Signcryption with Proxy Re-encryption

Chandrasekar S., Ambika K., Pandu Rangan C.

Department of Computer Science and Engineering,
Indian Institute of Technology Madras
sekarnet@gmail.com, ambikakvarma@gmail.com, prangan55@gmail.com

Abstract

Confidentiality and authenticity are two of the most fundamental problems in cryptography. Many applications require both confidentiality and authenticity, and hence an efficient way to get both together was very desirable. In 1997, Zheng proposed the notion of "signcryption", a single primitive which provides both confidentiality and authenticity in a way that's more efficient than signing and encrypting separately. Proxy re-encryption is a primitive that allows a semi-trusted entity called the "proxy" to convert ciphertexts addressed to a "delegator" to those that can be decrypted by a "delegatee", by using some special information given by the delegator, called the "rekey". In this work, we propose the notion of signcryption with proxy re-encryption (SCPRE), and motivate the same. We define security models for SCPRE, and also propose a concrete unidirectional, non-interactive identity-based SCPRE construction. We also provide complete proofs of security for the scheme in the security models defined. We finally provide directions for further research in this area.

1 Introduction

Cryptography has found various applications in email, e-commerce, secure web protocols, etc. Most of these applications demand both confidentiality and authenticity to be provided. So an efficient way of providing both together was desired. It was achieved in 1997, when Zheng[4] proposed a primitive called signcryption, which provides confidentiality and authenticity in a way that's more efficient than combining any encryption and signature schemes. Later several signcryption schemes have been published. The first identity-based signcryption scheme was by Malone-Lee[5] in 2002. Later in 2003, Boyen[6] proposed the first provably secure id-based scheme. In 2005, a more efficient scheme was provided by Chen and Malone-Lee[7]. There have also been other signcryption schemes, and some are variants of signcryption, like those in [8].

Proxy re-encryption is a concept introduced by Blaze et al[9] in 1998, that allows a semi-trusted entity called the "proxy" to convert ciphertexts addressed to an entity B called the "delegator", to another entity C called the "delegatee", while maintaining that the proxy cannot learn anything about the underlying plaintext, and C cannot learn anything about the underlying plaintext without co-operation from the proxy. B does this delegation by providing a special piece of information, called the "rekey", to the proxy. Proxy re-encryption has found various applications like secure email forwarding, etc. But the scheme by Blaze et al[9] was inherently bidirectional – allowed only two-way delegations. Also collusion between proxy and the delegator or delegatee compromised the other entity's secret key. These problems were first fully addressed by Ateniese et al [11]. Later, several proxy re-encryption schemes were proposed, including the first CCA-secure scheme by Canetti et al [12], the first identity-based scheme by Matsuo [13], and the first CCA-secure id-based scheme by Green and Ateniese [14], and the first RCCA-secure and collusion resistant scheme secure in the standard model by Libert and Vergnaud [16].

1.1 Signcryption with Proxy Re-encryption (SCPRE)

We introduce the notion of signcryption with proxy re-encryption (SCPRE). Thus an SCPRE scheme provides confidentiality, authenticity, and proxy re-encryption capabilities in a very efficient way. Such a primitive can have several applications, including:

- *Authentic email forwarding:* Email is an ideal candidate for applying signcryption. A natural application of SCPRE here is to allow signcrypted messages (emails) to be redirected to a delegated person when the actual receiver is unavailable.
- *Secure and authentic distributed storage:* One of the well-known applications of proxy re-encryption[11] can be extended using SCPRE, when the authenticity of the content stored is desirable.

1.2 Contributions of this work

Our contributions presented in this work, can be summarized as:

- Introduction of the notion of signcryption with proxy re-encryption (SCPRE)
- Definition of security models for the confidentiality and non-repudiation of SCPRE, which are slightly weaker than a natural adaptation of the chosen ciphertext attack (CCA2) and the chosen message attack (CMA) models, respectively.
- A concrete construction of an efficient unidirectional, non-interactive, identity-based SCPRE are provided. The scheme is non-transitive and key optimal.
- The confidentiality and non-repudiation of the scheme are formally proved, in the security models defined.

2 The Scheme

2.1 Definition of SCPRE

An SCPRE scheme consists of the following algorithms:

1. Setup: The algorithm accepts a security parameter l, and outputs a master secret key s.
2. Extract: The algorithm accepts an identity ID_u and outputs the secret key S_u
3. Extract-rekey: It accepts two ID_1 and ID_2 and outputs the rekey from ID_1 to ID_2.
4. Signcrypt: It accepts message m, and two identities ID_1 and ID_2 and outputs the signcryption on m from ID_1 to ID_2.
5. De-signcrypt: It accepts a signcryption ϕ and an identity ID_r and outputs the de-signcryption of ϕ by ID_r.
6. Re-encrypt: It accepts a signcryption ϕ, and an identity ID_d and outputs the re-encrypted (second level) signcryption $\phi'$of ϕ to ID_d.

7. De-re-encrypt: It accepts a second-level signcryption ϕ' and ID_d and outputs the de-signcryption of ϕ' by ID_d.

2.2 The Scheme

Our SCPRE scheme is derived from the identity-based signcryption scheme proposed by Chen et al[7]. Our scheme is presented below.

Setup

Let l be the security parameter of the system. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two prime ordered groups of order $q = \Theta(2^l)$, where $\mathbb{G}_1$ is represented additively, and $\mathbb{G}_2$ is represented multiplicatively. Let P be a generator of $\mathbb{G}_1$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear pairing. We assume that the BCDH assumption holds in $< e, \mathbb{G}_1, \mathbb{G}_2 >$.

We use four hash functions $H_0, H_1, H_2,$ and H_3 in our scheme, where
$H_0 : \{0,1\}^* \to \mathbb{G}_1,$
$H_1 : \mathbb{G}_1 \times \{0,1\}^n \to \mathbb{Z}_q^*,$
$H_2 : \mathbb{G}_2 \to \{0,1\}^{n+t}$
$H_3 : \mathbb{G}_1 \times \{0,1\}^* \to \mathbb{G}_1.$

Here n is the number of bits in the message, and t is the number of bits used to represent an element in $\mathbb{G}_1$.

The private key generator (PKG) chooses the master secret key $s \in_R \mathbb{Z}_q^*$ and sets the master public key $P_{pub} = sP$.

The public parameters published are $\mathsf{param} =< e, \mathbb{G}_1, \mathbb{G}_2, n, q, P, P_{pub}, H_0, H_1, H_2 >$

Each user u has his identity ID_u as his public key. He gets two secret keys S_u and $S_{u||delegatee}$ by providing ID_u and $ID_u||$"delegatee" to the following Extract algorithm, respectively.

Extract(ID_u)
The PKG computes the secret key as $S_u = s \cdot H_0(ID_u)$, where $H_0(ID_u)$ is generally denoted as Q_u

Signcrypt(m, S_A, ID_B)
To encrypt a message m to B, user A does the following steps.

1. Choose $r \in_R \mathbb{Z}_q^*$
2. Compute $X = rQ_A$ and $h = H_1(X||m)$
3. Compute the signature $Z = (r + h)S_A$
4. Choose $k \in_R \mathbb{G}_2$
5. Compute $w = e(S_A, Q_B)^r$ and set $\lambda = w \cdot k$
6. $y = H_2(k) \oplus (m||Z)$
7. The signcryption is $\phi =< X, y, \lambda, ID_A >$.

De-signcrypt($\phi =< X, y, \lambda, ID_A >, S_B$)

The receiver B, upon receiving the signcryption ϕ, does the following.

1. $w = e(X, S_B)$
2. Compute $k = \lambda \cdot w^{-1}$
3. Recover $m||Z = y \oplus H_2(k)$
4. $h_1 = H_1(X||m)$
5. If $e(Z, P) = e(P_{pub}, X + h_1 Q_A)$ then $< m, (X, Z), ID_A >$ is output as the message and signature. Otherwise, $\perp$ is output.

Rekey-Extract(S_B, ID_C)
B sends $rk_{B \rightarrow C} =< -S_B + H_3(e(S_B, Q_{C||delegatee})) >$ to proxy.

Re-encrypt($\phi =< X, y, \lambda, ID_A >,\ rk_{B \rightarrow C}, ID_B, ID_C >$)

Proxy computes re-encrypted signcryption $\phi' =< X,\ y,\ \lambda \cdot e(X, rk_{B \rightarrow C}), ID_A, ID_B >$ and sends ϕ' to C.

De-re-encrypt($\phi' =< X, y, \lambda', ID_A, ID_B >, S_{C||delegatee}$)

On receipt of a level 2 signcryption, C decrypts using the following algorithm:

1. $w = e(X, H_3(e(Q_B, S_{C||delegatee})))$
2. Compute $k = \lambda' \cdot w^{-1}$
3. Recover $m||Z = y \oplus H_2(k)$
4. $h_1 = H_1(X||m)$
5. If $e(Z, P) = e(P_{pub}, X + h_1 Q_A)$ then output $< m, (X, Z), ID_A >$; else output $\perp$.

2.3 Correctness

We now show the correctness of our scheme. Let $< X, y, \lambda', ID_A, ID_B >$ be a second-level signcryption addressed to C.

$$\begin{aligned} y \oplus H_2(\lambda' \cdot e(X, H_3(e(Q_B, S_{C||delegatee})))^{-1}) &= y \oplus H_2(k \cdot e(X, H_3(e(S_B, Q_{C||delegatee}))) \cdot \\ &\quad e(X, H_3(e(S_B, Q_{C||delegatee})))^{-1}) \\ &= y \oplus H_2(k) \\ &= m||Z \end{aligned}$$

This shows that the signature $< m, (X, Z), ID_A >$ is recovered correctly, which implies the correctness of the scheme (due to correctness of the underlying signature [3]).

3 Security Models

We define the security models for confidentiality and non-repudiation. It can be seen that for most applications unforgeability of signcryptions is not necessary and non-repudiation would suffice.

3.1 Confidentiality

The security model we define is slightly weaker than the natural adaptation of the adaptive chosen ciphertext attack model, called the IND-SCPRE-CCA2. It is a given-ID attack model.

At the start of the game, the adversary is given a challenge identity ID_{B^*}, which is the identity of the user whose confidentiality the adversary is challenged to break.

Phase 1:

In this phase, the adversary is given access to the following oracles:

- Random oracles H_0, H_1, H_2, H_3
- Extract (ID): returns the secret key of the identity ID
- Extract-rekey (ID_1, ID_2): returns the rekey from ID_1 to ID_2
- Signcrypt(m,ID_1,ID_2) : returns a signcryption ϕ, on message m from ID_1 to ID_2
- De-signcrypt (ϕ, ID_2): if ϕ is a valid (first level) signcryption addressed to identity ID_2, it returns the corresponding message and signature $< m, (X, Z), ID_1 >$; else it returns $\perp$, indicating rejection.
- Re-encrypt(ϕ, ID_1,ID_2): returns the signcryption ϕ', which is the re-encryption of the signcryption ϕ from ID_1 to ID_2
- De-re-encrypt(ϕ', ID_2): if ϕ' is a valid second level signcryption addressed to ID_2, it returns the corresponding message and signature$< m, (X, Z), ID_1 >$; else it returns $\perp$, indicating rejection.

The adversary can access the above oracles with the following restrictions:

- It is not allowed to query Extract(ID_{B^*})
- It is not allowed to query both Extract-rekey(ID_{B^*}, ID_3) and Extract(ID_3 ||"delegatee"), for any ID_3
- It is not allowed to query both Re-encrypt(ϕ', ID_{B^*}, ID_3) and Extract(ID_3 ||"delegatee"), for any ϕ', ID_3

Challenge Phase

At the end of phase 1, the adversary gives two messages m_0 and m_1 and a sender identity ID_A to the challenger. The challenger now chooses a random bit $b \in_R \{0, 1\}$ and outputs the challenge signcryption as $\phi^* = \mathsf{Signcrypt}(m_b, S_A, ID_{B^*})$.

Phase 2:

As in phase 1, the adversary can make polynomial number of queries to the oracles with the following additional restrictions:

- It is not allowed to query De-signcrypt(ϕ^*, ID_{B^*})
- It is not allowed to query De-re-encrypt(Re-encrypt$(\phi^*, ID_{B^*}, ID_3), ID_3)$ for any ID_3
- It is not allowed to query $rk_{B^* \to C} =$ Extract-rekey (ID_{B^*}, ID_C) and De-re-encrypt$({\phi^*}', ID_C)$, where ${\phi^*}' = \mathsf{Re-encrypt}(\phi^*, rk_{B^* \to C}, ID_{B^*}, ID_C)$.

Finally, the adversary outputs its guess b'. The adversary wins the game iff $b = b'$. We say that the system is IND-SCPRE-CCA2 secure if $Pr[b = b'] \geq \frac{1}{2} + \epsilon$ where ϵ is negligible in the security parameter l. The value ϵ is called the advantage of the adversary. So, the system is IND-SCPRE-CCA2 secure if any PPT adversary has only negligible advantage in this game.

3.2 Non-repudiation

The security model we define is slightly weaker than the natural adaptation of the adaptive chosen message attack model (CMA), called the EUF-SCPRE-CMA. It is a given-ID attack model.

At the start of the game, the adversary is given a challenge identity ID_{A*}. The adversary can query the oracles specified in the confidentiality game polynomial number of times with the following restrictions:

- It is not allowed to query Extract(ID_{A^*})
- It cannot query both Extract-rekey (ID_{A^*}, ID_3) and Extract$(ID_3||$"delegatee"$)$, for any ID_3.

- It cannot query both Re-encrypt(ϕ', ID_{A^*}, ID_3) and Extract$(ID_3||$"delegatee"$)$ for any ϕ', ID_3

Finally, the adversary produces a signature $< m, (X, Z), ID_{A^*} >$ such that Signcrypt(m, ID_{A^*}, ID_3) was never queried, for any ID_3. It succeeds if the signature is valid. So, the system is EUF-SCPRE-CMA secure if the probability that any PPT adversary succeeds in the above game is negligible.

4 Security Results

The following are the security results for our SCPRE scheme:

4.1 Confidentiality

Theorem 1 *If there exists a probabilistic polynomial time adversary that wins in the IND-SCPRE-CCA2 game with advantage ϵ, then there exists a probabilistic polynomial time machine that solves the BCDH problem with probability*

$$\geq \epsilon \cdot \frac{1}{q_2}\left(1 - \left(q_s\left(\frac{q_s + q_1}{q}\right) + \sum_{i=1}^{3}\alpha_i(l)\right)\right)$$

where q_1, q_2 and q_s are the number of queries made by the adversary to H_1, H_2 and Signcrypt oracles, respectively, and $\alpha_i(l)$ are negligible, for all i.

4.2 Non-repudiation

Theorem 2 *If there exists a probabilistic polynomial time adversary that wins in the EUF-SCPRE-CMA game with probability ϵ, then there exists a probabilistic polynomial time machine that breaks Chen et al's scheme's non-repudiation with probability*

$$\geq \epsilon \cdot \left(1 - \left(q_s\left(\frac{q_s + q_1}{q}\right) + \beta_1(l) + \sum_{i=2}^{3}\alpha_i(l)\right)\right)$$

where q_1, q_2 and q_s are the number of queries made by the adversary to H_1, H_2 and Signcrypt oracles, respectively, and $\beta_1(l)$ and $\alpha_i(l)$ are negligible, for all i.

5 Proof of Confidentiality (IND-SCPRE-CCA2)

5.1 Security as a signcryption scheme

We now show a PPT algorithm (simulator) that simulates an environment indistinguishable from the real world (as in the model) to an adversary which has any non-negligible advantage in winning the IND-CCA2 game (specified in [7]) of signcryption scheme, can solve a BCDH problem with non-negligible probability.

5.1.1 Simulation

Setup

The simulator chooses a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ where $(\mathbb{G}_1, +)$ and $(\mathbb{G}_2, .)$ are groups of the same prime order q. We assume that the Bilinear Computational Diffie Hellman (BCDH) assumption holds on $< e, \mathbb{G}_1, \mathbb{G}_2 >$.

Initially the simulator is given an instance of BCDH problem on $(e, \mathbb{G}_1, \mathbb{G}_2)$, (P, aP, bP, cP) as the challenge.

The simulator then sets up the public parameters to the adversary as $Param =< e, \mathbb{G}_1, \mathbb{G}_2, q, P,\ P_{pub} = aP, n, H_0, H_1, H_2 >$ where H_0, H_1, H_2 are defined as random oracles. Here, n is the length of the message, in bits. It also chooses an identity, ID_{B^*}, and presents it to the adversary as the challenge identity.

Oracle simulations

In all simulations, any list L is assumed to be empty at the start of the simulation.

Initially the random oracles H_0, H_1 and H_2 are simulated as shown below.

$\underline{H_0(ID)\text{ oracle:}}$

If $ID = ID_{B^*}$ then output bP.

Else, if there exists entry of the form $< x, ID, Q, S >$ in the list L_0, then output Q. Else, choose $x \in_R \mathbb{Z}_q^*$, set $Q = xP$ and $S = xP_{pub}$. Make an entry $< x, ID, Q, S >$ into L_0 list, and output Q.

$H_1(X||m)$ oracle:
If an entry of the form $< (X||m), h_1 >$ exists in L_1, then output h_1. Else, choose $h_1 \in_R \mathbb{Z}_q^*$, enter $< (X||m), h_1 >$ into L_1, and output h_1.

$H_2(k)$ oracle:
If an entry of the form $< k, h_2 >$ already exists in L_2, then output h_2. Else, choose $h_2 \in_R \{0,1\}^{n+t}$, enter $< k, h_2 >$ into L_2, and output h_2.

The H_0 oracle query of ID sets the secret key of ID. The simulation of the Extract oracle is given below.

Extract(ID) oracle:
If an entry of the form $< x, ID, Q, S >$ exists in L_0, then output S. Else query $H_0(ID)$ to set the secret key of ID; now, fetch the entry of the form $< x, ID, Q, S >$ from L_0 list and output S

Signcrypt(m, ID_1, ID_2) oracle:

Case 1: $ID_1 \neq ID_{B^*}$

1. $S_1 = \text{Extract}(ID_1)$
2. Output $\mathsf{Signcrypt}(m, S_1, ID_2)$

Case 2: $ID_1 = ID_{B^*}$

In steps 1 to 4, a signature on m from ID_1 is created, using the control the simulator has over the random oracles. In steps 5 to 9, the normal encryption routine is carried out as in the algorithm.

1. Choose $r, h \in_R \mathbb{Z}_q^*$.
2. $Q_{B^*} = H_0(ID_{B^*})$
3. Compute $X = rP - hQ_{B^*}$, and set $Z = rP_{pub}$.
4. Add the entry $< (X||m), h >$ to the L_1 list.
5. S_2 =Extract(ID_2)
6. Compute $w = e(X, S_2)$
7. Choose $k \in_R \mathbb{G}_2$, and encrypt it as $\lambda = w.k$

8. Now encrypt the message as $y = H_2(k) \oplus (m||Z)$
9. Output $(X, y, \lambda, ID_{B^*})$.

De-signcrypt($\phi = (X, y, \lambda, ID_1), ID_2$) oracle:

Case 1: $ID_2 \neq ID_{B^*}$

1. $S_1 = \text{Extract}(ID_1)$
2. Output $\mathsf{De-signcrypt}(\phi, S_2)$

Case 2: $ID_2 = ID_{B^*}$

With very high probability, the adversary could have computed a valid signcryption only by querying the appropriate k to the H_2 oracle. Thus, the simulator does the following.

For each entry $< k, h_2 >$ in L_2 list do:

1. Compute $(m||Z) = y \oplus h_2$
2. Compute $Q_1 = H_0(ID_1)$, and $h_1 = H_1(X||m)$
3. Validate the sign by checking if $e(Z, P) = e(P_{pub}, X + h_1Q_1)$. If so, proceed to the next step. If not, go to next entry in L_2.
4. Validate the correctness of k by checking if $\lambda = k \cdot e(Z - h_1S_1, bP)$. If so, output $< m, (X, Z), ID_1 >$as the message and signature. If not, go to the next entry in L_2.
5. If no message-signature was returned till this step, output $\perp$.

5.1.2 Reduction

In phase 1, the adversary queries the oracles with the restrictions specified in the security model.

At the end of phase 1, the adversary outputs two messages, m_0, m_1 and a sender identity ID_A. The simulator does the following: set $X^* = cP$, choose $y^* \in_R \{0,1\}^{n+t}$ and $\lambda^* \in_R \mathbb{G}_2$, and finally output $< X^*, y^*, \lambda^*, ID_A >$ as the challenge to the adversary.

In phase 2, the adversary queries the oracles, again with the restrictions specified in the security model. At the end of phase 2, the adversary outputs its guess b' as mentioned in the game, which the simulator discards.

In subsection 5.1.3, we prove that its infeasible for any adversary to distinguish our oracle simulations from the real world (the model) with any non-negligible probability. Hence, any adversary that has advantage ϵ, should do either of these: distinguish that the challenge ciphertext is invalid, or attempt to win the game by computing the bit, with ϵ probability.

It can be seen that the challenger ciphertext is valid as long as $H_2(k)$ where $k = \lambda^*/e(X^*, S_{B^*})$ is set properly. Also, y^* can be a signcryption of m_0 or m_1 with equal probability, and so the adversary has to know the value of $H_2(k)$ to get any information about m_b. Thus, in either case the adversary has to query $H_2(k)$; but, then the simulator can solve the BCDH instance with probability $1/q_2$, by choosing a random element x from L_2 list and outputting λ^*/x, thereby showing a contradiction.

5.1.3 Analysis

We analyze the simulation of each oracle and show that the simulation is indistinguishable from real world, for any adversary.

Random oracles' and Extract oracle's indistinguishability

It can be easily seen that the random oracles H_0,H_1,H_2 and Extract oracle are simulated as in the real world.

Signcrypt oracle's indistinguishability

The signcrypt oracle produces valid signcryptions, except in Step 4 of Case 2 when an entry $< (X||m), h >$ already exists in the L_1 list and therefore an error happens. The probability that this bad event happens in any of the q_s signcrypt queries is $\leq q_s \left(\frac{q_s+q_1}{q}\right)$, as X is a random element in $\mathbb{G}_1$.

De-signcrypt oracle's indistinguishability

It can be observed that the De-signcrypt oracle is simulated correctly in all cases, except for the case when the adversary has produced a *valid* signcryption without making the query $H_2(k)$, but the De-signcrypt oracle rejects it and returns $\perp$.

But as H_2 is a random oracle, and because only a negligible fraction of the signature space has valid signatures, the probability that the adversary can create a valid signcryption without querying $H_2(k)$ is negligible (in l). Now, the probability that this bad event happens in at least one of the q_d de-signcrypt queries that the adversary makes, is also negligible, say, $\alpha_1(l)$.

Thus, the probability that the adversary does *not* distinguish any of the oracles' simulations from the real world is

$$\geq 1 - \left(q_s \left(\frac{q_s + q_1}{q} \right) + \alpha_1(l) \right)$$

which can be seen as $1 - \nu(l)$, where $\nu(l)$ is negligible in l.

5.2 Proof: Part 1

We prove that the security of our system as a signcryption scheme in the IND-CCA2 model under given-ID attacks by probabilistic polynomial time adversaries, is maintained even in the presence of Extract-rekey and De-re-encrypt oracles. The security model is same as that of the IND-CCA2 model (of course, here the adversary is given the Extract-rekey and De-re-encrypt oracles also), except that in Phase 1 there is one additional restriction:

- It cannot query both Extract-rekey (ID_{B^*}, ID_3) and Extract$(ID_3||$"delegatee"$)$, for any ID_3

and in Phase 2 there are two additional restrictions:

- It cannot query both Extract-rekey (ID_{B^*}, ID_3) and Extract$(ID_3||$"delegatee"$)$, for any ID_3
- It cannot query $rk_{B^* \to C}$ = Extract-rekey (ID_{B^*}, ID_C) and De-re-encrypt$(\phi^{*'}, ID_C)$, where $\phi^{*'} = \mathsf{Re-encrypt}(\phi^*, rk_{B^* \to C}, ID_{B^*}, ID_C)$, for any ID_C

5.2.1 Simulation and Reduction

The simulator interacts with the IND-CCA2 challenger of the signcryption scheme. We denote each oracle $\mathcal{O}$ provided by the challenger as $\mathcal{O}'$.

Setup

The simulator is given the public parameters, $Param = < e, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, n,\ H'_0, H'_1, H'_2 >$ where H'_0, H'_1, H'_2 are defined as random oracles, by the challenger. Here, n is the length of the message, and $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ where $\mathbb{G}_1$ and $\mathbb{G}_2$ are groups of same prime order q, is a bilinear pairing, where the BCDH assumption holds. The simulator also gets an identity, ID_{B^*}, as its challenge identity.

The simulator now sets up the public parameters for the adversary as $Param = < e, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, n, H_0, H_1, H_2, H_3 >$, where H_3 is a random oracle, and also gives the same ID_{B^*} as its challenge identity.

Oracle Simulations

Queries to the $H_0(ID)$, $H_1(X||m)$, $H_2(k)$, Extract(ID), Signcrypt(m, ID_1, ID_2), and De-signcrypt$(\phi,\ ID_2)$ oracles are forwarded to the corresponding challenger's oracles, and the responses from the challenger are forwarded back to the adversary.

$\underline{H_3(j)}$ oracle:
If an entry of the form $< j, h_3 >$ already exists in L_3, then output h_3. Else, choose $h_3 \in_R \mathbb{G}_1$, enter $< j, h_3 >$ into L_3 and output h_3.

Extract-rekey $(ID_1,\ ID_2)$

Case 1: $ID_1 = ID_{B^*}$

- If there exists an entry of the form $< ID_1, ID_2, R >$ in the L_r list, then return R,
- Else choose $R \in_R \mathbb{G}_1$, make an entry $< ID_1, ID_2, R >$ to the L_r list, and return R

Case 2: $ID_1 \neq ID_{B^*}$

- If there exists an entry of the form $< ID_1, ID_2, R >$ in the L_r list then return R

- Else compute S_1= Extract(ID_1) and $h = H_3(e(S_1, Q_{2||delegatee}))$; then enter $< ID_1, ID_2, -S_1 + h >$ in the L_r list and return $(-S_1 + h)$.

<u>De-re-encrypt($\phi' =< X, y, \lambda', ID_1, ID_2 >, ID_3$) oracle:</u>

Case 1: $ID_2 \neq ID_{B^*}$

1. $S_{3||delegatee}$= Extract(ID_3||"delegatee")
2. Return $\mathsf{De-re-encrypt}(\phi', S_{3||delegatee})$

Case 2: $ID_2 = ID_{B^*}$

1. If there exists an entry $< ID_{B^*}, ID_3, R >$ from L_r list
   (a) Compute $\lambda = \lambda'.e(X, -R)$
   (b) If De-signcrypt($\phi =< X, y, \lambda, ID_1 >$, ID_{B^*}) returns a valid $< m, (X, Z), ID_1 >$ then forward this to the adversary
2. $S_{3||delegatee}$= Extract(ID_3||"delegatee")
3. Return $\mathsf{De-re-encrypt}(\phi', S_{3||delegatee})$

Reduction

In phase 1, the adversary can query the above oracles any polynomial number of times, but with the restrictions specified in the security model.

At the end of phase 1, adversary outputs two messages, m_0, m_1 and a sender identity ID_A. The simulator forwards this to the challenger and gets the challenge ciphertext ϕ^* addressed to ID_{B^*}. It then forwards this to adversary as its challenge ciphertext.

In phase 2, the adversary can query the oracles any polynomial number of times, again with the restrictions specified in the security model.

Finally, at the end of phase 2, the adversary outputs its guess b' as mentioned in the game, which is sent as the simulator's guess to the challenger.

5.2.2 Analysis

We show that the simulated world is indistinguishable from the real world, for any adversary.

It can be seen that all the oracles and the challenge, except the H_3, Extract-rekey and De-re-encrypt oracles are simulated by forwarding to the IND-CCA2 challenger of section 5.1, and hence are indistinguishable from the real world, by the IND-CCA2 proof. It can also be seen that the random oracle H_3 is simulated as in the real world.

De-re-encrypt oracle's indistinguishability

We now show that the de-re-encrypt oracle simulation is indistinguishable from the real world oracle.

When $ID_2 \neq ID_{B^*}$, it behaves as in the real world. When $ID_2 = ID_{B^*}$, if the second level signcryption was created using the random rekey, simulator will decrypt it properly. If the adversary has not queried the rekey and produced a valid second level signcryption, then step 2 will decrypt it properly. If the adversary has queried the rekey and produced a valid second level signcryption, then with very high probability the first step will fail, and the second step will decrypt it properly. It can also be seen that any other signcryption will be handled in the same way as the real world de-re-encrypt oracle. Hence, the de-re-encryption oracle is simulated as in the real world, except for a negligible probability $\alpha_2(l)$.

Extract-rekey indistinguishability

We show that the extract-rekey oracle is indistinguishable from the real world oracle, by simulating an environment that is indistinguishable to the environment simulated in our simulation, and then showing that the extract-rekey oracle is indistinguishable in this new simulation. In our simulation we assume that the adversary distinguishes that the rekey from, ID_{B^*} to say, ID_C, is invalid. Note that this proof will hold for any ID_C.

Setup

The simulator chooses a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ where $\mathbb{G}_1$ and $\mathbb{G}_2$ are groups of same prime order q. We make the Bilinear Computational Diffie Hellman (BCDH) assumption in $< e, \mathbb{G}_1, \mathbb{G}_2 >$.

Initially the simulator is given (P, aP, bP, cP). As before, it sets up the public parameters, $Param =< e, \mathbb{G}_1, \mathbb{G}_2, q, P,\ P_{pub} = aP, n, H_0, H_1, H_2, H_3 >$ where H_0, H_1, H_2 and H_3 are defined as random oracles, and n is the length of the messages.

<u>Oracle simulations</u>

<u>$H_0(ID)$ oracle:</u>

If $ID = ID_B$ then output bP

If $ID = ID_C||$"delegatee" then output cP

Else, if there is entry of the form $< x, ID, Q, S >$ in the L_0, output Q. Else choose an $x \in_R \mathbb{Z}_q^*$, set $Q = xP$ and $S = xP_{pub}$. Make an entry $< x, ID, Q, S >$ into L_0 list, and output Q.

The $H_1(X||m), H_2(k)$, Extract(ID), Signcrypt(m, ID_1, ID_2), and De-signcrypt$(\phi$,$ID_2)$ oracles are simulated in the same way as in the CCA2 proof of our signcryption scheme.

The $H_3(j)$, and Extract-rekey(ID_1,ID_2) oracles are simulated as in part 1 of proof.

<u>De-re-encrypt($\phi' =< X, y, \lambda', ID_1, ID_2 >, ID_3$) oracle:</u>

Case 1: $ID_2 \neq ID_{B^*}$

1. $S_{3||delegatee}$= Extract($ID_3||$"delegatee")
2. Return $\mathsf{De-re-encrypt}(\phi', S_{3||delegatee})$

Case 2: $ID_2 = ID_{B^*}$

1. If there exists an entry $< ID_{B^*}, ID_3, R >$ in the L_r list
    (a) Compute $\lambda = \lambda'.e(X, -R)$
    (b) If De-signcrypt($\phi =< X, y, \lambda, ID_1 >$, ID_{B^*}) returns a valid signature $< m, (X, Z), ID_1 >$ then forward this to the adversary. Else goto step 2.
2. If $ID_3 \neq ID_C$

(a) $S_{3||delegatee}$= Extract($ID_3||$"delegatee")

(b) Return $\mathsf{De-re-encrypt}(\phi', S_{3||delegatee})$

The above oracle's simulation is as in the simulation of part 1 proof except for the case when the adversary creates a valid level 2 signcryption to ID_C with ID_{B^*} as delegator. For the adversary to be able to create such a signcryption, it should have queried $H_3(e(S_{B^*}, Q_{C||delegatee}))$. Then, the simulator can solve BCDH with probability $1/q_3$ by choosing a random entry from the L_3 list, and giving it as the solution.

In phase 1, the adversary can query the above oracles (a polynomial number of times) with the restrictions specified in the security model. At the end of phase 1, the adversary outputs two messages, m_0, m_1 and a sender identity ID_A. The simulator chooses $x \in_R \mathbb{Z}_q^*$, $X^* = xP$, $y^* \in_R \{0,1\}^{n+t}$ and $\lambda^* \in_R \mathbb{G}_2$, and outputs $< X^*, y^*, \lambda^*, ID_A >$.

Analysis

Now, we show that this simulation is indistinguishable from the simulation in part 1 proof. It can be easily seen that all the oracles are simulated as in the original simulation. Now, it can also be easily seen that the challenge is as in the part 1 proof's simulation, since the challenge is formed in exactly the same way as in the part 1 proof's simulation.

Hence, this simulation is indistinguishable from the simulation of part 1 proof.

Reduction

It can be seen from that when the adversary queries the rekey from ID_{B^*} to ID_C the simulator responds with a random $R \in_R \mathbb{G}_1$.This R can be written as $R = -S_{B^*} + T$.

We say that the adversary has distinguished the Extract-rekey oracle if it has distinguished that at least one of the rekeys is invalid. First note that as long as $H_3(e(S_{B^*}, Q_{C||delegatee})) = T$, the rekey is valid. So, what $H_3(e(S_{B^*}, Q_{C||delegatee}))$ is set to, determines whether a rekey is valid or not. Thus, if the adversary determines that one of the rekeys (of course, from B^*, say, to C) is invalid, it could not have done it without ensuring that $H_3(e(S_{B^*}, Q_{C||delegatee}))$ is incorrectly set. Also, we can see that $H_3(e(S_{B^*}, Q_{C||delegatee}))$ is *random and independent* for a rekey from ID_{B^*} to ID_C. Thus, distinguishing the Extract-rekey oracle could have been done in only by querying $H_3(e(S_{B^*}, Q_{C||delegatee}))$ and simulator returns a value $\tau \neq T$. The adversary now checks if $e(\tau - R, P) = e(Q_{B^*}, P_{pub})$, which will not be true, and hence determine that the rekey is invalid. But in that case, the simulator can solve BCDH with $1/q_3$ probability by choosing a random entry in the L_3 list. The probability with which this can happen, because of the BCDH assumption, is also negligible in l.

Thus, the rekey oracle is distinguishable with probability at most $\alpha_3(l)$, where $\alpha_3(l)$ is negligible in l.

Thus, the probability that the adversary does *not* distinguish the simulation from the real world is

$$\geq 1 - \left(q_s \left(\frac{q_s + q_1}{q} \right) + \sum_{i=1}^{3} \alpha_i(l) \right)$$

which can be written as $1 - \nu'(l)$, where $\nu'(l)$is negligible in l.

5.3 Proof: Part 2

Now we prove the complete IND-SCPRE-CCA2 security of our system.

5.3.1 Simulation and Reduction

Setup

The simulator interacts with the challenger of the part 1 game. It is given the public parameters, $Param =< e, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, n, H'_0, H'_1, H'_2, H'_3 >$ where H'_0, H'_1, H'_2, H'_3 are defined as random oracles and n is the length of the message. Here, $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ where $\mathbb{G}_1$ and $\mathbb{G}_2$ are groups of same prime order q, is a bilinear pairing, where the BCDH assumption holds. The simulator also gets an identity, ID_{B^*}, as its challenge identity.

The simulator now sets up the public parameters for the adversary as $Param =< e, \mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, n, H_0, H_1, H_2, H_3 >$, and also gives the same ID_{B^*} as its challenge identity.

Oracle Simulations

Queries to all oracles except the Re-encrypt, are forwarded to the challenger's corresponding oracles, and the responses are forwarded back to the adversary.

Re-encryption(ϕ, ID_2, ID_3) oracle:

Return $\mathsf{Re-encrypt}(\phi$, Extract-rekey(ID_2, ID_3),$ID_2, ID_3)$

Reduction

In phase 1, the adversary can query the oracles with the restrictions specified in the security model, any polynomial number of times.

At the end of phase 1, the adversary outputs two messages, m_0, and m_1, and a sender identity ID_A. The simulator forwards this to the challenger and gets the challenge ciphertext ϕ^*addressed to ID_{B^*}. It then forwards this to adversary as its challenge.

In phase 2, the adversary queries the oracles a polynomial number of times, again with the restrictions specified in the security model. At the end of the phase, the adversary outputs its guess b', which is the simulator sends as its guess, to the challenger.

5.3.2 Analysis

All the random oracles $H_{0,} H_1$, H_2, H_3, the Extract oracle, the Extract-rekey oracle, the Signcrypt oracle, and the De-signcrypt oracle queries are simulated by forwarding them to the challenger and returning the responses. Since the challenger simulates these oracles indistinguishably from real world, our simulation of these oracles are also indistinguishable from real world. Also, it can be seen that the Re-encrypt and De-re-encrypt oracles are simulated as specified in the algorithm and hence the whole simulation is indistinguishable from real world.

Thus, the probability that the adversary does *not* distinguish the simulation from the real world is

$$\geq 1 - \left(q_s \left(\frac{q_s + q_1}{q} \right) + \sum_{i=1}^{3} \alpha_i(l) \right)$$

which can be seen as $1 - \nu'(l)$, where $\nu'(l)$is negligible in l.

Thus, with probability at least $1-\left(q_s\left(\frac{q_s+q_1}{q}\right)+\sum_{i=1}^{3}\alpha_i(l)\right)$, the adversary will not distinguish the oracle simulation. Now, any adversary that does not distinguish the oracle simulations should either distinguish that the challenge ciphertext is invalid, or should attempt to win the game by computing b, with ϵ probability. It can also be noted that our simulator does to its challenger whatever the adversary does to it (both forwarding b, or aborting claiming distinguishability of simulations). Hence, by the structure of the proof, BCDH will be solved in either case, with probability $1/q_2$. Hence, we have the theorem 1 for the confidentiality of our system.

6 Proof of Non-repudiation (EUF-SCPRE-CMA)

6.1 Security as a signcryption scheme

6.1.1 Simulation & Reduction

Setup

The simulator interacts with the challenger in the non-repudiation game of Chen et al's scheme [7]. It initially gets the public parameters $param' =< e, \mathbb{G}_1, \mathbb{G}_2, q, n, P, P_{pub}, H'_0, H'_1, H'_2 >$ from the challenger, and sets the public parameters to the adversary as $param =< e, \mathbb{G}_1, \mathbb{G}_2, q, n, P, P_{pub}, H_0, H_1, H_2 >$.

The simulator then chooses an identity, ID_{A^*}, as the challenge identity and gives it to the adversary.

Oracle simulations

Queries to the $H_0(ID)$, $H_1(X||m)$, and Extract(ID) oracles are forwarded to the challenger's corresponding oracles, and the responses are forwarded back to the adversary.

Of the random oracles, H_2 alone is simulated by the simulator without forwarding to challenger, as shown below.

$H_2(k)$ oracle:
If there exists an entry of the form $< k, h_2 >$ in L_2, then output h_2. Else,

choose $h_2 \in_R \{0,1\}^{n+t}$, enter $< k, h_2 >$ into L_2 and output h_2.

<u>Signcrypt(m, ID_1, ID_2) oracle:</u>

Case 1: $ID_1 \neq ID_{A^*}$

1. S_1= Extract(ID_1)
2. Output Signcrypt(m, S_1, ID_2)

Case 2: $ID_1 = ID_{A^*}$

In steps 1 and 2, a signature on m from ID_1 is created using the oracles provided by the challenger in an appropriate way. Then, in steps 3 to 7, the normal encryption routine is carried out as in the algorithm.

1. $\phi =$ Signcrypt$'(m, ID_1, ID_2)$
2. $< m, (X, Z), ID_{A^*} >=$ De-signcrypt'(ϕ, ID_2)
3. S_2 =Extract(ID_2)
4. Compute $w = e(X, S_2)$
5. Choose $k \in_R \mathbb{Z}_q^*$ and compute $\lambda = k \cdot w$
6. $y = H_2(k) \oplus (m||Z)$
7. Output $< X, y, \lambda, ID_{A^*} >$

<u>De-signcrypt($\phi = (X, y, \lambda, ID_1)$, ID_2) oracle:</u>

Case 1: $ID_2 \neq ID_{A^*}$

1. $S_2 =$ Extract(ID_2)
2. Output $\mathsf{De-signcrypt}(\phi, S_2)$

Case 2: $ID_2 = ID_{A^*}$

With very high probability, the adversary could have computed a valid signcryption only by querying the appropriate k to the H_2 oracle. Thus, the simulator does the following.

For each entry $< k, h_2 >$ in L_2 list do:

1. Compute $(m||Z) = y \oplus h_2$
2. Compute $Q_1 = H_0(ID_1)$, $Q_2 = H_0(ID_2)$ and $h_1 = H_1(X||m)$
3. Validate the sign by checking if $e(Z, P) = e(P_{pub}, X + h_1 Q_1)$. If failed, go to next entry in L_2. Else go to next step.
4. Now, validate the correctness of k by checking if $\lambda = k \cdot e(Z - h_1 S_1, Q_2)$. If failed, go to next entry in L_2. Else, output $< m, (X, Z), ID_1 >$.
5. Finally, if no message was returned, output $\perp$

6.1.2 Reduction

The adversary can query the above oracles any polynomial number of times, with the restrictions as specified in the security model. Finally it produces a signature $< m, (X, Z), ID_{A^*} >$. Now the simulator will produce a forgery for Chen et al scheme's non-repudiation game as follows:

1. Choose an identity $ID_2 \neq ID_{A^*}$; compute $S_2 = \text{Extract}(ID_2)$.
2. Compute $w = e(X, S_2)$.
3. $y = H_2'(w) \oplus (m||Z||ID_{A^*})$.
4. Output $< X, y >, ID_2$ as forgery for Chen et al scheme's

It can be seen that, the simulator's forgery Chen et al's game is valid whenever the adversary's forgery to our non-repudiation game is valid.

6.1.3 Analysis

We analyze the simulation of each oracles and show that the simulation is indistinguishable from the real world.

Random oracles' indistinguishability

It can be easily seen that the random oracles H_0, H_1 and H_2 are simulated indistinguishably from the real world, due to Chen et al's proof.

Extract and Signcrypt oracle's indistinguishability

Extract oracle and signcrypt oracle are simulated indistinguishably from the real world, due to Chen et al's proof. It can be noted that the probability of distinguishing the Signcrypt oracle, in Chen et al's proof and hence our proof, is at most $q_s \left(\frac{q_s+q_1}{q}\right)$.

De-signcrypt oracle's indistinguishability

It can be observed that the de-signcrypt oracle is simulated correctly in all cases, except for the case when the adversary has produced a valid signcryption without querying $H_2(k)$, but the De-signcryption oracle rejects it and returns $\perp$.

But as H_2 is a random oracle and because only a negligible fraction of the signature space has valid signatures, the probability that the adversary can create a valid signcryption without querying $H_2(k)$ is negligible in l. Thus, the probability that this bad event happens in at least one of the q_d de-signcrypt queries, is also negligible, say, $\beta_1(l)$.

Thus, the adversary does not distinguish the simulation from the real world with probability

$$\geq \left(1 - \left(q_s \left(\frac{q_s + q_1}{q}\right) + \beta_1(l)\right)\right)$$

6.2 Proof: Part 1

We prove that the security of our system as a signcryption scheme in the EUF-CMA model under given-ID attacks by probabilistic polynomial time adversaries, is maintained even in the presence of Extract-rekey and De-re-encrypt oracles.

The security model is same as that of the CMA proof (of course, the Extract-rekey and De-re-encrypt oracles are provided additionally), except there is one additional restriction:

- It cannot query both Extract-rekey (ID_{A^*}, ID_3) and Extract$(ID_3||$"delegatee"$)$, for any ID_3.

Finally, the adversary produces a signature $< m, (X, Z), ID_{A^*} >$ such that Signcrypt(m, ID_{A*}, ID_3) was never queried, for any ID_3. The adversary succeeds if the signature is valid.

Simulation and Reduction

Everything is simulated as in the part 1 simulation of confidentiality game with ID_{A^*}instead of ID_{B^*}. Finally, the adversary will produce a forgery, which will be forwarded as the simulator's forgery, to the challenger.

Analysis

It can be seen that the indistinguishability of the Rekey-extract and De-re-encrypt oracles will follow from the proofs given in the part 1 of the confidentiality game. Hence, the probability that the adversary cannot distinguish the simulation from the real world is

$$\geq \left(1 - \left(q_s\left(\frac{q_s + q_1}{q}\right) + \beta_1(l) + \sum_{i=2}^{3} \alpha_i(l)\right)\right)$$

6.3 Proof: Part 2

Now we prove the complete EUF-SCPRE-CMA security of our system.

Simulation and Reduction

Everything is simulated as in the part 2 simulation of confidentiality game with ID_{A^*}instead of ID_{B^*}. Finally, the adversary will produce a forgery, which will be forwarded as the simulator's forgery, to the challenger.

Analysis

Obviously, the probability that the adversary cannot distinguish the simulation from the real world is

$$\geq \left(1 - \left(q_s\left(\frac{q_s + q_1}{q}\right) + \beta_1(l) + \sum_{i=2}^{3} \alpha_i(l)\right)\right)$$

Hence theorem 2 follows.

7 Conclusion

In this work, we have introduced the notion of signcryption with proxy re-encryption (SCPRE). We have also provided the first concrete construction of an SCPRE scheme, which is non-interactive and unidirectional. We have further defined the security models for confidentiality and non-repudiation, and formally prove the security of the system in the models defined.

7.1 Costs

Computation costs

We represent the costs as (no. of: pairings, $\mathbb{G}_1$ multiplications, $\mathbb{G}_2$ exponentiations). The sender of a signcryption requires (1,2,1), the receiver requires (3,1,1), the proxy (1,0,0), and the delegatee (4,2,1).

Communication costs

The communication cost are represented in the the number of bits required. A level 1 signcryption takes $2|\mathbb{G}_1| + |\mathbb{G}_2| + |m| + |ID|$ bits, whereas a level 2 signcryption $2|\mathbb{G}_1| + |\mathbb{G}_2| + |m| + 2|ID|$ bits. Here, $|x|$ is used to represent the number of bits to represent x.

7.2 Future Work

The problems open in the area of SCPRE include:

- An SCPRE scheme that is confidential and non-repudiable, in models that are a natural adaptation of the CCA2 model and the CMA model.
- An SCPRE scheme secure in the standard model.
- A collusion-resistant SCPRE scheme.

References

[1] A.Shamir, "Identity based cryptosystems and signature schemes". In Advances in Cryptology - Crypto 1984, LNCS 196, pp. 47-53, Springer-Verlag, 1984

[2] Boneh, M. Franklin, “Identity-based Encryption from weil Pairing”. In Crypto 2001, LNCS 2139, pp. 213-229, Springer-Verlag 2001

[3] J. C. Cha and J. H. Cheon, “An Identity-based signature from Gap Diffie-Hellman Groups”. In PKC 2003, LNCS 2567, pp. 18-30, Springer-Verlag-2003.

[4] Yuliang Zheng. Digital Signcryption or How to Achieve Cost (Signature & Encryption) $<<$ Cost(Signature) + Cost(Encryption). In advances in Cryptology - Crypto 1997, LNCS 1294, pp 165-179, Springer-Verlag, 1997.

[5] J. Malone-Lee, “Identity-Based Signcryption”. Cryptology ePrint Archive Report 2002/098D.

[6] X. Boyen, “Multipurpose Identity-Based Signcryption: A Swiss Army Knife for Identity-Based Cryptography”. Crypto 2003, LNCS 2729, pp. 383-399, Springer-Verlag, 2003.

[7] L. Chen, J. Malone-Lee, “Improved Identity-Based Signcryption”. Public Key Cryptography (PKC 2005), LNCS 3386, pp. 362-379, Springer-Verlag, 2005.

[8] Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh and Jean-Jacques Quisquater, Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps, Asicrypto’05, LNCS 3788, pp. 515-532, Springer-Verlag, 2005.

[9] Matt Blaze, G. Bleumer, and M. Strauss, “Divertible protocols and atomic proxy cryptography”. In Eurocrypt 1998, LNCS 1403, pp 127-144, 1998.

[10] Yevgeniy Dodis and Anca Ivan, “Proxy cryptography revisited”. In Proceedings of the Tenth Network and Distributed System Security Symposium, February 2003.

[11] Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger, “Improved Proxy Reencryption Schemes with Applications to Secure Distributed Storage”. ACM TISSEC,9(1): pp 1-30,Feb 2006.

[12] Ran Canetti and Susan Hohenberger, “Chosen Ciphertext Secure Proxy Re-encryption”, CCS 2007: Proceedings of the 14th ACM conference on Computer and communications security, pp.185-194, 2007.

[13] Toshihiko Matsuo, “Proxy Re-encryption Systems for Identity-Based Encryption”, Pairing 2007, LNCS 4575, pp. 247-267, Springer-Verlag, 2007.

[14] Matthew Green, Giuseppe Ateniese, “Identity-Based Proxy Re-encryption, Applied Cryptography and Network Security”, LNCS 4521, pp. 288-306, Springer-Verlag, 2007

[15] Clayton D. Smith, “Digital Signcryption”, Masters Thesis, The University of Waterloo, Canada, 2005

[16] B. Libert and D. Vergnaud, “Unidirectional chosen-ciphertext secure proxy re-encryption”, PKC 2008, LNCS 4939, pp. 360–379, 2008.