X. Huang, W. Susilo.** Strongly secure certificateless short signatures. *Journal of Systems and Software*, 2012, Vol. 85, No. 6, 1409–1417.

[23] **Y. C. Chen, R. Tso, W. Susilo, X. Huang, G. Horng.** Certificateless signatures: structural extensions of security models and new provably secure schemes. *IACR Cryptology ePrint Archive*, 2013, Article 193.

[24] **D. He, B. Huang, J. Chen.** New certificateless short signature scheme. *IET Information Security*, 2013, Vol. 7, No. 2, 113-117.

[25] **J. Li, X. Huang, Y. Zhang, L. Xu.** An efficient short certificate-based signature scheme. *Journal of Systems and Software*, 2012, Vol. 85, No. 2, 314–322.

[26] **B. G. Kang, J. H. Park, S. G. Hahn.** A Certificate-based signature scheme. In: *Proc. of CT-RSA'04*, LNCS, 2964, 2004, pp. 99–111.

[27] **J. Li, X. Huang, Y. Mu, W. Susilo, Q. Wu.** Certificate-based signature: security model and efficient construction. In: *Proc. of EuroPKI' 07*, LNCS, 4582, 2007, pp. 110-125.

[28] **J. K. Liu, J. Baek, W. Susilo, J. Zhou.** Certificate-based signature scheme without pairings or random oracles. In: *Proc. of ISC'08*, LNCS, 5222, 2008, pp. 285–297.

[29] **J. Zhang.** On the security of a certificate-based signature scheme and its improvement with pairings. In: *Proc. of ISPEC'09*, LNCS, 5451, 2009, pp. 47–58.

[30] **W. Wu, Y. Mu, W. Susilo, X. Huang.** Certificate-based signatures revisited. *Journal of Universal Computer Science*, 2009, Vol. 15, No. 8, 1659–1684.

[31] **J. K. Liu, F. Bao, J. Zhou.** Short and efficient certificate-based signature. In: *Proc. of NETWORKING 2011 Workshops*, LNCS, 6827, 2011, pp. 167–178.

[32] **L. Cheng, Y. Xiao, G. Wang.** Cryptanalysis of a certificate-based on signature scheme. *Procedia Engineering*, 2012, Vol. 29, 2821–2825.

[33] **M. Bellare, P. Rogaway.** Random oracles are practical: a paradigm for designing efficient protocols. In: *Proc. of 1st ACM conference on Computer and Communications Security*, 1993, pp. 62–73.

[34] **R. Canetti, O. Goldreich, S. Halevi.** The random oracle methodology, revisited. *J. ACM*, 2004, Vol. 51, No. 4, 557–594.

Received July 2015.