Journal of Universal Computer Science, vol. 19, no. 10 (2013), 1351-1374
submitted: 6/3/12, accepted: 29/3/13, appeared: 28/5/13

Key-Insulated Signcryption

Jia Fan [1]
(Science and Technology on Communication Security Laboratory,
Chengdu, China
fanjia0628@gmail.com)

Yuliang Zheng
(University of North Carolina at Charlotte, USA
yzheng@uncc.edu)

Xiaohu Tang
(Southwest Jiaotong University, Chengdu, China
xhutang@swjtu.edu.cn)

Abstract: Signcryption is a public key cryptographic technique that is particularly suited for mobile communications thanks to its light computational and communication overhead. The wide spread use of signcryption in a mobile computing environment, however, is accompanied by an increased risk of private key exposure. This paper addresses the issue of key exposure by proposing a key-insulated signcryption technique. We define a security model for key-insulated signcryption and prove that the key-insulated signcryption technique is provably secure in the security model.
Key Words: signcryption, key exposure, key insulation
Category: H.2, H.3.7, H.5.4

1 Introduction

Public key encryption and digital signature are two basic primitives in modern cryptography. They provide us with confidentiality and authenticity independently. If we require both functions, usually we have to carefully sign and encrypt data sequentially. The traditional "Sign-then-Encrypt" or "Encrypt-then-Sign" method consumes a sum of costs by both primitives. In 1997, Zheng proposed a public key primitive, called signcryption [Zheng, 1997], which offers both functions with a cost far less than the sum of costs for performing the two primitives separately. In addition to advantages in efficiency, signcryption typically offers a higher level of security than a traditional method. According to [An et al., 2002], neither "Sign-then-Encrypt" nor "Encrypt-then-Sign" scheme could satisfy the requirement of indistinguishability against adaptively chosen ciphertext attack (IND-CCA) and that of existential unforgeability against chosen message attack (EUF-CMA) simultaneously, while most of the signcryption schemes do. The

[1] The first author is supported by Innovation Fund 2012 of China Electronic Technology Group Corporation.

International Organization for Standardization has recently established the first global standard on signcryption techniques to ensure compatibility among applications that employ signcryption to provide data confidentiality and integrity in an efficient manner [ISO, 2011].

Signcryption has been applied in mobile communications, such as mobile ad hoc networks [Park and Lee, 2005], secure e-mails [Gamage et al., 1999], mobile grid web services [Park et al., 2005], electronic payment [Seo and Kim, 1999] and so on. Mobile devices, such as cell phones and smart cards which are inexpensive and lightweight, bring us convenience while at the same time increase the risk of private key exposure. Like most public key cryptographic techniques, it is typical for signcryption to implicitly assume that a user's private key is kept secure. It is easy to observe that most of the provably secure signcryption schemes (no matter how strong they are) under traditional definitions will be broken in an event where the user's private key is exposed [Zheng, 1997, An et al., 2002, Boyen, 2003, Chen and Malone-Lee, 2005]. Therefore, most of the adversaries in practice may choose to break a provably secure system by stealing the mobile device to extract the private key, rather than to break it directly without any knowledge of the private key.

Along with the increasing use of mobile devices, key exposure problem has become a great threat in various application environments of signcryption. Two methods have been proposed by researchers to address this problem, these being a threshold technique [Li et al., 2006, Li et al., 2008] and a forward secure technique [Libert and Quisquater, 2003]. With the threshold signcryption, a sender or receiver's private key is split into several sub-private keys. To do signcryption or unsigncryption, a required number of sub-private keys should take part in computation. This technique ensures security as long as the number of exposed sub-private keys is not larger than the required threshold. With the forward secure signcryption, security is guaranteed even if an attacker manages to get hold of the private key of a sender but not that of a receiver.

To address the key exposure issue, we introduce a new method called key-insulated signcryption (we will call it KISC in short). With a KISC system, the life span of the system is divided into discrete time periods, while a user's private key is split into two parts stored in two separate devices both held by the same user: a temporary private key, stored in a mobile device, which is convenient to use but is relatively insecure, and a master private key, stored in a home device, which is physically secure but less likely to move. The user's public key remains unchanged throughout the life span of the system, while the temporary private key is refreshed at every period via interactions between the two devices. Cryptographic operations in a given period only involves the corresponding temporary private key without further access to the home device. Informally, the security of a KISC system contains the following two aspects. First, when the mobile device is unsafe (e.g. the temporary private key at some period is exposed), the security of the whole system in the remaining periods can be guaranteed. Second, even if the home device is compromised (e.g. the master private key is exposed), the security of the whole system is still ensured as long as none of the temporary private keys is exposed. Particularly, for confidentiality, even if a sender's private key is totally exposed, an attacker of KISC still cannot correctly decrypt the ciphertext. As a result KISC achieves forward security. Moreover, KISC makes use of the threshold and key updating technique by separately storing the private key and updating the temporary private key periodically. Therefore, KISC offers a security guarantee against key exposure from two important factors in practice, say time and space.

Key-insulation was first presented in [Dodis et al., 2002]. Till now, it has been applied in public key encryption [Dodis et al., 2002] as well as digital signature [Dodis and Katz et al., 2003]. We are the first to consider key-insulated signcryption. In a different direction, the idea of key-insulation has been extended to intrusion-resilient cryptosystems by Itkis and Reyzin [Itkis and Reyzin, 2002]. In such a system, the master private key is also updated periodically, and even after both of the devices are exposed arbitrarily in many periods, the system is still secure, as long as the exposures are not simultaneous. Although intrusion-resilient schemes have attractive properties, their efficiency is much lower than key-insulated schemes [Dodis et al., 2003, Itkis and Reyzin, 2002]. For this reason, we do not further consider intrusion-resilient signcryption.

2 Proposed Model of KISC

The security requirements of a KISC system include confidentiality, unforgeability, and secure key update. We define confidentiality and unforgeability by considering either the mobile device or the home device is unsafe respectively. First, we describe the syntax of KISC as follows.

2.1 Syntax of of KISC

A KISC scheme contains the following six algorithms, among which $SetupPub$, $KeyGen$, $DevUpd$, $Signcrypt$ are probabilistic, and $MobUpd$, $Unsigncrypt$ are deterministic.

- $SetupPub(1^k, N)$: On input a security parameter 1^k and a total number of time periods N, it returns a system public parameter *pub*.

  A trusted authority runs this algorithm and publishes pub to all users.

- $KeyGen(pub)$: On input *pub*, it returns a public key pk_P, a master private key sk_P^*, and an initial temporary private key sk_{P_0} for ID_P.

A user ID_P runs this algorithm, then publishes pk_P, stores (pub, sk_P^*) in his home device and (pub, sk_{P_0}) in his mobile device.

- $DevUpd(pub, sk_P^*, i, j)$: On input pub, sk_P^*, two time period indexes i, j, it returns a helper key $sk_{P_{i,j}}$.

  In order to help update the temporary private key of user ID_P from period i to period j, the home device of ID_P runs this algorithm, and then sends $sk_{P_{i,j}}$ to the mobile device in a secure manner (e.g. connected by a USB cable).

- $MobUpd(pub, sk_{P_i}, i, j, sk_{P_{i,j}})$: On input pub, $sk_{P_{i,j}}$, i, j, and a temporary private key sk_{P_i} for period i, this algorithm returns a temporary private key sk_{P_j} for period j.

  The mobile device of user ID_P runs this algorithm to update the temporary private key from sk_{P_i} to sk_{P_j}. Finally, it erases $sk_{P_{i,j}}$ and stores (pub, sk_{P_j}, j) to replace (pub, sk_{P_i}, i).

- $Signcrypt(pub, sk_{S_i}, pk_R, i, j, M)$: On input pub, sk_{S_i}, pk_R, sender's time period index i, receiver's time period index j, and a message $M \in \mathcal{M}$ ($\mathcal{M}$ is the message space), it returns a signcryptext σ on a message M.

  The mobile device of a sender at period i runs this algorithm to generate a signcryptext σ for a message M, and sends (σ, i, j) to a receiver expecting this receiver to unsigncrypt it at period j.

- $Unsigncrypt(pub, pk_S, sk_{R_j}, i, j, \sigma)$: On input pub, sk_{R_j}, a sender's time period index i, a receiver's time period index j, pk_S and σ, it returns a message M or a special symbol $\perp$ if the signcryptext is invalid.

  The mobile device of receiver at period j runs this algorithm to unsigncrypt σ generated by a sender at period i.

For consistency, we require that for all $\sigma \leftarrow Signcrypt(pub, sk_{S_i}, pk_R, i, j, M)$, we have $M = Unsigncrypt(pub, pk_S, sk_{R_j}, i, j, \sigma)$.

The process of temporary private key update is described in Table 1. In order to update the user's initial temporary private key, we set $i \in [0, ..., N]$ in $DevUpd$ and $MobUpd$. In other algorithms we set $i \in [1, ..., N]$. Note that $j \in [1, ..., N]$ in all algorithms.

Note that i and j in the above algorithms are not mutually constrained. This results in two interesting properties: 1. Any user can update his temporary private key to future time periods as well as to past time periods. 2. A sender can signcrypt a message to a receiver who is in a different time period.

Home Device (pub, sk_P^*)		Mobile Device (pub, sk_{P_i}, i)
		Choose $j \in [1, ..., N]$
	$\xleftarrow{i,j}$	
$sk_{P_{i,j}} \leftarrow DevUpd(pub, sk_P^*, i, j)$		
	$\xrightarrow{sk_{P_{i,j}}}$	
		$sk_{P_j} \leftarrow MobUpd(pub, sk_{P_i}, i, j, sk_{P_{i,j}})$
		erases $sk_{P_{i,j}}$
		$sk_{P_i} \leftarrow sk_{P_j}$, $i \leftarrow j$

Table 1: KISC: temporary private key update from period i to j

2.2 Security Definitions of KISC

In this section, we will formalize security definitions for KISC in three aspects. To give a clear view, we list the names of all the security definitions for KISC as follows:

- Confidentiality
  - IND-CCA security when the mobile device is unsafe.
  - IND-CCA security when the home device is unsafe.
- Unforgeability
  - sEUF-CMA security when the mobile device is unsafe.
  - sEUF-CMA security when the home device is unsafe.
- Secure Key Update:
  - This definition ensures the security of the update step on the mobile device, particularly when the partial private key $sk_{P_{i,j}}$ is in use or it is not erased yet.

The security of confidentiality and unforgeability are defined below through a series of attack games. Each game is played between an adversary $\mathcal{A}$ and its environment Σ which contains a challenger $\mathcal{C}$ and several types of oracles, such as Extract Oracle $\mathcal{O}_{ex}$, Unsigncryption Oracle $\mathcal{O}_{usc}$ and Signcryption Oracle $\mathcal{O}_{sc}$. These oracles will answer queries from respective adversaries.

2.2.1 Definition of Confidentiality

For confidentiality, we offer two definitions. One considers the case that the home device is safe while the mobile device is unsafe, say the temporary private keys during several time periods are exposed. The other one considers the case that the mobile device is safe while the home device is unsafe, say the master private key is exposed.

Mobile Device is Unsafe. We define an attack game, called indistinguishability in KISC under chosen ciphertext attack in case the mobile device is unsafe (IND-CCA-M). Specifically, it contains five stages as follows:

- *Stage 1:* $\mathcal{C}$ computes $pub \leftarrow SetupPub(1^k, N)$, $(sk_A^*, sk_{A_0}, pk_A) \leftarrow KeyGen(pub)$. Then it gives (pub, pk_A) to $\mathcal{A}$, equips $\mathcal{O}_{ex}$, $\mathcal{O}_{usc}$ and $\mathcal{O}_{sc}$ with $(pub, sk_A^*, sk_{A_0}, pk_A)$.
- *Stage 2:* $\mathcal{A}$ is able to ask for a number of queries, each is one of the following three types.
  - Extract Query: $\mathcal{A}$ submits a time period i to $\mathcal{O}_{ex}$, which then returns sk_{A_i} to $\mathcal{A}$.
  - Signcryption Query: $\mathcal{A}$ submits (M, pk_R, i, j) to $\mathcal{O}_{sc}$, which then returns to $\mathcal{A}$ with an outcome of $Signcrypt(pub, sk_{A_i}, pk_R, i, j, M)$.
  - Unsigncryption Query: $\mathcal{A}$ submits (σ, pk_S, i, j) to $\mathcal{O}_{usc}$, which then returns to $\mathcal{A}$ with an outcome of $Unsigncrypt(pub, pk_S, sk_{A_j}, i, j, \sigma)$.
- *Stage 3:* $\mathcal{A}$ submits $(M_0, M_1, pk_{S^*}, sk_{S^*_{i^*}}, i^*, j^*)$ to $\mathcal{C}$, where M_0 and M_1 are of equal length and $\mathcal{A}$ has not asked for an extract query on j^*. $\mathcal{C}$ chooses a random bit β, and computes $\sigma^* \leftarrow Signcrypt(pub, sk_{S^*_{i^*}}, pk_A, i^*, j^*, M_\beta)$. Finally, it returns σ^* to $\mathcal{A}$.
- *Stage 4:* It is almost the same as Stage 2, except that $\mathcal{A}$ is not allowed to ask for an unsigncryption query on $(\sigma^*, pk_{S^*}, i^*, j^*)$, or an extract query on j^*.
- *Stage 5:* $\mathcal{A}$ outputs a guess bit β'. $\mathcal{C}$ checks whether $\beta' = \beta$. If it is, then $\mathcal{A}$ wins the challenge.

The advantage for $\mathcal{A}$ to win the challenge in the IND-CCA-M game is defined as

$$\epsilon = |Pr[\beta' = \beta] - 1/2|.$$

In the above i, j, i^*, j^* are all chosen from $[1, ..., N]$, and M, M_0, M_1 are all in $\mathcal{M}$, except that in extract queries $i \in [0, ..., N]$. We denote the maximum number of extract, signcryption and unsigncryption queries made by the adversary as q_e, q_s and q_u respectively.

Definition 1 *A KISC scheme is* $(t, q_s, q_u, q_e, \epsilon)$ *IND-CCA-M secure, if for any adversary* $\mathcal{A}$ *running in time* t*, where* t*,* q_s q_u *and* q_e $(q_e \leq N)$ *are all polynomials in* k*, and the advantage* ϵ *is negligible in* k*. Particularly, if* $q_e = N$*, we say the KISC scheme is strong IND-CCA-M secure.*

Home Device is Unsafe. We define an attack game, called indistinguishability in KISC under chosen ciphertext attack in case the home device is unsafe (IND-CCA-H). This attack game is similar to the IND-CCA-M attack game, except that at Stage 1, $\mathcal{C}$ gives (pub, pk_A, sk_A^*) to $\mathcal{A}$, and there is no $\mathcal{O}_{ex}$ oracle in this game, since the adversary does not ask for any extract queries.

Definition 2 *A KISC scheme is* (t, q_s, q_u, ϵ) *IND-CCA-H secure, if for any adversary* $\mathcal{A}$ *running in time* t*, where* t*,* q_s q_u *are all polynomials in* k*, the advantage* ϵ *is negligible in* k*.*

2.2.2 Definition of Unforgeability

Similar as the definition of confidentiality, we also define two types of definitions for unforgeability as follows.

Mobile Device is Unsafe. We describe an attack game, called strong existential unforgeability under chosen message attack in case the mobile device is unsafe(sEUF-CMA-M). Specifically, it contains three stages, where Stage 1 and Stage 2 are described the same as the Stage 1 and Stage 2 in IND-CCA-M game, while Stage 3 is described as follows:

– *Stage 3:* $\mathcal{A}$ outputs $(\sigma^*, pk_{R^*}, sk_{R^*_{j^*}}, i^*, j^*)$ to $\mathcal{C}$, where σ^* is not one of the results returned by $\mathcal{O}_{sc}$ and $\mathcal{A}$ has never required an extract query on i^*. Then, $\mathcal{C}$ checks whether $Unsigncrypt(pub, pk_A, sk_{R^*_{j^*}}, i^*, j^*, \sigma^*) = \perp$. If it is not, then $\mathcal{A}$ wins the challenge.

The advantage for $\mathcal{A}$ to win the challenge in the sEUF-CMA-M game is defined as ϵ which is the probability of an event that $\mathcal{A}$ wins the challenge. Note that $i, j, i^*, j^*, M, M_0, M_1, q_e, q_s, q_u$ are all defined as the same as in the IND-CCA-M game.

Definition 3 *A KISC scheme is* $(t, q_s, q_u, q_e, \epsilon)$ *sEUF-CMA-M secure, if for any adversary* $\mathcal{A}$ *running in time* t*, where* t*,* q_s q_u *and* q_e *are all polynomials in* k*, the advantage* ϵ *is negligible in* k*. Particularly, if* $q_e = N$*, we say the KISC scheme is strong sEUF-CMA-M secure.*

Home Device is Unsafe. We describe an attack game, called strong existential unforgeability in KISC under chosen message attack in case the home device is unsafe (sEUF-CMA-H). The differences between sEUF-CMA-H and sEUF-CMA-M are similar as the differences between IND-CCA-H and IND-CCA-M.

The attack game of sEUF-CMA-H is similar as sEUF-CMA-M, except that at Stage 1, the $\mathcal{C}$ gives (pub, pk_A, sk_A^*) to $\mathcal{A}$, and there is no $\mathcal{O}_{ex}$ in this game, thus the adversary does not ask for any extract queries. Finally, the (t, q_s, q_u, ϵ) sEUF-CMA-H security is defined as follows, which is similar as we define the $(t, q_s, q_u, q_e, \epsilon)$ sEUF-CMA-M security.

Definition 4 *A KISC scheme is (t, q_s, q_u, ϵ) sEUF-CMA-H secure, if for any adversary $\mathcal{A}$ running in time t, where t, q_s q_u are all polynomials in k, the advantage ϵ is negligible in k.*

2.2.3 Definition of Secure Key Update

In the KISC system, when the mobile device gets the partial private key $sk_{P_{i,j}}$, it computes a new private key sk_{P_j} by running $MobUpd(pub, sk_{P_i}, i, j, sk_{P_{i,j}})$. According to the above security definitions (either for confidentiality or for unforgeability) in case that the mobile device is unsafe, the adversary is allowed to have access to sk_{P_i} and sk_{P_j} but not $sk_{P_{i,j}}$. While in practice, if the mobile device is not secure, then the adversary may probably get the value of $sk_{P_{i,j}}$ (*e.g.* the mobile device is stolen when $sk_{P_{i,j}}$ is not yet erased). In some cases, the exposure of $sk_{P_{i,j}}$ may bring security problems. For example, if $sk_{P_{i,j}} = sk_{A^*}$ and the adversary is able to get the partial private key $sk_{P_{i,j}}$, then the resulting scheme is neither sEUF-CMA-M nor IND-CCA-M secure as long as one of the temporary private key is exposed.

To address the above problem, another security requirement called secure key update is needed, that is the exposure of $sk_{P_{i,j}}$ does not leak more information than the exposure of both sk_{P_i} and sk_{P_j}. Therefore, this requirement ensures that the damage of exposing $sk_{P_{i,j}}$ is no more than the damage of exposing both sk_{P_i} and sk_{P_j}. A formal definition is as follows.

Definition 5 *A KISC scheme has secure key update if for any i $(0 \leq i \leq N)$, j $(1 \leq j \leq N)$ and any identity ID_P with public key pk_P, $sk_{P_{i,j}}$ can be efficiently computed by sk_{P_i}, sk_{P_j}, pk_P and pub.*

3 Proposed KISC Scheme

We describe the proposed KISC scheme in Table 2 and Table 3.

$SetupPub(1^k, N)$: Run by a trusted authority. 1. Choose groups $\{\mathbb{G}, \mathbb{G}_\mathbb{T}\}$: $\mathbb{G}$ is generated by g, both groups are of prime order p, and a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ exists; 2. Choose random elements $\{g_1, g_2, g_3, u_0, u_1, ..., u_{n_1}, v_0, v_1, ..., v_{n_2}, w_0, w_1, ..., w_{n_3}\}$ all from $\mathbb{G}$, let $U \leftarrow (u_1, ..., u_{n_1})$, $V \leftarrow (v_1, ..., v_{n_2})$, $W \leftarrow (w_1, ..., w_{n_3})$; 3. Choose four collision resistant hash functions $\{H_1, H_2, H_3, H_4\}$: $H_1 : \{0,1\}^* \rightarrow \{0,1\}^{n_1}$, $H_2 : \{0,1\}^* \rightarrow \{0,1\}^{n_2}$, $H_3 : \{0,1\}^* \rightarrow \mathbb{Z}_p$, $H_4 : \{0,1\}^* \rightarrow \{0,1\}^{n_3}$; 4. Return $pub \leftarrow \{\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, g_3, u_0, v_0, w_0, U, V, W, H_1, H_2, H_3, H_4\}$.
$KeyGen(pub)$: Run by a user ID_P. 1. Choose two random elements α_P and γ_P in $\mathbb{Z}_p$; 2. $g_P \leftarrow g^{\alpha_P + \gamma_P}$; 3. $pk_P \leftarrow g_P$; 4. $sk_P^* \leftarrow \{g_1^{\alpha_P}, g_2^{\alpha_P}, g_P\}$; 5. $sk_{P_0} \leftarrow \{g_1^{\gamma_P}, g_2^{\gamma_P}, g_P\}$; 6. Return (pk_P, sk_P^*, sk_{P_0}). The user publishes pk_P, stores sk_P^* on its home device, and stores sk_{P_0} on its mobile device.
$DevUpd(pub, sk_P^*, i, j)$: Run by the home device of user ID_P. 1. Choose two random elements $r_1, r_2 \in \mathbb{Z}_p$; 2. $\tau_P \leftarrow H_1(g_P, j)$, write as $(\tau_{P_1} ... \tau_{P_{n_1}}) \in \{0,1\}^{n_1}$; 3. $\psi_P \leftarrow H_2(g_P, j)$, write as $(\psi_{P_1} ... \psi_{P_{n_2}}) \in \{0,1\}^{n_2}$; 4. $d'_{P_1} \leftarrow g_1^{\alpha_P}(u_0 \prod_{x=1}^{n_1} u_x^{\tau_{P_x}})^{r_1}$; 5. $d'_{P_2} \leftarrow g^{r_1}$; 6. $d'_{P_3} \leftarrow g_2^{\alpha_P}(v_0 \prod_{y=1}^{n_2} v_y^{\psi_{P_y}})^{r_2}$; 7. $d'_{P_4} \leftarrow g^{r_2}$; 8. Return $sk_{P_{i,j}} \leftarrow \{d'_{P_1}, d'_{P_2}, d'_{P_3}, d'_{P_4}\}$.
$MobUpd(pub, sk_{P_i}, i, j, sk_{P_{i,j}})$: Run by the mobile device of user ID_P. 1. Parse $sk_{P_{i,j}}$ as $\{d'_{P_1}, d'_{P_2}, d'_{P_3}, d'_{P_4}\}$; 2. $d_{P_1} \leftarrow g_1^{\gamma_P} \cdot d'_{P_1}$; 3. $d_{P_2} \leftarrow d'_{P_2}$; 4. $d_{P_3} \leftarrow g_2^{\gamma_P} \cdot d'_{P_3}$; 5. $d_{P_4} \leftarrow d'_{P_4}$; 6. Return $sk_{P_j} \leftarrow \{d_{P_1}, d_{P_2}, d_{P_3}, d_{P_4}, g_1^{\gamma_P}, g_2^{\gamma_P}, g_P\}$.

Table 2: $SetupPub \& KeyGen \& DevUpd \& MobUpd$ of KISC

$Signcrypt(pub, sk_{S_i}, pk_R, i, j, M)$: Run by a sender ID_S.
1. Choose two random elements $t, s \in \mathbb{Z}_p$;
2. $\sigma_0 \leftarrow e(g_1, g_R)^t \cdot M$;
3. $\sigma_1 \leftarrow g^t$;
4. $\tau_R \leftarrow H_1(g_R, j)$, write as $(\tau_{R_1}...\tau_{R_{n_1}}) \in \{0,1\}^{n_1}$;
5. $\sigma_2 \leftarrow (u_0 \prod_{x=1}^{n_1} u_x^{\tau_{R_x}})^t$;
6. $\sigma_3 \leftarrow d_{S_4}$;
7. $\theta \leftarrow H_3(\sigma_0, \sigma_1, \sigma_2, \sigma_3, g_S, g_R, i, j)$;
8. $z \leftarrow g^\theta g_3^s$;
9. $c \leftarrow H_4(z)$; write as $(c_1...c_{n_3}) \in \{0,1\}^{n_3}$;
10. $\sigma_4 \leftarrow d_{S_3}(w_0 \prod_{x=1}^{n_3} w_x^{c_x})^t$;
11. $\sigma_5 \leftarrow s$;
12. Return $\sigma \leftarrow (\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$.
$Unsigncrypt(pub, pk_S, sk_{R_j}, i, j, \sigma)$: Run by a receiver ID_R.
1. Parse σ as $(\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$;
2. $\psi_S \leftarrow H_2(g_S, i)$, write as $(\psi_{S_1}...\psi_{S_{n_2}}) \in \{0,1\}^{n_2}$;
3. $\theta \leftarrow H_3(\sigma_0, \sigma_1, \sigma_2, \sigma_3, y_S, y_R, i, j)$;
4. $z \leftarrow g^\theta g_3^{\sigma_5}$;
5. $c \leftarrow H_4(z)$; write as $(c_1...c_{n_3}) \in \{0,1\}^{n_3}$;
6. If $e(\sigma_4, g) \neq e(g_2, g_S) \cdot e(v_0 \prod_{y=1}^{n_2} v_y^{\psi_{S_y}}, \sigma_3) \cdot e(w_0 \prod_{x=1}^{n_3} w_x^{c_x}, \sigma_1)$, then return $\perp$;
7. Otherwise return $M \leftarrow \frac{\sigma_0 \cdot e(\sigma_2, d_{R_2})}{e(d_{R_1}, \sigma_1)}$.

Table 3: $Signcryt\&Unsigncrypt$ of KISC

4 Security Proofs of the Proposed Scheme

In this section, we are going to provide security proofs for our proposed KISC scheme in the standard model. The standard model and the random oracle model are useful tools in provable security. The random oracle model takes each hash function used in the cryptosystem as a random oracle. While the standard model does not have this limitation. Thus, cryptosystems proved secure in the standard model usually only rely their security on complexity assumptions, which cannot be solved in polynomial time. Below we explain some notions, which are related to our proofs.

1. The (t, ϵ) Discrete Logarithm assumption holds in $\mathbb{G}$ means the advantage of finding a solution for the Discrete Logarithm problem in $\mathbb{G}$ is at most ϵ in time t.

Security Conclusions	Assumptions
IND-CCA-M IND-CCA-H	Assume IBE is IND-CPA secure; assume H_3 is collision resistant; assume H_4 is collision resistant; assume Discrete Logarithm assumption holds in $\mathbb{G}$.
sEUF-CMA-M sEUF-CMA-H	Assume IBS is EUF-CMA secure; assume H_3 is collision resistant; assume Discrete Logarithm assumption holds in $\mathbb{G}$.
Secure Key Update	No assumptions

In the above, IBE and IBS means Waters IBE and Paterson and Schuldt IBS respectively. If N is a polynomial in k, then proposed scheme can be further extended from IND-CCA-M(sEUF-CMA-M) to strong IND-CCA-M(sEUF-CMA-M).

Table 4: A Summary of Security Conclusions on Our KISC Scheme

2. A hash function H is (t, ϵ) collision resistant means the probability of finding a collision for the hash H is at most ϵ in time t.
3. An identity-based encryption (IBE) is (t, q_e, ϵ) IND-CPA (indistinguishable under chosen plain-text attack) secure means any polynomial adversary runs in time t which chooses two equal length challenge messages M_0, M_1 and a challenge receiver identity ID^*, can only distinguish a ciphertext σ_w^* is encrypted from M_0 or M_1 with at most a negligible probability ϵ, even with the help of q_e extract queries on private key of any identities except ID^*.
4. An identity-based signature (IBS) is (t, q_s, q_e, ϵ) EUF-CMA (existentially unforgeable under chosen identity and message attack) secure means any polynomial adversary in time t can only forge a valid signature σ^* on a message M^* with signer identity ID^* with at most a negligible probability ϵ, even with the help of q_e extract queries except on ID^* and q_s signature queries except on message M^* with signer identity ID^*.

Next, we will provide rigorous proofs from five aspects according to the model definition in Section 2.2. For a clear view of this section, we abstract the security conclusions for our proposed scheme as well as the assumptions based in Table 4.

4.1 Proofs of Confidentiality

In this section, we first review the Waters IBE, then provide security proofs for the IND-CCA-M and IND-CCA-H security of the proposed KISC scheme.

4.1.1 Review of the Waters IBE

The security of our proposed KISC scheme is partly based on the IND-CPA security of Waters IBE. To simplify the analysis in proofs on confidentiality, we briefly review the Waters IBE as follows:

- **Setup:** The master public parameter for the whole system is generated as: $mpk_w \leftarrow \{\mathbb{G}, \mathbb{G}_T, e, g, g_{w1}, g_{w2}, u_0, U, H_1\}$, where almost all the elements are generated the same way as in the KISC scheme, except $g_{w1} \leftarrow g^{\alpha}$, α is a random element in $\mathbb{Z}_p$ and g_{w2} is a random element in $\mathbb{G}$. The master private key is g_{w2}^{α}.
- **KeyGen:** The private key for ID_P is generated as: $skw_P \leftarrow (dw_{P_1}, dw_{P_2}) \leftarrow (g_{w2}^{\alpha}(u_0 \prod_{i=1}^{n_1} u_i^{\tau_{P_i}})^r, g^r)$, where r is chosen randomly from $\mathbb{Z}_p$, and $\tau_P \leftarrow H_1(ID_P)$ can be written as $(\tau_{P_1} ... \tau_{P_{n_1}}) \in \{0,1\}^{n_1}$.
- **Encrypt:** The ciphertext on a message M to a receiver ID_R is $\sigma_w \leftarrow (\sigma_{w0}, \sigma_{w1}, \sigma_{w2}) \leftarrow (e(g_{w1}, g_{w2})^t \cdot M, g^t, (u_0 \prod_{i=1}^{n_1} u_i^{\tau_{R_i}})^t)$.
- **Decrypt:** To decrypt a ciphertext σ_w, ID_R computes $M \leftarrow \frac{\sigma_{w0} \cdot e(\sigma_{w2}, dw_{R_2})}{e(dw_{R_1}, \sigma_{w1})}$.

Waters IBE has been proved to be (t, q_e, ϵ) IND-CPA secure in [Waters, 2005] assuming the decisional-BDH assumption holds. IND-CPA security is a basic requirement for IBE schemes. Adversaries in IND-CPA attack game run in time t, and are only allowed to ask for at most q_e extract queries, but without extract query on a challenge identity.

4.1.2 IND-CCA-M security

Theorem 1 *Our proposed KISC scheme is $(t, q_s, q_u, q_e, \epsilon)$ IND-CCA-M secure, assuming that the Waters IBE [Waters, 2005] is $(t + \mathcal{O}(q_e + q_s + q_u), q_e, \epsilon_{enc})$ IND-CPA secure, the (t, ϵ_{dl}) Discrete Logarithm assumption holds in $\mathbb{G}$, H_3 is (t, ϵ_{H_3}) and H_4 is (t, ϵ_{H_4}) collision resistant in time t. If N is a polynomial in k, then it is strong IND-CCA-M secure. Specifically, the advantage ϵ satisfies the following condition:*

$$\epsilon \leq \frac{\epsilon_{enc}}{1 - q_u(\epsilon_{H_3} + \epsilon_{H_4} + \epsilon_{dl} + 1/p + 1/p^2)}.$$

Proof of Theorem 1. In the IND-CCA-M game, we use a simulator $\mathcal{S}$ to simulate an adversary $\mathcal{A}$'s environment. That is, $\mathcal{S}$ simulates the behavior of challenger $\mathcal{C}$ as well as oracles $\mathcal{O}_{ex}$, $\mathcal{O}_{usc}$, $\mathcal{O}_{sc}$. Besides, $\mathcal{S}$ is also an adversary in the IND-CPA attack game of Waters IBE.

Specifically, $\mathcal{S}$ simulates the IND-CCA-M game as follows. Note that as an adversary in the attack game for Waters IBE scheme, $\mathcal{S}$ is first given mpk_w.

– *Stage 1:* $\mathcal{S}$ runs the following steps to simulate the behavior of challenger $\mathcal{C}$:

1. Parse mpk_w as $\{\mathbb{G}, \mathbb{G}_T, e, g, g_{w1}, g_{w2}, u_0, U, H_1\}$;
2. $g_1 \leftarrow g_{w2}$;
3. Choose a random element $\gamma_A \in \mathbb{Z}_p$, compute $g_A \leftarrow g_{w1} g^{\gamma_A}$.
4. Choose a random element $\mu \in \mathbb{Z}_p$, compute $g_2 \leftarrow g^{\mu}$.
5. Choose a random element $\varsigma \in \mathbb{Z}_p$, compute $g_3 \leftarrow g^{\varsigma}$.
6. Choose random elements $\delta_0, \delta_1, ..., \delta_{n_2} \in \mathbb{Z}_p$, from $y = 0$ to n_2 compute $v_y \leftarrow g^{\delta_y}$, set $V \leftarrow (v_1 ... v_{n_2})$.
7. Choose random elements $k_1, ..., k_{n_3} \in \mathbb{Z}_p$, and from $x = 1$ to n_3 compute $w_x \leftarrow {g_1}^{k_x}$, set $W \leftarrow (w_1 ... w_{n_3})$.
8. Choose random elements $\rho^*, \lambda \in \mathbb{Z}_p$, compute $c^* \leftarrow H_4(g^{\rho^*})$, write c^* as $(c_1^* ... c_n^*) \in \{0,1\}^{n_3}$.
9. Compute $\tau^* \leftarrow \sum_{x=1}^{n_3} k_i c_i^*$, $w_0 \leftarrow {g_1}^{-\tau^*} g^{\lambda}$.
10. Generate H_2, H_3, H_4 according to the *Setup* algorithm of signcryption.
11. Return $pub \leftarrow \{\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, g_3, u_0, v_0, w_0, U, V, W, H_1, H_2, H_3, H_4\}$ and $pk_A \leftarrow g_A$ to $\mathcal{A}$.

– *Stage 2:* In this stage $\mathcal{S}$ simulates all the three types of oracle queries. Each type of queries is simulated as follows:

- Extract Query: When $\mathcal{A}$ submits a time period i to $\mathcal{S}$, if $i = 0$, then $\mathcal{S}$ returns $sk_{A_i} \leftarrow \{g_1^{\gamma_A}, g_2^{\gamma_A}, g_A\}$. Otherwise $\mathcal{S}$ runs the following steps:
  1. Require an extract query on identity $ID_{A_i} = (g_A, i)$ in the encryption attack game to get $skw_{A_i} = (dw_{A_1}, dw_{A_2})$;
  2. $d_{A_1} \leftarrow g_1^{\gamma_A} \cdot dw_{A_1}$;
  3. $d_{A_2} \leftarrow dw_{A_2}$;
  4. Choose a random element $r_2 \in \mathbb{Z}_p$;
  5. $\psi_A \leftarrow H_2(g_A, i)$, write as $(\psi_{A_1} ... \psi_{A_{n_2}}) \in \{0,1\}^{n_2}$;
  6. $d_{A_3} \leftarrow {g_A}^{\mu}(v_0 \prod_{y=1}^{n_2} v_y^{\psi_{P_y}})^{r_2}$;
  7. $d_{A_4} \leftarrow g^{r_2}$;
  8. Return $sk_{A_i} \leftarrow \{d_{A_1}, d_{A_2}, d_{A_3}, d_{A_4}, g_1^{\gamma_A}, g_2^{\gamma_A}, g_A\}$.

- Signcryption Query: When $\mathcal{A}$ submits (M, pk_R, i, j) to $\mathcal{S}$, $\mathcal{S}$ runs the following steps:
  1. Run Step 4 to 7 of dealing with the Extract Query to get d_{A_3} and d_{A_4};
  2. Run *Signcrypt* algorithm to get a signcryptext σ;
  3. Return σ.
- Unsigncryption Query: When $\mathcal{A}$ submits (σ, pk_S, i, j) to $\mathcal{S}$, $\mathcal{S}$ runs the following steps:
  1. Run Step 1 to 6 of *Unsigncrypt* algorithm;
  2. If $\sum_{i=1}^{n_3} k_i c_i = \tau^*$, return $\perp$.
  3. Choose a random element $r'_1 \in \mathbb{Z}_p$;
  4. $(sk'_{R_1}, sk'_{R_2}) \leftarrow (g_1^{\gamma_A} \cdot g_{w1}^{\frac{-\lambda}{\sum_{x=1}^{n_3} k_x c_x - \tau^*}} (w_0 \prod_{x=1}^{n_3} w_x^{c_x})^{r'_1}, g_{w1}^{\frac{-1}{\sum_{x=1}^{n_3} k_x c_x - \tau^*}} \cdot g^{r'_1})$.
  5. $\sigma'_4 \leftarrow \frac{\sigma_4}{g_A^{\mu} \cdot \sigma_3^{\delta_0 + \sum_{y=1}^{n_2} \delta_y \psi_{S_y}}}$.
  6. Return $M \leftarrow \frac{\sigma_0 \cdot e(\sigma'_4, sk'_{R_2})}{e(sk'_{R_1}, \sigma_1)}$.

  It is easy to verify that $(sk'_{R_1}, sk'_{R_2}) = (g_1^{\alpha_A + \gamma_A} (w_0 \prod_{x=1}^{n_3} w_x^{c_x})^{r''_1}, g^{r''_1})$ where $r''_1 = \frac{-\alpha_A}{\sum_{x=1}^{n_3} k_x c_x - \tau^*} + r'_1$, and $\sigma'_4 = (w_0 \prod_{x=1}^{n_3} w_x^{c_x})^t$.

– *Stage 3:* When $\mathcal{A}$ submits $(M_0, M_1, PK_{S^*}, sk_{S^*_{i^*}}, i^*, j^*)$ to $\mathcal{S}$, $\mathcal{S}$ runs the following steps:
  1. Forward $(M_0, M_1, ID_{R^*} = (y_A, j^*))$ to the challenger in the attack game for encryption to get a challenge $\sigma^*_w = (\sigma^*_{w0}, \sigma^*_{w1}, \sigma^*_{w2})$;
  2. $\sigma^*_0 \leftarrow e(g_1, \sigma^*_{w1})^{\gamma_A} \cdot \sigma^*_{w0}$;
  3. $(\sigma^*_1, \sigma^*_2) \leftarrow (\sigma^*_{w1}, \sigma^*_{w2})$;
  4. Choose a random element $r'^*_2 \in \mathbb{Z}_p$;
  5. Ask an Extract query to get $sk_{S^*_{i^*}} \leftarrow (d_{S^*_1}, d_{S^*_2}, d_{S^*_3}, d_{S^*_4})$;
  6. $\sigma^*_3 \leftarrow d_{S^*_4}$;
  7. $\psi_{S^*} \leftarrow H_2(y_{S^*}, i^*)$, write as $(\psi_{S^*_1} ... \psi_{S^*_{n_2}}) \in \{0,1\}^{n_2}$;

8. $\sigma_4^* \leftarrow d_{S_3^*} \cdot {\sigma_1^*}^{\lambda}$;
9. $\theta^* \leftarrow H_3(\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, g_{S^*}, g_A, i^*, j^*)$;
10. $\sigma_5^* \leftarrow \frac{\rho^* - \theta^*}{\varsigma}$;
11. Return $\sigma^* \leftarrow (\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, \sigma_4^*, \sigma_5^*)$.

It can be easily verified that $\sigma^* = (e(g_1, g_A)^{t^*} \cdot M_\beta, g^{t^*}, (u_0 \prod_{i=1}^{n_1} u_i^{\tau_{R^*_i}})^{t^*}, g^{r_2^*}, d_{S_3^*}(w_0 \prod_{i=1}^{n_3} w_i^{c_i^*})^{t^*}, s^*)$, where $c^* = H_4(g^{\theta^*} g_3^{s^*})$ is written as $(c_1^* ... c_{n_3}^*)$.

– *Stage 4:* At this Stage $\mathcal{S}$ simulates oracles the same way as at Stage 2.

– *Stage 5:* When $\mathcal{A}$ outputs a guess bit β'. $\mathcal{S}$ forwards it to the challenger of the IND-CPA attack game.

Now we analyze the errors during $\mathcal{S}$'s simulation. From the above analysis, the simulation is almost perfect except in the unsigncryption query when the signcryptext is valid and $\sum_{x=1}^{n_3} k_x c_x = \tau^*$. For each unsigncryption query, if $c \neq c^*$, then the probability that $\sum_{x=1}^{n_3} k_x c_x = \tau^*$ is $1/p$, since all the values of k_x are chosen uniformly at random and are hidden from the adversary's view. Else if $c = c^*$, then one of the following cases happens:

1. $z \neq z^*$: In this case, the adversary finds a collision for H_4;
2. $z = z^*$, $\sigma_5^* \neq \sigma_5$: In this case, the adversary finds a solution for the Discrete Logarithm problem on g_3 by computing $\log g_3 \leftarrow \frac{\theta - \theta^*}{\sigma_5^* - \sigma_5}$;
3. $z = z^*$, $\sigma_5^* = \sigma_5$, $(\sigma_0, \sigma_1, \sigma_2, \sigma_3, g_S, g_A, i, j) \neq (\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, g_{S^*}, g_A, i^*, j^*)$: In this case $\theta = \theta^*$, the adversary finds a collision for H_3;
4. $z = z^*$, $\sigma_5^* = \sigma_5$, $(\sigma_0, \sigma_1, \sigma_2, \sigma_3, g_S, g_A, i, j) = (\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, g_{S^*}, g_A, i^*, j^*)$: By the *Signcrypt* algorithm, it is easy to verify that $\sigma_4 = \sigma_4^*$. Therefore, $\sigma = \sigma^*$. According to the game rule, $\mathcal{A}$ is not allowed to ask such an unsigncryption query at Stage 4. And at Stage 2, the probability that $\mathcal{A}$ asks an unsigncryption query on a signcryptext $\sigma = \sigma^*$ is at most $1/p^2$ (which means σ is generated by choosing the same t and s).

Therefore, for each unsigncryption query, the probability that $\mathcal{A}$ makes mistakes is at most $\epsilon_{H_3} + \epsilon_{H_4} + \epsilon_{dl} + 1/p + 1/p^2$. During the whole simulation, the probability that $\mathcal{A}$ makes mistakes is at most $q_u(\epsilon_{H_3} + \epsilon_{H_4} + \epsilon_{dl} + 1/p + 1/p^2)$.

From the simulation, it is easy to see that if the simulation is perfect and $\mathcal{A}$ wins the challenge, then $\mathcal{S}$ also wins the challenge in IND-CPA attack game. Therefore, we have

$$\epsilon_{enc} \geq \epsilon \cdot (1 - q_u(\epsilon_{H_3} + \epsilon_{H_4} + \epsilon_{dl} + 1/p + 1/p^2)).$$

The running time for $\mathcal{S}$ in the IND-CPA attack game is $(t + \mathcal{O}(q_e + q_s + q_u))$, which is sum of $\mathcal{A}$'s running time t and $\mathcal{S}$'s simulation time $\mathcal{O}(q_e + q_s + q_u)$.

Now we obtain our conclusion that

$$\epsilon \leq \frac{\epsilon_{enc}}{1 - q_u(\epsilon_{H_3} + \epsilon_{H_4} + \epsilon_{dl} + 1/p + 1/p^2)}.$$

If N is a polynomial in k, we can set $q_e \leftarrow N$, then this scheme is strong IND-CCA-M secure. □

4.1.3 IND-CCA-H Security

Theorem 2 *Our proposed KISC scheme is (t, q_s, q_u, ϵ) IND-CCA-H secure, assuming that the Waters IBE scheme is $(t + \mathcal{O}(q_s + q_u), 0, \epsilon_{enc})$ IND-CPA secure, H_3 is (t, ϵ_{H_3}) and H_4 is (t, ϵ_{H_4}) collision resistant, and the (t, ϵ_{dl}) Discrete Logarithm assumption holds in $\mathbb{G}$. Specifically, the advantage satisfies the following condition:*

$$\epsilon \leq \frac{\epsilon_{enc}}{1 - q_u(\epsilon_{H_3} + \epsilon_{H_4} + \epsilon_{dl} + 1/p + 1/p^2)}.$$

Proof of Theorem 2. In the IND-CCA-H game, a simulator $\mathcal{S}$ to simulate the environment quite similar as the simulation in the proof of IND-CCA-H security, except the changes listed as follows.

- The adversary does not ask for extract queries in this game.
- γ_A is replaced with α_A throughout this proof.
- *Stage 1:* Step 7. Return pub, pk_A as well as $sk_A^* \leftarrow (g_1^{\alpha_A}, g_2^{\alpha_A}, g_A)$ to $\mathcal{A}$.

According to a similar analysis as in the proof of IND-CCA-M security, we get our conclusion that

$$\epsilon \leq \frac{\epsilon_{enc}}{1 - q_u(\epsilon_{H_3} + \epsilon_{H_4} + \epsilon_{dl} + 1/p + 1/p^2)}.$$

The running time for $\mathcal{S}$ in the attack game for encryption is $(t + \mathcal{O}(q_s + q_u))$, since there are no extract queries in this attack time. □

4.2 Proofs of Unforgeability

In this section, we first review the Paterson and Schuldt IBS, then provide security proofs for the sEUF-CMA-M and sEUF-CMA-H security of the proposed KISC scheme.

4.2.1 Review of Paterson and Schuldt IBS

The security of our proposed KISC scheme is partly based on the EUF-CMA security of Paterson and Schuldt IBS [Paterson and Schuldt, 2006]. To simplify the analysis in proofs on unforgeability, we briefly review the Paterson and Schuldt IBS as follows:

- **Setup:** The master public parameter is $mpk_s \leftarrow \{\mathbb{G}, \mathbb{G}_T, e, g, g_2, g_{s1}, v_0, w_0, V, W, H_2, H_4\}$, where almost all the elements except g_{s1} are generated the same way as in the KISC scheme, except $g_{s1} \leftarrow g^{\alpha}$, α is a random element in $\mathbb{Z}_p$. The master private key is g_2^{α}.
- **KeyGen:** The private key for ID_P is generated as: $sks_P \leftarrow \{ds_{P_1}, ds_{P_2}\} \leftarrow (g_2^{\alpha}(v_0 \prod_{j=1}^{n_2} v_j^{\psi_{P_j}})^r, g^r)$, r is chosen randomly from $\mathbb{Z}_p$, and $\psi_P \leftarrow H_2(ID_P)$ can be written as $(\psi_{P_1}...\psi_{P_{n_2}}) \in \{0,1\}^{n_2}$.
- **Sign:** The signature by signer ID_P on message M_s is $\sigma_s \leftarrow (\sigma_{s0}, \sigma_{s1}, \sigma_{s2}) \leftarrow (g^t, ds_{P_2}, ds_{P_1}(w_0 \prod_{i=1}^{n_3} w_i^{c_i})^t)$ where $c \leftarrow H_4(M_s)$ can be written as $(c_1...c_{n_3}) \in \{0,1\}^{n_3}$.
- **Verify:** To verify the validity of a signer ID_P's signature σ_s, a verifier checks if $e(\sigma_{s2}, g) = e(g_1, g_2) \cdot e(v_0 \prod_{i=1}^{n_2} v_i^{\psi_{P_i}}, \sigma_{s1}) \cdot e(w_0 \prod_{i=1}^{n_3} w_i^{c_i}, \sigma_{s0})$. If it is, return $\top$, otherwise return $\perp$.

The Paterson and Schuldt IBS scheme has been proved to be (t, q_s, q_e, ϵ) EUF-CMA secure assuming the computational-BDH assumption holds, which means for any adversary running in polynomial time t, asking q_s sign queries and q_e extract queries, ϵ, the probability of successfully forge a signature on a new message, is negligible.

4.2.2 sEUF-CMA-M security

Theorem 3 *Our proposed KISC scheme is $(t, q_s, q_u, q_e, \epsilon)$ sEUF-CMA-M secure, assuming that the Paterson and Schuldt IBS scheme is $(t + \mathcal{O}(q_e + q_s + q_u), q_e, \epsilon/3)$ EUF-CMA secure, H_3 is $(t, \epsilon/3)$ collision resistant, and the $(t, \epsilon/3)$ Discrete Logarithm assumption holds in $\mathbb{G}$. In particular, if N is a polynomial in k, then it is strong sEUF-CMA-M secure.*

Proof of Theorem 3. In the sEUF-CMA-M game, the adversary $\mathcal{A}$'s goal is to forge a valid signcryptext $\sigma^* = (\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, \sigma_4^*, \sigma_5^*)$ where $\sigma^* \neq \sigma^{(\nu)}$. Throughout this proof, the variables with superscript $^{(\nu)}$ denote the variables computed in the ν-th signcryption oracle. And the variables with superscript * denote the variables computed at Stage 3. According to the result of $\mathcal{A}$'s forgery, we divide it into four types as follows:

1. $z^* \neq z^{(\nu)}$ (for all ν from 1 to q_s),
2. $z^* = z^{(\nu)}$ and $\sigma_5^* \neq \sigma_5^{(\nu)}$ for some $\nu \in \{1, ..., q_s\}$,
3. $z^* = z^{(\nu)}$, $\sigma_5^* = \sigma_5^{(\nu)}$ and $(\sigma_0^{(\nu)}, \sigma_1^{(\nu)}, \sigma_2^{(\nu)}, \sigma_3^{(\nu)}, g_S^{(\nu)}, g_R^{(\nu)}, i^{(\nu)}, j^{(\nu)}) \neq (\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, g_A, g_{R^*}, i^*, j^*)$ for some $\nu \in \{1, ..., q_s\}$,
4. $z^* = z^{(\nu)}$, $\sigma_5^* = \sigma_5^{(\nu)}$ and $(\sigma_0^{(\nu)}, \sigma_1^{(\nu)}, \sigma_2^{(\nu)}, \sigma_3^{(\nu)}, g_S^{(\nu)}, g_R^{(\nu)}, i^{(\nu)}, j^{(\nu)}) = (\sigma_0^*, \sigma_1^*, \sigma_2^*, \sigma_3^*, g_A, g_{R^*}, i^*, j^*)$ for some $\nu \in \{1, ..., q_s\}$.

We will show that a successful type I forgery will lead to a successful attack for the Paterson and Schuldt IBS scheme, a successful type II forgery will lead to a solution for the Discrete Logarithm assumption in $\mathbb{G}$, a successful type III forgery will lead to a break for the collision-resistant hash function H_3, and the type IV forgery is always unsuccessful since in this case $\sigma_4^* = \sigma_4^{(i)}$. According to the *Signcrypt* algorithm we have $\sigma^* = \sigma^{(i)}$, which is not allowed by the game rule.

Before this attack, the simulator $\mathcal{S}$ flips a random coin to guess which kind of successful forgery (among the first three forgeries) $\mathcal{A}$ will output, then sets up $\mathcal{A}$'s input appropriately, and all our simulations are perfect.

Type I Forgery: In the sEUF-CMA-M game, we use a simulator $\mathcal{S}$ to simulate the adversary $\mathcal{A}$'s environment. That is, $\mathcal{S}$ simulates the behavior of the challenger $\mathcal{C}$ as well as the oracles $\mathcal{O}_{ex}$, $\mathcal{O}_{usc}$, $\mathcal{O}_{sc}$. $\mathcal{S}$ is also an adversary in the EUF-CMA attack game for the Paterson and Schuldt IBS scheme. Specifically, $\mathcal{S}$ simulates the sEUF-CCA-M game as follows. Note that as an adversary in the EUF-CMA attack game, $\mathcal{S}$ is first given mpk_s.

- *Stage 1:* $\mathcal{S}$ runs the following steps to simulate the challenger $\mathcal{C}$:
  1. Parse mpk_s as $\{\mathbb{G}, \mathbb{G}_T, e, g, g_2, g_{s1}, v_0, w_0, V, W, H_2, H_4\}$.
  2. Choose a random element $\gamma_A \in \mathbb{Z}_p$, compute $g_A \leftarrow g_{s1} g^{\gamma_A}$.
  3. Choose a random element $\mu \in \mathbb{Z}_p$, compute $g_1 \leftarrow g^{\mu}$.
  4. Choose a random element $\varsigma \in \mathbb{Z}_p$, compute $g_3 \leftarrow g^{\varsigma}$.
  5. Choose random elements $\delta_0, \delta_1, ..., \delta_{n_1}$ from $\mathbb{Z}_p$, compute $u_0 \leftarrow g^{\delta_0}$,$u_1 \leftarrow g^{\delta_1}$,...,$u_{n_1} \leftarrow g^{\delta_{n_1}}$, set $U = (u_1 ... u_{n_1})$.
  6. Generate H_1, H_3 according to the *Setup* algorithm of KISC scheme.

7. Return $pub \leftarrow \{\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, g_3, u_0, v_0, w_0, U, V, W, H_1, H_2, H_3, H_4\}$ and $pk_A \leftarrow g_A$ to $\mathcal{A}$.

– *Stage 2:* In this stage $\mathcal{S}$ simulates all the three types of oracles. Each type of queries is simulated as follows:
  - Extract Query: When $\mathcal{A}$ submits a time period i to $\mathcal{S}$, $\mathcal{S}$ runs the following steps:
    1. Require an extract query in the EUF-CMA attack game on $ID_P = (g_A, i)$ to get $sks_P \leftarrow (ds_{P_1}, ds_{P_2})$;
    2. $d_{A_3} \leftarrow g_2^{\gamma_A} \cdot ds_{P_1}$;
    3. $d_{A_4} \leftarrow ds_{P_2}$;
    4. Choose a random element $r_1 \in \mathbb{Z}_p$;
    5. $\tau_A \leftarrow H_1(g_A, i)$, write as $(\tau_{A_1} ... \tau_{A_{n_1}}) \in \{0,1\}^{n_1}$;
    6. $d_{A_1} \leftarrow g_A^{\mu}(v_0 \prod_{y=1}^{n_2} v_y^{\tau_{A_y}})^{r_1}$;
    7. $d_{A_2} \leftarrow g^{r_1}$;
    8. Return $sk_{A_i} \leftarrow \{d_{A_1}, d_{A_2}, d_{A_3}, d_{A_4}, g_1^{\gamma_A}, g_2^{\gamma_A}, g_A\}$.
  - Signcryption Query: When $\mathcal{A}$ submits (M, pk_R, i, j) to $\mathcal{S}$, $\mathcal{S}$ runs the following steps:
    1. Choose a random element $\phi \in \mathbb{Z}_p$;
    2. $z \leftarrow g^{\phi}$;
    3. $\mathcal{S}$ requires a signature query on $(z, ID_S = (g_A, i))$ to get a signature $\sigma_s \leftarrow (\sigma_{s0}, \sigma_{s1}, \sigma_{s2})$;
    4. $(\sigma_1, \sigma_3, \sigma_4) \leftarrow (\sigma_{s0}, \sigma_{s1}, g_2^{\gamma_A} \cdot \sigma_{s2})$;
    5. $\sigma_0 \leftarrow e(\sigma_1, g_R)^{\mu} \cdot M$;
    6. $\tau_R \leftarrow H_1(g_R, j)$, write as $(\tau_{R_1} ... \tau_{R_{n_1}}) \in \{0,1\}^{n_1}$;
    7. $\sigma_2 \leftarrow \sigma_1^{\delta_0 + \sum_{x=1}^{n_1} \delta_x \tau_{R_x}}$;
    8. $\theta \leftarrow H_3(\sigma_0, \sigma_1, \sigma_2, \sigma_3, g_A, g_R, i, j)$;
    9. $\sigma_5 \leftarrow (\phi - \theta)/\varsigma$;
    10. Return $\sigma \leftarrow (\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$.

- Unsigncryption Query: When $\mathcal{A}$ submits (σ, pk_S, i, j) to $\mathcal{S}$, $\mathcal{S}$ runs the following steps:
  1. Run Step 4 to Step 7 of dealing with the Extract Query to get d_{A_1} and d_{A_2};
  2. Run the *Unsigncrypt* algorithm, and return its result.

– *Stage 3:* When $\mathcal{A}$ outputs $(\sigma^*, pk_{R^*}, sk_{R^*_{j^*}}, i^*, j^*)$ to $\mathcal{S}$, $\mathcal{S}$ runs the following steps:
  1. Run Step 1 to Step 4 of the *Unsigncrypt* algorithm to get z^*;
  2. $(\sigma^*_{s_0}, \sigma^*_{s_1}) \leftarrow (\sigma^*_1, \sigma^*_3)$;
  3. $\sigma^*_{s_2} \leftarrow \sigma^*_4 / g_2^{\gamma_A}$;
  4. $\sigma^*_s \leftarrow (\sigma^*_{s_0}, \sigma^*_{s_1}, \sigma^*_{s_2})$;
  5. Output $(M^*_s = z^*, \sigma^*_s, ID_{S^*} = (g_A, i^*))$ in the EUF-CMA game as its forgery.

Now we can see that if $\mathcal{A}$ finally makes a successful forgery, then $\mathcal{S}$ also makes a valid forgery in the EUF-CMA game. The running time for $\mathcal{S}$ in the attack game for signature is $(t + \mathcal{O}(q_e + q_s + q_u))$, which is a sum of $\mathcal{A}$'s running time t and $\mathcal{S}$'s simulation time $\mathcal{O}(q_e + q_s + q_u)$. If N is a polynomial in k, then we can set $q_e \leftarrow N$.

Type II Forgery: In the sEUF-CMA-M game, let $\mathcal{A}$ be a type II adversary and $\mathcal{S}$ be a simulator which simulates the adversary $\mathcal{A}$'s environment. Besides, $\mathcal{S}$ is given an element $g'_3 \in \mathbb{G}$, and $\mathcal{S}$ is aimed to compute $\varsigma \in \mathbb{Z}_p$ satisfying $g'_3 = g^{\varsigma}$.

$\mathcal{S}$ simulates the game as a normal challenger in the definition except that in the Setup system step, it sets $g_3 \leftarrow g'_3$. Finally, if $\mathcal{A}$ outputs a successful type II forgery that $z^* = z^{(\nu)}$ and $\sigma^*_5 \neq \sigma_5^{(\nu)}$ for some $\nu \in \{1, ..., q_s\}$, then $\mathcal{S}$ computes $\varsigma \leftarrow ((\theta^* - \theta^{(\nu)})/(\sigma_5^{(\nu)} - \sigma^*_5)) \mod p$.

Type III Forgery: In the sEUF-CMA-M game, let $\mathcal{A}$ be a type III adversary for the signcryption scheme, and $\mathcal{S}$ be a simulator which simulates the adversary's environment. Besides, $\mathcal{S}$ is aimed to find a collision for H_3.

In this case, $\mathcal{S}$ simulates the game as a normal challenger in the definition. Finally, if $\mathcal{A}$ outputs a successful type III forgery that $z^* = z^{(i)}$, $\sigma^*_5 = \sigma_5^{(i)}$ and $(\sigma_0^{(\nu)}, \sigma_1^{(\nu)}, \sigma_2^{(\nu)}, \sigma_3^{(\nu)}, g_S^{(\nu)}, g_R^{(\nu)}, i^{(\nu)}, j^{(\nu)}) \neq (\sigma^*_0, \sigma^*_1, \sigma^*_2, \sigma^*_3, g_A, g_{R^*}, i^*, j^*)$ for some $\nu \in \{1, ..., q_s\}$, then $\mathcal{S}$ finds a collision for hash function H_3, since in this case $\theta^* = \theta^{(\nu)}$. □

4.2.3 sEUF-CMA-H Security

Theorem 4 *The KISC scheme is (t, q_s, q_u, ϵ) sEUF-CMA-H secure, assuming that the Paterson and Schuldt IBS [Paterson and Schuldt, 2006] is $(t + \mathcal{O}(q_s + q_u), q_s, 0, \epsilon/3)$ EUF-CMA secure, H_3 is $(t, \epsilon/3)$ collision resistant, and the $(t, \epsilon/3)$ Discrete Logarithm assumption holds in $\mathbb{G}$.*

Proof of Theorem 4. The analysis is almost the same as in the proof of sEUF-CMA-M security except the analysis for the Type I forgery. The difference between the analysis of Type I forgery in proof of Theorem 3 and Theorem 4 are similar as the difference between the proof of Theorem 1 and Theorem 2. That is, the adversary $\mathcal{A}$ does not ask for extract queries in this game, γ_A is replaced with α_A throughout this proof, and at Step 7 of Stage 1, it returns *pub*, pk_A as well as $sk_A^* \leftarrow (g_1^{\alpha_A}, g_2^{\alpha_A}, g_A)$ to $\mathcal{A}$.

From the above analysis we can see that if $\mathcal{A}$ finally makes a successful forgery, then $\mathcal{S}$ also makes a valid forgery in the EUF-CMA attack game. And here the running time for $\mathcal{S}$ in the EUF-CMA attack game is $(t + \mathcal{O}(q_s + q_u))$, since there is no extract query in this game. Finally, combining with the analysis for other types of forgery, we get our conclusion. □

4.3 Proof of Secure Key Update

Theorem 5 *The proposed KISC scheme satisfies secure key update.*

Proof of Theorem 5. In the proposed KISC scheme, according to the $MobUpd$ algorithm, we have $sk_{P_j} \leftarrow (d_{P_1}, d_{P_2}, d_{P_3}, d_{P_4}, g_1^{\gamma_P}, g_2^{\gamma_P}, g_P)$ where $d_{P_1} \leftarrow g_1^{\gamma_P} \cdot d'_{P_1}$, $d_{P_2} \leftarrow d'_{P_2}$, $d_{P_3} \leftarrow g_2^{\gamma_P} \cdot d'_{P_3}$, $d_{P_4} \leftarrow d'_{P_4}$.

Obviously, with the knowledge of sk_{p_j}, we can compute $sk_{P_{i,j}} \leftarrow (d'_{P_1}, d'_{P_2}, d'_{P_3}, d'_{P_4})$ where $d'_{P_1} \leftarrow d_{P_1}/g_1^{\gamma_P}$, $d'_{P_2} \leftarrow d_{P_2}$, $d'_{P_3} \leftarrow d_{P_3}/g_2^{\gamma_P}$, $d'_{P_4} \leftarrow d_{P_4}$. □

5 More Discussions

Our proposed KISC scheme smartly combines the Waters IBE and a variation of Paterson and Schuldt IBS. Recall that the signature in Paterson and Schuldt IBS is $(g^t, ds_{P_2}, ds_{P_1}(w_0 \prod_{i=1}^{n_3} w_i^{c_i})^t)$ with $c \leftarrow H_4(M_s)$. The original scheme only satisfies weak unforgeability. To achieve strong unforgeability, we apply a general transfer method proposed by Boneh, Shen and Waters [Boneh et al., 2006]. The signature in the resulting scheme is $(g^t, ds_{P_2}, ds_{P_1}(w_0 \prod_{i=1}^{n_3} w_i^{c_i})^t, s)$ with $c \leftarrow H_4(g^{M_s} g_3^s)$.

In the following ,we will discuss our proposed KISC scheme from efficiency, properties and securities. We compare our KISC scheme with regular methods constructed from Waters IBE and a variant of Paterson and Schuldt IBS in Table 5 and Table 6. Note that we do not compare our KISC scheme with other identity-based signcryption schemes which tried to prove securities in the standard model, since most of such schemes [Yu et al., 2009, Jin et al., 2010, Zhang, 2010] have been successfully attacked.

– Efficiency: In our KISC scheme, we use a random element t, and compute g^t for both encryption and signature part. In contrast to ours, a regular method would require two independent values t and t', one for the encryption part and the other one for the signature part, and thus needs to generate g^t as well as $g^{t'}$. Therefore, our signcryptext saves one element $g^{t'}$, and in signcryption algorithm saves the computation of $g^{t'}$. Table 5 provides a detailed comparison. Note that if pre-computation is allowed, the computation efficiency can be further improved.

– Properties: Besides key-insulation, our proposed KISC scheme satisfies forward security and public verifiability. According to Libert and Quisquater's point of view, to design a signcryption scheme satisfying both forward security and public verifiability is not an easy work [Libert and Quisquater, 2003]. Forward security in signcryption means even if the sender's private key is exposed, the attacker still cannot recover the message signcrypted by the sender before. To achieve this property, we allow the attacker to own the challenge sender's private key in the attack game on confidentiality. Public verifiability means the validity of signcryptext can be verified only by public information. To satisfy this, we set Step 1 to Step 6 of the Unsigncrypt algorithm to verify the validity signcryptext by only making use of public information, such as the sender and receiver's public key.

– Securities: In order to compare the securities with regular identity-based signcryption, we considers IND-CCA security on confidentiality and EUF-CMA security on unforgeability, which are regarded as basic requirements of signcryption [An et al., 2002]. From the provided security proofs we can see that our KISC scheme is both IND-CCA and EUF-CMA secure.

6 Conclusion

In this paper, we proposed a solution called key-insulated signcryption to solve the key-exposure problem in signcryption. Our scheme offers interesting properties as well as advantages in efficiency when compared with regular signcryption schemes. However, from Table 5 we can see that the advantage is not significant

	Signcryptext Size	Signcryption Cost	Unsigncryption Cost
KISC	$\|\mathbb{G}_T\| + 4\|\mathbb{G}\| + \|Z_p\|$	$1pairing + 6exp + 2H_w$	$6pairing + 2exp + 2H_w$
E-then-S	$\|\mathbb{G}_T\| + 5\|\mathbb{G}\| + \|Z_p\|$	$1pairing + 7exp + 2H_w$	$6pairing + 2exp + 2H_w$
S-then-E	$\|\mathbb{G}_T\| + 5\|\mathbb{G}\| + \|\mathbb{Z}_p\|$	$1pairing + 7exp + 2H_w$	$6pairing + 2exp + 2H_w$

E-then-S and S-then-E means Encrypt-then-Sign and Sign-then-Encrypt by Waters IBE and the variation of Paterson and Schuldt IBS respectively. $|*|$ means the length of elements in group $*$. $pairing$, exp and H_w means the computation time of doing pairing, modular exponentiation and Waters-hash once, where Waters-hash is $H_w(W, c) = w_0 \prod_{x=1}^{n} w_x^{c_x}$.

Table 5: Comparison on efficiency

	IND-CCA ?	EUF-CMA ?	Forward Security ?	Public Verifiability ?	Key Insulation ?
KISC	Yes	Yes	Yes	Yes	Yes
E-then-S	No	Yes	No	Yes	No
S-then-E	No	No	Yes	No	No

Table 6: Comparison on securities and properties

with unsigncryption. The computational cost for unsigncryption is the same as a simple combined scheme. It would be interesting to investigate more efficient key insulated signcryption schemes that admit provable security in the standard model.

References

[An et al., 2002] An J.H., Dodis Y., Rabin T.: On the security of joint signature and encryption. In Advances in Cryptology - EUROCRYPT2002, volume 2332 of Lecture Notes in Computer Science, pages 83-107, Amsterdam, The Netherlands, 2002. Springer-Verlag.

[Boyen, 2003] Boyen X.: Multipurpose Identity-Based signcryption (a swiss army knife for Identity-Based cryptography). In Advances in Cryptology -CRYPTO2003, volume 2729 of Lecture Notes in Computer Science, pages 383-399, Santa Barbara, 2003. Springer-Verlag.

[Boneh et al., 2006] Boneh D., Shen E., Waters B.: Strongly unforgeable signatures based on computational Diffie-Hellman. In Public Key Cryptography - PKC2006, volume 3958 of Lecture Notes in Computer Science, pages 229-240, New York, NY, USA, 2006. Springer-Verlag.

[Chen and Malone-Lee, 2005] Chen L., Malone-Lee J.: Improved Identity-Based signcryption. In Public Key Cryptography - PKC2005, volume 3386 of Lecture Notes in Computer Science, pages 362-379, Les Diablerets, Switzerland, 2005. Springer-Verlag.

[Dodis et al., 2003] Dodis Y., Franklin M.K., Katz J., Miyaji A., Yung M.: Intrusion-Resilient Public-Key Encryption. In CT-RSA2003, volume 2612 of Lecture Notes in Computer Science, pages 19-32, San Francisco, CA, USA, 2003. Springer-Verlag.

[Dodis et al., 2002] Dodis Y., Katz J., Xu S., Yung M.: Key-insulated public key cryptosystems. In Advances in Cryptology - EUROCRYPT2002, volume 2332 of Lecture Notes in Computer Science, pages 65-82, Amsterdam, The Netherlands, 2002. Springer-Verlag.

[Dodis and Katz et al., 2003] Dodis Y., Katz J., Xu S., Yung M.: Strong Key-Insulated signature schemes. In Public Key Cryptography - PKC2003, volume 2567 of Lecture Notes in Computer Science, pages 130-144, Miami, FL, USA, 2003. Springer-Verlag.

[ISO, 2011] The International Organization for Standardization (ISO): ISO/IEC 29150 Information technology - Security techniques - Signcryption. December 2011.

[Gamage et al., 1999] Gamage C., Leiwo J., Zheng Y.: Encrypted message authentication by firewalls. In Public Key Cryptography - PKC1999, volume 1560 of Lecture Notes in Computer Science, pages 69-81, Skamakura, Japan, 1999. Springer-Verlag.

[Itkis and Reyzin, 2002] Itkis G., Reyzin L.: SiBIR: Signer-Base Intrusion-Resilient signatures. In Advances in Cryptology - CRYPTO2002, volume 2442 of Lecture Notes in Computer Science, pages 499-514, Santa Barbara, California, USA, 2002. Springer-Verlag.

[Jin et al., 2010] Jin Z., Wen Q., Du H.: An improved semantically-secure identity-based signcryption scheme in the standard model. Computers & Electrical Engineering (CEE), 36(3):545-552, 2010.

[Libert and Quisquater, 2003] Libert B., Quisquater J-J.: New identity based signcryption schemes from pairings. In IEEE Information Theory Workshop, pages 155-158, Paris, France, 2003.

[Li et al., 2006] Li F., Xin X., Hu Y.: Id-based signcryption scheme with (t, n) shared unsigncryption. International Journal on Network Security, 3(2):155-159, 2006.

[Li et al., 2008] Li F., Xin X., Hu Y.: Id-based threshold proxy signcryption scheme from bilinear pairings. IJSN, 3(3):206-215, 2008.

[Park and Lee, 2005] Park B.N., Lee W.: Ismanet: A secure routing protocol using identitybased signcryption scheme for mobile ad-hoc networks. IEICE Transactions on Communications, E88-B(6):2548-2556, 2005.

[Park et al., 2005] Park N., Moon K., Chung K., Won D., Zheng Y.: A security acceleration using XML signcryption scheme in mobile grid web services. In ICWE2005, volume 3579 of Lecture Notes in Computer Science, pages 191-196, Sydney, Australia, 2005. Springer-Verlag.

[Paterson and Schuldt, 2006] Paterson K.G., Schuldt J.C.N.: Efficient Identity-Based signatures secure in the standard model. In ACISP2006, volume 4058 of Lecture Notes in Computer Science, pages 207-222, Melbourne, Australia, 2006. Springer-Verlag.

[Seo and Kim, 1999] Seo M., Kim K.: Electronic funds transfer protocol using domain-verifiable signcryption scheme. In ICISC1999, volume 1787 of Lecture Notes in Computer Science, pages 269-277, Seoul, Korea, 1999. Springer-Verlag.

[Waters, 2005] Waters B.: Efficient identity-based encryption without random oracles. In Advances in Cryptology - EUROCRYPT2005, volume 3494 of Lecture Notes in Computer Science, pages 114-127, Aarhus, Denmark, 2005. Springer- Verlag.

[Yu et al., 2009] Yu Y., Yang B., Sun Y., Zhu S.: Identity based signcryption scheme without random oracles. Computer Standards & Interfaces, 31(1):56-62, 2009.

[Zhang, 2010] B. Zhang.: Cryptanalysis of an identity based signcryption scheme without random oracles. Journal of Computational Information Systems, 6(6):1923-1931, 2010.

[Zheng, 1997] Y. Zheng.: Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In Advances in Cryptology - CRYPTO1997, volume 1294 of Lecture Notes in Computer Science, pages 165-179, Santa Barbara, 1997. Springer-Verlag.