PKI Based Signcryption without Pairing: an Efficient Scheme with Tight Security Reduction

S. Sree Vivek*
TCS Lab, BSB 324
Dept. of Computer Science and Engineering
IIT Madras, Chennai, India. 600036
svivek@cse.iitm.ac.in

S. Sharmila Deva Selvi
TCS Lab, BSB 324
Dept. of Computer Science and Engineering
IIT Madras, Chennai, India. 600036
sharmila@cse.iitm.ac.in

Salini Selvaraj Kowsalya
Dept of Computer Science and Engineering
Anna University
Chennai, India
salini.sk4068@gmail.com

C. Pandu Rangan
TCS Lab, BSB 324
Dept. of Computer Science and Engineering
IIT Madras, Chennai, India. 600036
prangan@cse.iitm.ac.in

Abstract

Signcryption is a cryptographic primitive that fulfill the functionalities of digital signature and public key encryption simultaneously, at a cost significantly lower than that required by the traditional sign-then-encrypt or encrypt-then-sign approach. In this paper, we address the question whether it is feasible to construct a PKI based signcryption scheme with tight security reduction in the insider security model of signcryption without pairing. This question seems to have never been addressed in the literature before. We answer the question positively in this paper. We give a novel PKI based signcryption scheme and the security is based on CDH- assumption. Ours is the first scheme of its kind which is secure in insider security model proved with tight security reduction. All other PKI based systems without pairing neither have insider security nor have tight reduction. In-spite of a slightly higher count of exponentiation, our scheme is the most efficient one currently, thanks to the tight reduction we have established to our scheme.

Keywords: signcryption, random oracle model, tight security reduction, insider security threats.

1 Introduction

In 1997, Zheng introduced the concept of Public Key Signcryption for providing both confidentiality and authentication for the same application. The use of encryption can provide message confidentiality i.e. it guarantees that the message is not tampered by the adversary during transmission. On the other hand, the use of signature guarantees source/sender authentication. Taking advantage of both encryption and signatures, signcryption was introduced which provides the integrity of message, source authentication and message confidentiality at once for applications where all these are necessary.

Since the introduction of signcryption, several schemes were proposed in the Public Key Infrastructure (PKI) setting. In 1997, Zheng [1] proposed the first and the foremost signcryption scheme in PKI based setting without giving any formal security proof. In 2002, Joosang Beak [2] formally proved the security of Zheng's signcryption in multi-user setting. They also showed that the scheme in [1] does not meet the strong security requirements. Also, in the scheme given in [1] only with the knowledge of receiver private key, signature can be verified. To overcome this, in 1998, Deng et al. [3] proposed

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, volume: 3, number: 4, pp. 72-84
*Corresponding author: Tel: +91-0091-4422575387, Email: svivek@cse.iitm.ac.in

a modified Zheng's scheme, which is computationally inefficient than the scheme in [1], but provides signature verifiability even without the knowledge of the receiver private key. In fact, it is this property that that makes the schemes in [1, 3] lack insider security.

Later, in 2002 An et al. introduced the notion of Insider Security and Outsider security [4] for signcryption. In insider security model, the indistinguishability property is preserved even if the sender's private key is compromised. Also it guarantees unforgeability when receiver's private key is leaked. This is captured in the security model by giving the private key of the sender to the adversary during confidentiality game and private key of the receiver to the adversary during unforgeability game. An et. al. also showed that it is necessary to include the public key of sender to encryption and public key of the receiver to signature for providing insider security. On the other hand, in outsider security model, security is provided against only the third parties. This implies that, the schemes which are insider secure are indeed outsider secure but not the other way. In 2005, Alexander Dent, formalized the security notion for insider security in [5] and for outsider security in [6]. Since then, several signcryption schemes were introduced and claimed to be secure against insider attacks. In [7] Libert and Quisquater introduced a new signcryption scheme and proved it to be efficient and secure using Gap Diffie Hellman assumption. In [8] Yang et al proposed an attack against [7] and gave a new scheme. In 2006, the scheme in [8] was shown insecure by Tan in [9]. In 2006, Ma proposed a scheme with insider security and that was shown insecure again by Tan [10].

In 2007, Goh et al [11] introduced the concept of Tight reduction in signature scheme which gives a concrete security reductions that give explicit bounds on the adversary's success probability as a function of expended resources. The importance for tight security reductions is that, the success probability of an adversary running in some time t is roughly *equal* to the probability of solving the underlying hard problem in roughly the same amount of time.

Even though computation of bilinear pairing has become efficient, we focus on schemes without pairing because pairing computations are generally considered more expensive and finding out pairing friendly curves are difficult [12] and most of the efficient curves and means of compressing are patented. Thus, we have only a hand full of elliptic curves that support pairing for designing cryptosystems.

In 2012, Yannick Seurin [13] showed that the security proof using Forking lemma in some sense is the best possible reduction for discrete logarithm based schemes. A security reduction based on forking lemma has inherent loss of a factor of q_h in the tightness of reduction. All the current PKI-based schemes without pairing are discrete logarithm based and hence there is no hope of getting a tight reduction for these schemes. Hence we have taken a fresh look at the design of PKI based signcryption schemes without pairing. Specifically, we design a novel signcryption scheme that has tight reduction to CDH problem.

Legend: HPA- Hard Problem Assumption, S- Signcryption, U- Unsigncryption, IND- Indistinguishability, UF- Unforgeability, GDH- Gap Diffie Hellman, CDH-Computational Diffie Hellman, GDL- Gap Discrete Logarithm, $|G|$ - The size of the group element, $|m|$- the size of the message.

1.1 Our Contribution

From table 1 and table 2, it is clear that although many secure signcryption schemes have been proposed in the random oracle model [1], [7], [8], [14], [15], none of the non-pairing Public Key Infrastructure based signcryption schemes offer neither tight security reduction nor insider security. In this paper, we propose a PKI- based signcryption scheme in the random oracle model that does not use bilinear pairing and having tight reduction to CDH problem. Now, let us consider only the non-pairing based signcryption schemes. For the Zheng's scheme, the security proof uses forking lemma. Again, based on

Table 1: Review of existing signcryption schemes in the random oracle model with pairing

Scheme	Complexity		HPA	Ciphertext Size	Tight	Security Level
	Exp	Pairing				
Libert et al [7]	4(S:3 U:1)	2(S:0 U:2)	CDH	$3\|G\|+\|m\|$	Yes Yes	Outsider Secure
Yang et al. [8]	6(S:4 U:2)	2(S:0 U:2)	GDH	$3\|G\|+\|m\|$	Yes	Not Secure
Chung ki Li [14]	4(S:3 U:1)	2(S:0 U:2)	GDH	$3\|G\|+\|m\|$	Yes	Insider Secure
Ma [15]	3(S:2 U:1)	2(S:0 U:2)	q-SDH	$\|G\|+\|Z_q\|$	Yes	Not Secure

Table 2: Review of existing signcryption schemes in the random oracle model without pairing

Scheme	No of Exp	HPA	Ciphertext Size	Tight	Security Level
Zheng [1]	3 (S:1 U:2)	IND-DDH	$2\|Z_q\|+\|m\|$	No	Outsider
Baek et al [2]	4 (S:1 U:3)	IND-GDH, UF-DL	$2\|Z_q\|+\|m\|$	No	Outsider
Zheng [16]	4 (S:1 U:3)	IND-GDH, UF-GDL	$2\|Z_q\|+\|m\|$	No	Outsider
Ours	9 (S:3U: 6)	CDH	$2\|G\|+\|m\|+\|Z_q\|$	Yes	Insider Secure in multiuser setting

the observation in Goh et al. [11], the ciphertext size should be 2^3 times that of the original ciphertext size to ensure security. For comparison purposes, in Discrete Logarithmic assumptions, we set the size of $|Z_q|$ to be 1000 bits.

It is clear that our scheme is much more efficient because though there are nine exponentiations in our scheme, they are done *modulo* 1000 bit numbers while the 4 exponentiations in Zheng's scheme are done *modulo* 8000 bit numbers. Also, in our scheme, if the signcryption is invalid, during the process of unsigncryption it can be found out after doing 3 exponentiations itself. We prove the security of our scheme based on CDH (Computational Diffie Hellman) assumption in the multi-user setting model. In sum, the security reduction of Zheng's [1] and Baek's [2] schemes are lose reductions to weaker problems while our scheme is offering a tight reduction to a stronger problem. The details are summarized in table 3.

Note that it is easy to show that the schemes in [1], [2] and [16] are not insider secure. This is not a flaw of these schemes. At the time of designing of these systems, the security definitions of signcryption did not have insider security.

Table 3: Signcryption schemes without pairing

Scheme	No of Exp	Hard Problem Assumption	Key Size	Ciphertext Size
Zheng [1]	3	IND-DDH	8000	$2^3(2000)+\|m\| = 16000+\|m\|$
Baek et al [2]	4	IND-GDH, UF-DL	8000	$2^3(2000)+\|m\| = 16000+\|m\|$
Zheng [16]	4	IND-GDH, UF-GDL	8000	$2^3(2000)+\|m\| = 16000+\|m\|$
Ours	9	CDH	1000+1000= 2000	$2*1000+1000+\|m\| = 3000+\|m\|$

2 Preliminaries

In this section we review the computational assumptions, generic model and security model for signcryption.

2.1 Computational Assumptions

2.1.1 Computation Diffie-Hellman Problem (CDH)

Let g be the generator of the group $\mathbb{G}$. Then given $(g, g^a, g^b) \in \mathbb{G}^3$ for unknown $a, b \in \mathbb{Z}_q^*$, the Computational Diffie Hellman problem is to find g^{ab} in polynomial time.

Definition The advantage of solving the CDH problem in $\mathbb{G}$ by any probabilistic polynomial time algorithm $\mathscr{A}$ is defined as:

$$Adv_{\mathscr{A}}^{CDH} = Pr\left[\mathscr{A}(g, g^a, g^b) = g^{ab} \mid a, b \in \mathbb{Z}_q\right]$$

The *CDH Assumption* is that, the advantage of solving $Adv_{\mathscr{A}}^{CDH}$, for any probabilistic polynomial time algorithm $\mathscr{A}$, is negligibly small.

2.2 Generic Model:

A PKI based signcryption scheme in the common reference model consists of the following four probabilistic polynomial algorithms.

- ***Initialization:*** This algorithm takes as input the security parameter 1^k and returns the public parameters *params* of the system.
- ***KeyGen Algorithm*** $\mathbb{G}_k$: This algorithm takes as input the global params *params* and returns the public/ private key pair (pk_u, sk_u) of the user. This algorithm is run by the user
- ***Signcrypt***(m, sk_S, pk_R): Let S be the sender with private-public key pair $langlesk_S, pk_S\rangle$ and R be the receiver with private-public key pair $\langle sk_R, pk_R\rangle$. This algorithm takes a message m from some message space $\mathbb{M}$, the private key of the sender sk_S and the public key of the receiver pk_R, and outputs a signcryption $C = Signcrypt(m, sk_S, pk_R)$ in some signcryption space C.

- ***Unsigncrypt***(C, sk_R, pk_S)**:** This takes as input a signcryption C, the private key of the receiver sk_R and the public key of the sender pk_S, and outputs the message $m = Unsigncrypt(C, sk_R, pk_S)$ if C is a valid signcryption of message m from S to R else it outputs $Invalid$.

3 Security Model

In this section, we give the security model for PKI based signcryption schemes.

3.1 IND-iCCA Security:

This IND-iCCA security of signcryption is formally defined as a game between an adversary $\mathscr{A}$ and a challenger $\mathscr{C}$.

Setup: $\mathscr{C}$ invokes the $Initialization$ algorithm and sets up the global parameters $params$. It also generates a target user's key pair (pk_T, sk_T) using $KeyGen$ algorithm and gives $params, pk_T$ to the adversary $\mathscr{A}$ and keeps sk_T to himself.

Phase I: In this phase the adversary $\mathscr{A}$ is allowed to access the $Signcrypt$ and $Unsigncrypt$ oracle during the Training phase.

KeyGen Queries: When $\mathscr{A}$ queries this oracle, $\mathscr{C}$ executes the $KeyGen(.)$ algorithm and return $\langle pk_i, sk_i \rangle$ to $\mathscr{A}$. It should be noted that the adversary $\mathscr{A}$ should not ask $KeyGen$ queries for the target user T.

Unsigncrypt Queries:When $\mathscr{A}$ queries with a signcryption C, a sender public key pk_S and receiver public key pk_R, $\mathscr{C}$ performs $Unsigncrypt(C, sk_R, pk_S)$ and sends back the output to $\mathscr{A}$.

Signcrypt Queries: : When $\mathscr{A}$ makes a query with a message m, a sender public key pk_S and receiver public key pk_R, $\mathscr{C}$ responds with $Signcrypt(m, sk_S, pk_R)$ and sends back the output to $\mathscr{A}$

Challenge: In this phase $\mathscr{A}$ chooses two plain texts $\{m_0, m_1\}$ of equal length and the sender $\mathbb{S}$ and sends to the $\mathscr{C}$. Now $\mathscr{C}$ selects a random bit $b \in \{0,1\}$ and returns the challenge ciphertext $\mathscr{C}^* = Signcrypt(m_b, sk_S, pk_T)$ to $\mathscr{A}$.

Phase II: In this phase $\mathscr{A}$ is allowed to access the oracle as in phase I, but with the restriction that it should not ask the query $Unsigncrypt(\mathscr{C}^*, pk_S, pk_T)$.

Guess Phase: $\mathscr{A}$ now returns a bit $b^{'}$ to $\mathscr{C}$. $\mathscr{A}$ wins the game if $b = b^{'}$. The advantage of $\mathscr{A}$ in the above game is defined as:

$$Adv_{\mathscr{A}}^{IND-iCCA} = |Pr\left[b^{'} = b\right] - \frac{1}{2}|$$

Definition: We say that a signcryption scheme is $IND-iCCA$ secure if $Adv_{\mathscr{A}}^{IND-CCA}$ is negligible for any PPT adversary $\mathscr{A}$.

Remark: It should be noted that in the above game the adversary is allowed to know the private key of the sender used for generating the challenge signcryption. This captures the insider security notion of confidentiality. Also since $\mathscr{A}$ is free to choose any pk_S during Phase I, Phase II, Challenge Phase, this ensures security in the multiple user setting since the challenger unsigncrypts ciphertexts from multiple sender to the single receiver.

3.2 sUF-iCMA Security:

This security notion is formalized by the following game between an adversary $\mathscr{A}$ and a challenger $\mathscr{C}$.

Setup: $\mathscr{C}$ invokes the $Initialization$ algorithm and sets up the global parameters $params$. It also generates using $KeyGen$ algorithm a key pair (pk_T, sk_T) and gives $params, pk_T$ to the adversary $\mathscr{A}$.

Training Phase: In this phase the adversary $\mathscr{A}$ can adaptively submit *Signcrypt* queries and *Unsigncrypt* queries as in Phase 1 of the confidentiality game.

Output: $\mathscr{A}$ outputs the forgery C^*, m^* from the sender T to the receiver R such that it passes the verification test. The advantage of $\mathscr{A}$ attacking the scheme is defined as follows:

$$Adv_{\mathscr{A}}^{sUF-CMA} = Pr[Unsigncrypt(C^*, sk_R, pk_T) = m^*]$$

Remark: Here the adversary can freely choose the public key of the receiver R and the message m and ask for the signcryption queries $Signcrypt(m.pk_T, pk_R)$. Since $\mathscr{C}$ gives the signcryption queries for multiple receiver, this model ensures security in the multiple user setting. Also since the adversary $\mathscr{A}$ knows the private key of the receiver, this ensures strong unforgeability in the insider security model.

4 Proposed Signcryption Scheme

In this section, we propose our novel scheme for signcryption.

1. **Initialize:** Let p and q be two large prime numbers such that q divides $p-1$. Let $\mathbb{G}$ be a cyclic group of prime order p. Let g be the generator of the group $\mathbb{G}$. Let H_1, H_2, H_3 be three cryptographic hash functions defined by:

$$H_1 : \mathbb{G}^2 \times \{0,1\}^{l_m} \times \mathbb{G}^6 \to \mathbb{Z}_q$$

$$H_2 : \mathbb{G}^5 \to \mathbb{G}$$

$$H_3 : \mathbb{G}^4 \to \mathbb{Z}_q$$

2. **KeyGen:** Let the public/private key of sender S be $pk_S = \langle pk_{S_1}, pk_{S_2} \rangle$, $sk_S = \langle sk_{S_1}, sk_{S_2} \rangle$ and that of receiver R be $pk_R = \langle pk_{R_1}, pk_{R_2} \rangle$, $sk_R = \langle sk_{R_1}, sk_{R_2} \rangle$ where

$$\langle sk_{S_1}, sk_{S_2} \rangle = \langle x_1, x_2 \rangle, \langle pk_{S_1}, pk_{S_2} \rangle = \langle g^{x_1}, g^{x_2} \rangle$$

$$\langle sk_{R_1}, sk_{R_2} \rangle = \langle y_1, y_2 \rangle, \langle pk_{R_1}, pk_{R_2} \rangle = \langle g^{y_1}, g^{y_2} \rangle$$

   Here $x_1, x_2, y_1, y_2 \in_R \mathbb{Z}q$. During signcryption from sender S to receiver R we will use only the first private key $\langle sk_{S_1} \rangle$ of the sender S and both the public keys ($\langle pk_{R_1}, pk_{R_2} \rangle$) of the receiver R.

3. **Signcrypt**$(m, sk_{S_1}, pk_{R1}, pk_{R2})$**:** This algorithm is used to signcrypt a message m, from sender S to receiver R. This algorithm takes the message m, the sender private key sk_{S_1} and the public key of the receiver pk_R and performs the following:

   - Choose $r \in \mathbb{Z}_q$ randomly and compute $C_1 = g^r \in \mathbb{G}$.
   - Compute $H = H_2(C_1, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2})$.
   - Compute $C_2 = H^r$, $Q = H^{sk_{S_1}}$.
   - Now, compute $C_3 = m \oplus H_3(C_1, (pk_{R_1})^r, (pk_{R_2})^r, Q)$.
   - Compute $h_1 = H_1(C_1, C_2, C_3, (pk_{R_1})^r, (pk_{R_2})^r, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2})$.
   - Generate $C_4 = sk_{S_1} + rh_1$.
   - Output the signcryption $C = \langle C_1, C_2, C_3, C_4 \rangle$.

4. **Unsigncryption**$(C, sk_R, pk_{S1}, pk_{S2})$: This algorithm is used to unsigncrypt a signcryption C, which is from sender S with public key pk_S to the receiver R with public/private key pair (pk_R, sk_R). In order to unsigncrypt, the receiver performs the following:

- Compute $H = H_2(C_1, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2})$.
- Calculate $h_1 = H_1(C_1, C_2, C_3, (C_1)^{sk_{R_1}}, (C_1)^{sk_{R_2}}, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2})$.
- If $g^{C_4}/C_1^{h_1} \stackrel{?}{=} pk_{S_1}$
  - Calculate $Q^{'} = H^{C_4}/C_2^{h_1}$.
  - Extract $m = C_3 \oplus H_3(C_1, (C_1)^{sk_{R_1}}, (C_1)^{sk_{R_2}}, Q^{'})$.
  - Return m.
- Else,
  - Abort and return $\perp$.

Correctness: To show the correctness of the scheme, we have to show that $Q^{'}$ computed during unsigncryption is $H^{sk_{S_1}}$. The correctness follows:

$$Q^{'} = \frac{H^{C_4}}{C_2^{h_1}} = \frac{H^{sk_{S_1}+rh_1}}{(H^r)^{h_1}} = \frac{H^{sk_{S_1}}H^{rh_1}}{H^{rh_1}} = H^{sk_{S_1}}$$

5 Security Proof

In this section, we provide the formal security proof for the insider secure signcryption scheme given in section 4 in the multiuser setting model.

Theorem 5.1. *If an IND-iCCA adversary $\mathscr{A}_I$ has an advantage ε_1 against the IND-iCCA2 security of the proposed scheme, asking $qH_i(i = 1,2,3)$ hash queries to random oracles $OH_i(i = 1,2,3)$, then there exist an algorithm $\mathscr{C}$ that solves the CDH problem with an advantage $\varepsilon^{'} = \varepsilon_1$.*

Proof: A challenger $\mathscr{C}$ is challenged with an instance of the CDH problem. i.e Given g^a, g^b, $\mathscr{C}$ has to find g^{ab} where a, b are not known to the $\mathscr{C}$. Let $\mathscr{A}_I$ be the adversary who is capable of breaking the IND-iCCA2 security of the proposed scheme. $\mathscr{C}$ can make use of $\mathscr{A}_I$ to find the solution of the CDH problem instance by playing the following interactive game with $\mathscr{A}_I$.

Setup: $\mathscr{C}$ sets the public key of some target user T as $pk_T = \langle pk_{T_1}, pk_{T_2} \rangle = \langle g^a, g^{a-z} \rangle$, where $z \in {Z_q}^*$ however its corresponding private keys are $\langle sk_{T_1}, sk_{T_2} \rangle = \langle a, a-z \rangle$ is not known even to $\mathscr{C}$. $\mathscr{C}$ designs the hash functions $H_i(i = 1,2,3)$ as random oracles $OH_i(i = 1,2,3)$ respectively. In order to maintain the consistency between the responses to the hash queries and *KeyGen* queries $\mathscr{C}$ maintains lists $L_i(i = 1,2,3)$ and L_k respectively. Finally $\mathscr{C}$ gives $\langle params,\ pk_{T1}, pk_{T2} \rangle$ to $\mathscr{A}_I$

Phase I: $\mathscr{A}_I$ performs a series of polynomially bounded number of hash queries, *KeyGen* queries, *Signcrypt* queries and *Unsigncrypt* queries adaptively in this phase. The various oracles are described below.

KeyGen: When $\mathscr{A}_I$ makes a query for signcryption key or unsigncryption key of a user U_i, $\mathscr{C}$ chooses $k_{1i}, k_{2i} \in_R \mathbb{Z}_q^*$ and calculates $g^{k_{1i}}$ and $g^{k_{2i}}$ and stores $\langle g^{k_{1i}}, k_{1i}, g^{k_{2i}}, k_{2i} \rangle$ in the list L_k and gives the public/private key pair of user $U_i$as $\langle g^{k_{1i}}, k_{1i}, g^{k_{2i}}, k_{2i} \rangle$ to $\mathscr{A}_I$.

H_1 **Oracle:** When $\mathscr{A}_I$ makes a query to the H_1 Oracle, $\mathscr{C}$ checks whether a tuple of the form $\langle C_1, C_2, C_3$ $(pk_{R_1})^r, (pk_{R_2})^r, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2}, h_{1i} \rangle$ exists in list $\mathbb{L}_1$. If so, returns h_{1i} to $\mathscr{A}_I$, else it chooses a random $h_{1i} \in \mathbb{Z}_q^*$ and adds the tuple $\langle C_1, C_2, C_3, (pk_{R_1})^r,\ (pk_{R_2})^r pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2}, h_{1i} \rangle$ in list $\mathbb{L}_1$ and outputs h_{1i} to $\mathscr{A}_I$.

H_2 **Oracle:** When $\mathscr{A}_I$ makes a query to the H_2 Oracle, $\mathscr{C}$ checks whether a tuple of the form $\langle C_1, pk_{S1}, pk_{S2}$, $pk_{R1}, pk_{R2}, h_{2i}, t_i \rangle$ exists in the list $\mathbb{L}_1$. If so, returns h_{2i} to $\mathscr{A}_I$, else it chooses a random $t_i \in \mathbb{Z}_q^*$, computes $h_{2i} = g^{t_i}$ and adds the tuple $\langle C_1, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2}, h_{2i}, t_i \rangle$ in list $\mathbb{L}_2$ and outputs h_{2i} to $\mathscr{A}_I$.

H_3 **Oracle:** When $\mathscr{A}_I$ makes a query to the H_3 Oracle, $\mathscr{C}$ checks whether a tuple of the form $\langle C_1, {pk_{R_1}}^r, {pk_{R_2}}^r, H^{sk_{S_1}}, h_{3i}\rangle$ exists in $\mathbb{L}_3$. If so, returns h_{3i} to $\mathscr{A}_I$, else it chooses a random $h_{3i} \in \mathbb{Z}_q^*$ and adds the tuple $\langle C_1, {pk_{R_1}}^r, {pk_{R_2}}^r, H^{sk_{S_1}}, h_{3i}\rangle$ in list $\mathbb{L}_3$ and outputs h_{3i} to $\mathscr{A}_I$.

UnSigncryption Oracle: When $\mathscr{A}_I$ submits an unsigncryption query with receiver $pk_R = pk_T$; i.e, the receiver is the target user. Here, $\mathscr{C}$ does not know the private key corresponding to pk_T; i.e, $sk_{T_2} = a - z$. $\mathscr{C}$ performs the following to respond to this query.

When $pk_R = pk_T$ i.e $\mathscr{A}_I$ submits queries in which T is the receiver. Assume that $\mathscr{A}_I$ has given $C = \langle C_1, C_2, C_3, C_4\rangle$ to $\mathscr{C}$ to unsigncrypt. Assume that the sender is A and receiver is T. Since $\mathscr{C}$ does not know the private keys of T, $\mathscr{C}$ has to simulate the unsigncryption. This is done as follows:

- Get the private key $sk_A = \langle sk_{A_1}, sk_{A_2}\rangle$ of A from the list $\mathbb{L}_k$ corresponding to the entry $pk_A = \langle pk_{A_1}, pk_{A_2}\rangle$.
- Calculate $H = H_2(C_1, pk_{A1}, pk_{A2}, pk_{T1}, pk_{T2})$.
- Let $\delta = H^{sk_{A_1}}$.
- Let $\bar{L}$ be the set of tuples corresponding to C_1, δ in $\mathbb{L}_3$; i.e, $\langle C_1, -, -, \delta, -\rangle \in \mathbb{L}_3$. Let us denote these tuples as $\langle C_1, \alpha_i, \pi_i, \delta, \gamma_i\rangle$
  1. Calculate $h_{1i} = H_1(C_1, C_2, C_3, \alpha_i, \pi_i, pk_{A1}, pk_{A2}, pk_{T1}, pk_{T2})$.
  2. Check if $g^{C_4}/{C_1}^{h_{1i}} \stackrel{?}{=} pk_{A_1}$ if false fetch the next record from the list $\bar{L}$ and goto step 1.
  3. Calculate $r_i = \frac{C_4 - sk_{A_1}}{h_{1i}}$.
  4. Check if $\alpha_i \stackrel{?}{=} {pk_{T_1}}^{r_i}$ if false fetch the next record from the list $\bar{L}$ and goto step 1.
  5. Check if $\pi_i \stackrel{?}{=} {pk_{T_2}}^{r_i}$ if false fetch the next record from the list $\bar{L}$ and goto step 1.
  6. Check if $g^{r_i} \stackrel{?}{=} C_1$ if false fetch the next record from the list $\bar{L}$ and goto step 1.
  7. Check if $H^{r_i} \stackrel{?}{=} C_2$ if false fetch the next record from the list $\bar{L}$ and goto step 1.

If (At-least one record in $\bar{L}$ passes all the tests in steps $2, 4, 5, 6$ and 7)

- Calculate $m_i = C_3 \oplus \gamma_i$.
- Return m_i

Else (No record in $\bar{L}$ satisfies all the conditions in steps $2, 4, 5, 6$ and 7.)

- Return $\perp$.

In this way all the unsigncryption queries from any sender to the target receiver T will be answered. If the receiver is not T, then unsigncryption can be done as per the protocol because $\mathscr{C}$ will know the private key of the receiver.

Signcryption Oracle: Since $\mathscr{C}$ knows the signcryption key of all users, $\mathscr{C}$ runs the actual algorithm and outputs the result to $\mathscr{A}_I$.

Challenge Phase: When $\mathscr{C}$ receives the messages m_0, m_1 it chooses a random message m_b where $b \in \{0, 1\}$ and creates the challenge signcryption as follows (with S as the sender identity and T as the target receiver identity).

1. Get the private key $sk_S = \langle sk_{S_1}, sk_{S_2}\rangle$ of the sender S corresponding to the public key $pk_S = \langle pk_{S_1}, pk_{S_2}\rangle$.

2. Set $C_1^* = g^b$, where g^b is the part of the CDH instance C wants to solve.
3. Choose random $t_j \in Z_q{}^*$ and set $H^* = H_2(C_1, pk_{S1}, pk_{S2}, pk_{T1}, pk_{T2}) = g^{t_j}$. Also store the tuple $\langle C_1, pk_{S1}, pk_{S2}, pk_{T1}, pk_{T2}, g^{t_j}, t_j \rangle$ in the list $\mathbb{L}_2$.
4. Set $C_2^* = (g^b)^{t_j}$.
5. Choose a random $h_3{}^* \in \mathbb{Z}_q{}^*$ and set $C_3^* = m_b || v \oplus h_3{}^*$ and store the tuple $\langle C_1, -, -, H^{sk_{S_1}}, h_3{}^* \rangle$ in the list $\mathbb{L}_3$.
6. Choose random $h_1{}^* \in Z_q{}^*$ and store the tuple $\langle C_1, C_2, C_3, -, -, pk_{S1}, pk_{S2}, pk_{T1}, pk_{T2}, h_1 \rangle$) in the list $\mathbb{L}_1$.
7. Choose random $C_4{}^* \in Z_q{}^*$

Output $C^* = \langle C_1, C_2, C_3, C_4 \rangle$ to $\mathscr{A}_I$.

Phase II: In this phase $\mathscr{A}_I$ is allowed to adaptively query the oracles as in Phase I, with the constraint that it should not ask $\mathscr{C}$ the query $Unsigncrypt(C^*, pk_{T1}, pk_{T2}, pk_{A1}, pk_{A2})$. When $\mathscr{A}_I$ queries H_3 oracle(i.e) $H_3(C_1, X_1, X_2, H^{sk_{S_1}})$ satisfying

$$X_1 = X_2 C_1{}^z$$

then send h_3^* to $\mathscr{A}_I$.

Guess Phase: At the end of Phase II, $\mathscr{A}_I$ outputs a bit δ'. If $\delta' = \delta$, then $\mathscr{C}$ retrieves the tuple $\langle C_1, X_1, X_2, H^{sk_{S_1}} \rangle$ from the list $\mathbb{L}_3$ satisfying the following condition

$$X_1 \stackrel{?}{=} X_2 (C_1)^z$$

Then output X_1 as the solution to the CDH problem instance. Hence, $\mathscr{C}$ solves the CDH problem with a probability $\varepsilon' = \varepsilon_1$, where ε_1 is the advantage of the adversary in breaking the IND-iCCA2 security of the scheme.

Correctness: Below we show that X_1 obtained above is g^{ab}.

- Since $pk_{T1} = g^a$ and $C_1 = g^b$, the value of X_1 should be g^{ab}
- Since $pk_{T2} = g^{a-z}$ and $C_1 = g^b$, the value of X_2 should be g^{ab-zb}
- Hence, $X_1 = X_2 (C_1)^z = g^{ab-zb} (g^b)^z = g^{ab}$

Since ε_1 is non-negligible, $\varepsilon^{'}$ is also non-negligible. This implies the probability of solving CDH by $\mathscr{C}$ is also non-negligible.

Theorem 5.2. *If an sUF-iCMA adversary $\mathscr{A}_{II}$ has an advantage ε_2 against the sUF-iCMA security of the proposed scheme, asking $qH_i (i = 1,2,3)$ hash queries to random oracles $OH_i (i = 1,2,3)$, then there exist an algorithm $\mathscr{C}$ that solves the CDH problem with an advantage $\varepsilon^{''} = \varepsilon_2$*

Proof: A challenger $\mathscr{C}$ is challenged with an instance of the CDH problem. i.e Given g^a, g^b, $\mathscr{C}$ has to find g^{ab} where a, b are not known to the $\mathscr{C}$. Let $\mathscr{A}_{II}$ be the adversary who is capable of breaking the sUF-iCMA security of the proposed scheme. $\mathscr{C}$ can make use of $\mathscr{A}_{II}$ to find the solution of the CDH problem instance by playing the following interactive game with $\mathscr{A}_{II}$.

Setup: $\mathscr{C}$ sets the public key of some target user T as $pk_{T_1} = g^a$ (the first instance of CDH problem), hence its corresponding private key $sk_{T_1} = a$ is not known to $\mathscr{C}$ since the $\mathscr{C}$ does not know a. To answer the signcryption and unsigncryption queries asked by $\mathscr{A}_{II}$ correctly, $\mathscr{C}$ designs the hash functions

$H_i(i = 1,2,3)$ as random oracles $OH_i(i = 1,2,3)$ respectively. In order to maintain the consistency between the responses to the hash queries and *KeyGen* queries $\mathscr{C}$ maintains lists $L_i(i = 1,2,3)$ and L_k respectively. Finally $\mathscr{C}$ gives $\langle params,\ pk_{T1}, pk_{T2}\rangle$ to $\mathscr{A}_{II}$.

Phase I: $\mathscr{A}_{II}$ performs a series of polynomially bounded number of hash queries, *KeyGen* queries, Signcrypt and Unsigncrypt queries in an adaptive fashion in this phase. The oracles and queries allowed are described below.

Key Gen and H_1 Queries: Answered in the same way as in Theorem 1 - Phase I

H_2 Queries: Similarly, to respond to this query, $\mathscr{C}$ checks whether a tuple of the form $\langle C_1, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2}, h_{2i}, t_i\rangle$ exists in $\mathbb{L}_1$. If so, returns h_{2i} to $\mathscr{A}_{II}$, else it chooses a random $t_i \in \mathbb{Z}_q^*$, computes $h_{2i} = \left(g^b\right)^{t_i}$ and adds the tuple $\langle C_1, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2}, h_{2i}, t_i\rangle$ in list $\mathbb{L}_2$ and outputs h_{2i} to $\mathscr{A}_{II}$.

H_3 Queries: Also, to respond to this query, $\mathscr{C}$ checks whether a tuple of the form $\langle C_1, C_1^{sk_{R_1}}, C_1^{sk_{R_2}}, H^{sk_{S_1}}, h_{3i}\rangle$ exists in $\mathbb{L}_3$. If so, returns h_{3i} to $\mathscr{A}_{II}$, else it chooses a random $h_{3i} \in \mathbb{Z}_q^*$ and adds the tuple $\langle C_1, C_1^{sk_{R_1}}, C_1^{sk_{R_2}}, H^{sk_{S_1}}, h_{3i}\rangle$ in list $\mathbb{L}_3$ and outputs h_{3i} to $\mathscr{A}_{II}$.

Signcryption Oracle: If $\mathscr{A}_{II}$ submits a signcryption query for a message m in which T acts as the sender and let any arbitrary user A be the receiver, then the signcrypt oracle $Signcrypt(m, pk_T, pk_A)$ is simulated by the challenger as follows:

1. Get the private key $sk_A = \langle sk_{A_1}, sk_{A_2}\rangle$ of A from the list $\mathbb{L}_k$ corresponding to the entry $pk_A = \langle pk_{A_1}, pk_{A_2}\rangle$.
2. Choose a random C_4 and h_1, and compute C_1 as $C_1 = \left(g^{C_4}/g^a\right)^{\frac{1}{h_1}}$ so that r_i will become $r_i = \frac{C_4 - a}{h_1}$.
3. Get $H = H_2(C_1, pk_{T1}, pk_{T2}, pk_{A1}, pk_{A2})$ from the list $\mathbb{L}_2$.
4. Choose random $P \in \mathbb{G}$ where $P = H^d$, $d \in \mathbb{Z}_q{}^*$.
5. Compute C_2 as $C_2 = \left(H^{C_4}/P\right)^{\frac{1}{h_1}}$.
6. Set $C_3 = m \oplus h_{3i}$ where $h_{3i} \in Z_q{}^*$ and store the tuple $\langle C_1, C_1{}^{sk_{A_1}}, C_1{}^{sk_{A_2}}, P, h_{3i}\rangle$ in the list $\mathbb{L}_3$.
7. Store the tuple $\langle C_1, C_2, C_3, C_1{}^{sk_{A_1}}, C_1{}^{sk_{A_2}}, pk_{T1}, pk_{T2}, pk_{A1}, pk_{A2}, h_1\rangle$ in the list $\mathbb{L}_1$.

Now the challenger sends $\langle C_1, C_2, C_3, C_4\rangle$ as the ciphertext to the adversary A_{II}. Hence C is the signcryption of the message m corresponding to the target receiver A from the sender T. On receiving the ciphertext, the adversary $\mathscr{A}_{II}$ checks for the correctness of the ciphertext; i.e, whether C is the signcryption of the message m with T as the sender and A as the receiver (since it possesses the private key of the receiver A)

1. Calculate $h_1 = H_1(C_1, C_2, C_3, C_1{}^{sk_{A_1}}, C_1{}^{sk_{A_2}}, pk_{T1}, pk_{T2}, pk_{A1}, pk_{A2})$.
2. Calculate $P' = H^{C_4}/C_2{}^{h_1}$.
3. Now we show that the simulated signcryption passes the validation test mentioned in the $Unsigncrypt$ algorithm of the scheme; i.e, we check $g^{C_4}/C_1^{h_1} \stackrel{?}{=} pk_{T_1}$.
4. Since the verification holds good, the simulation is a perfect one. Now the adversary will confirm the signcryption to be valid and hence it will extract the message $m = C_3 \oplus H_3(C_1, (C_1)^{sk_{A_1}}, (C_1)^{sk_{A_2}}, P')$.

Correctness:

$$\frac{g^{C_4}}{C_1^{h_1}} = \frac{g^{C_4}}{g^{\left(\frac{C_4-a}{h1}\right)h_1}} = \frac{g^{C_4}}{g^{C_4-a}} = g^a = pk_{T_1}$$

Unsigncryption Oracle: Since $\mathscr{C}$ knows the unsigncryption key of all users, $\mathscr{C}$ runs the actual algorithm and outputs the result to $\mathscr{A}_{II}$.

Forgery Phase: When the adversary outputs a forgery (C^*, m^*), from the sender T to the receiver R, the solution of CDH i.e g^{ab} can be found as follows.

1. Get the private key $sk_R = \langle sk_{R_1}, sk_{R_2} \rangle$ of the receiver R from the *KeyGen* list.
2. From the C_1, pk_T and pk_R obtained from the forgery, search the list $\mathbb{L}_2$ for the corresponding entry and get the corresponding t_i and $H = h_{2i}$ values.
3. Retrieve $h_1 = H_1(C_1, C_2, C_3, {C_1}^{sk_{R_1}}, {C_1}^{sk_{R_2}}, pk_{S1}, pk_{S2}, pk_{R1}, pk_{R2})$ from the list $\mathbb{L}_1$.
4. After obtaining the corresponding t_i, H and h_1 values, $\mathscr{C}$ computes $g^{ab} = \left[\frac{H^{C_4}}{C_2{}^{h_1}}\right]^{\frac{1}{t_i}}$

Correctness: We have to show that $\left[\frac{H^{C_4}}{C_2{}^{h_1}}\right]^{\frac{1}{t_i}}$ is indeed g^{ab}. The correctness follows:

$$\left[\frac{H^{C_4}}{C_2{}^{h_1}}\right]^{\frac{1}{t_i}} = \left[\frac{\left(H^{sk_{S_1}}.H^{r.h_1}\right)}{H^{r h_1}}\right]^{\frac{1}{t1}} = \left[H^{sk_{S_1}}\right]^{\frac{1}{t_i}} = \left[\left(g^{bt_i}\right)^a\right]^{\frac{1}{t_i}} = g^{ab}$$

Analysis:

If an sUF-iCMA adversary $\mathscr{A}_{II}$ has an advantage ε_2 against the sUF-iCMA security of the proposed scheme, then the probability of solving the CDH is

$$\varepsilon'' \geq \varepsilon_2$$

6 Conclusion

In this paper, we have proposed a PKI signcryption scheme secure against insider attacks without bilinear pairing offering a tight security reduction to CDH problem. While Zheng's[1] and Baek's[2] schemes are loosely reduced to a weaker problem (forking lemma based reduction to DDH, GDH, GDL), our scheme is tightly reduced to a harder problem (CDH). Moreover, in our security model we have provided the proof in multi-user setting.

References

[1] Y. Zheng, "Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption)," in *Proc. of the 17th Annual International Cryptology Conference (CRYPTO '97), Santa Barbara, California, USA, LNCS*, vol. 1294. Springer-Verlag, August 1997, pp. 165–179.

[2] J. Baek, R. Steinfeld, and Y. Zheng, "Formal proofs for the security of signcryption," *Journal of Cryptology*, vol. 20, no. 2, pp. 203–235, February 2007.

[3] F. Bao and R. H. Deng, "A signcryption scheme with signature directly verifiable by public key," in *Proc. of the 1st International Workshop on Practice and Theory in Public Key Cryptography (PKC'98), Yokohama, Japan, LNCS*, vol. 1431. Springer-Verlag, February 1998, pp. 55–59.

[4] J. H. An, Y. Dodis, and T. Rabin, "On the security of joint signature and encryption," in *Proc. of the 21th International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam (EUROCRYPT'02), The Netherlands, LNCS*. Springer-Verlag, April-May 2002, pp. 83–107.

[5] A. W. Dent, "Hybrid signcryption schemes with insider security," in *Proc. of the 10th Australasian Conference (ACISP'05), Brisbane, Australia, LNCS*, vol. 3574. Springer-Verlag, July 2005, pp. 253–266.

[6] ——, "Hybrid signcryption schemes with outsider security," in *Proc. of the 8th International Conference (ISC'05), Singapore, LNCS*, vol. 3650. Springer-Verlag, September 2005, pp. 203–217.

[7] B. Libert and J.-J. Quisquater, "Efficient signcryption with key privacy from gap diffie-hellman groups," in *Proc. of the 7th International Workshop on Theory and Practice in Public Key Cryptography (PKC'04), Singapore, LNCS*, vol. 2947. Springer-Verlag, March 2004, pp. 187–200.

[8] G. Yang, D. S. Wong, and X. Deng, "Analysis and improvement of a signcryption scheme with key privacy," in *Proc. of the 8th International Conference (ISC'05), Singapore, LNCS*, vol. 3650. Springer-Verlag, September 2005, pp. 218–232.

[9] C. H. Tan, "Analysis of improved signcryption scheme with key privacy," *Information Processing Letters*, vol. 99, no. 4, pp. 135–138, August 2006.

[10] ——, "Forgery of provable secure short signcryption scheme," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 90-A, no. 9, pp. 1879–1880, September 2007.

[11] E.-J. Goh, S. Jarecki, J. Katz, and N. Wang, "Efficient signature schemes with tight reductions to the diffie-hellman problems," *Journal of Cryptology*, vol. 20, no. 4, pp. 493–514, October 2007.

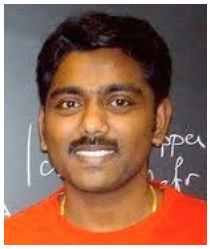
[12] D. Freeman, M. Scott, and E. Teske, "A taxonomy of pairing-friendly elliptic curves," *Journal of Cryptology*, vol. 23, no. 2, pp. 224–280, Aprill 2010.

[13] Y. Seurin, "On the exact security of schnorr-type signatures in the random oracle model," in *Proc. of the 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'12) Cambridge, UK, LNCS*, vol. 7237. Springer-Verlag, April 2012, pp. 554–571.

[14] C. K. Li, G. Yang, D. S. Wong, X. Deng, and S. S. M. Chow, "An efficient signcryption scheme with key privacy," *Journal of Computer Security*, vol. 18, no. 3, pp. 451–473, August 2010.

[15] C. Ma, "Efficient short signcryption scheme with public verifiability," in *Proc. of the 2nd SKLOIS conference on Information Security and Cryptology (Inscrypt'06), Beijing, China, LNCS*, vol. 4318. Springer-Verlag, November-December 2006, pp. 118–129.

[16] J. Fan, Y. Zheng, and X. Tang, "A single key pair is adequate for the zheng signcryption," in *Proc. of the 16th Australasian Conference (ACISP'11), Melbourne, Australia, LNCS*, vol. 6812. Springer-Verlag, July 2011, pp. 371–388.



S.Sree Vivek is a PhD scholar in Indian Institute of Technology - Madras, Chennai, India. He is working under the guidance of Prof C.Pandu Rangan. His areas of interest are design and analysis of Identity Based Cryptosystem, Cryptanalysis of cryptosystem, Key Agreement and works on signcryption schemes. His thesis research topic is studies on some encryption, signature and signcryption schemes.

S.Sharmila Deva Selvi is a PhD scholar in Indian Institute of Technology - Madras, Chennai, India. She is working under the guidance of Prof C.Pandu Rangan. Her areas of interest are Provable Security of Public Key Cryptosystem, Cryptanalysis of Identity Based and Certificateless cryptosystem and her works are mostly on the provable security of various flavors of signcryption schemes.

Salini Selvaraj Kowsalya was working in TCS lab with Prof.Pandurangan on ”Provabe Security” during her undergraduate degree at College of Engineering, Guindy, Anna University. Currently she is doing her Masters in Computer and Information Sciences at University of Wisconsin, Madison. She is working with Prof.Barton Miller in “Paradyn” research group and her present research is on “Analysis of Software Security using Dynamic Instrumentation”.

C.Pandu Rangan is a Professor in the department of computer science and engineering of Indian Institute of Technology - Madras, Chennai, India. He heads the Theoretical Computer Science Lab in IIT Madras. His areas of interest are in theoretical computer science mainly focusing on Cryptography, Algorithms and Data Structures, Game Theory, Graph Theory and Distributed Computing.