Efficient Distributed Signcryption Scheme as Group Signcryption*

DongJin Kwak and SangJae Moon

School of Electrical Eng. & Computer Science, Kyungpook National Univ., Korea.
neverdid@m80.knu.ac.kr, sjmoon@knu.ac.kr

Abstract. The existing distributed signcryption is designed for distributing a signcrypted message to a designated group. However, it does not provide confidentiality of sender ID and its extension to a group signcryption has certain weakness. This study solves the weakness and extends to an efficient group signcryption. In addition, the proposed distributed signcryption provides with sender ID confidentiality. The extension to a group signcryption is more efficient in computational load than the existing scheme. Moreover, the group manager doesn't need to keep and generate his group member's secret information. It can enhance the strength of security in the protocol.

Keywords: Signcryption, Group signature, Public-key Cryptography.

1 Introduction

The signcryption [Zhe1] is an asymmetric cryptographic method that can simultaneously provide message confidentiality and unforgeability with a lower computational and communicational overhead based on two shortened versions (SDSS1 and SDSS2) of DSS [DSS]. As a result, various signcryption schemes have been designed with additional properties [BD98,KM02,LM00]. Originally, signcryption could only be verified by a single designated verifier. However, a variant scheme with multiple designated verifiers was proposed by Zheng for the first time [Zhe2]. Thereafter, Mu and Varadharajan proposed the distributed signcryption using distributed encryption [MN99] in [MV00], where any party can "*signcrypt*" a message and distribute it to a designated group, and any member in the receiving group can "*unsigncrypt*" the message. However, this method does not provide confidentiality to the person who signcrypts the message, while the extension for group signcryption involves lots of additional complexities in computational load and some problems as a group signcryption.

Accordingly, the current paper presents a new distributed signcryption scheme that includes sender ID confidentiality and almost the same computa-

* This research has been supported by University IT Research Center Project.

J. Zhou, M. Yung, Y. Han (Eds.): ACNS 2003, LNCS 2846, pp. 403–417, 2003.

tional cost as the existing scheme. Since the existing signcryption scheme requires the sender certificate to verify the signcryption in advance, it cannot provide the confidentiality of sender ID without an additional overhead. In contrast, the proposed scheme avoids this extra overhead, plus non-repudiation by trusted party is offered to open the signcryption in the case of dispute. Furthermore, the proposed scheme is extended to a group signcryption with properties of group signatures and deals with the problems related to the existing extended scheme, which is too complicated to satisfying some requirements, such as anonymity and traceability. Especially, it is serious issue that a group manager generates and keeps his member's secrete information in the existing protocol. It makes a forgery possible by a malicious group manager. Considering these kind of problems, we renovate its extended group signcryption. The proposed extended scheme also has more efficient computational load for both signcryption and unsigncryption than the existing extended scheme.

The remainder of this paper is organized as follows. The next section outlines the original signcryption and the distributed encryption scheme, then Section 3 explains the existing distributed signcryption and its extension. Section 4 presents the new schemes. Section 5 analyzes the security of the proposed schemes and compares their computational cost with those of previous schemes. Some final conclusions are given in Section 6.

2 Preliminaries

This section briefly reviews the original signcryption [Zhe1] and distributed encryption [MN99]. Throughout this paper, p denotes a large prime number, ${Z_p}^*$ a multiplicative group of order q for $q|(p-1)$, and $g \in {Z_p}^*$ a primitive element.

2.1 Signcryption

Signcryption [Zhe1] refers to a cryptographic method that involves the functions of encryption and digital signature simultaneously. The main advantage of the scheme is the savings in computational cost of encryption and signature compared to traditional signature-then-encryption. Signcryption is based on the Digital Signature Standard(DSS) with a minor modification that makes the scheme more computationally efficient. The modified DSS is referred to as SDSS of which there are two versions. The total procedure is as follows. Assume Alice signcrypts a message and Bob unsigncrypts the message. $(x_a, y_a = g^{x_a})$ and $(x_b, y_b = g^{x_b})$ are the private key and public key pairs for Alice and Bob, respectively. Where $hash(\cdot)$ denotes a strong one-way hash function, $hash_k(\cdot)$ a keyed one-way hash function with key k, and $E_k(D_k)$ a symmetric encryption(decryption).

Alice

choose $z \in_R Z_q$
compute $k = y_b^z \bmod p$
split k into k_1 and k_2
compute $r = hash_{k_2}(m)$
$s = z(r + x_a)^{-1} \bmod q$ if SDSS1
$s = z(1 + x_a \cdot r)^{-1} \bmod q$ if SDSS2
$c = E_{k_1}(m)$

$\longrightarrow (c, r, s) \longrightarrow$

Bob

$k = (y_a \cdot g^r)^{s \cdot x_b} \bmod p$ if SDSS1
$k = (y_a^r \cdot g)^{s \cdot x_b} \bmod p$ if SDSS2
split k into k_1 and k_2
compte $m = D_{k_1}(c)$
verify $r? = hash_{k_2}(m)$

2.2 Distributed Encryption

Distributed encryption [MN99] has a single public key together with one or more decryption keys. As such, its range can be extended to group. Consider a group of members with a public key, referred to as the group public key. Each member of the group has a group private key that matches with the group public key. Any member of the group can then decrypt the message using his or her group private key.

Initialization of a group. A group manager that is trusted by the members of the group is assumed. The group manager is responsible for constructing the group public key and updating group members. In order to construct a group including n members, the manager selects a set of integers, $\epsilon_i \in_R Z_q$ for $i = 1, 2, ..., n$ and computes the coefficients $\alpha_0, ..., \alpha_n \in Z_q$ of the following polynomial:

$$f(x) = \prod_{i=1}^{n} (x - \epsilon_i) = \sum_{i=0}^{n} \alpha_i x^i \tag{1}$$

This function has the following property: Define $g \in Z_p^*$ and $g_i \leftarrow g^{\alpha_i} \bmod p$ for i = 0,1,...,n, which produces

$$F(\epsilon_\ell) = \sum_{i=0}^{n} {g_i}^{{\epsilon_\ell}^i} = 1 \bmod p \tag{2}$$

This is because $F(\epsilon_\ell) = g^{f(\epsilon_\ell)}$ and $f(\epsilon_\ell) = 0$ in Z_q, where ϵ_ℓ is a element of the set $\{\epsilon_i\}$. That is an important property of the system. However, there is the weak point if the polynomial $\{g_i\}$ is used as group public key. In that case, an adversary would add another illegal ϵ_i' to the set $\{\epsilon_i\}$ by using the

$\{g_i\}$ polynomial. Thus, to avoid this weakness, the method given in [MN99] is adopted. For the given $\{\alpha_0, \alpha_1, ..., \alpha_n\}$, a new set is defined $\{\alpha'_0, \alpha'_1, ..., \alpha'_n\}$, where $\alpha'_0 = \alpha_0, \alpha'_n = \alpha_n, \alpha'_1 = ... = \alpha'_{n-1} = \sum_{i=1}^{n-1} \alpha_i$. Define $\beta_i \leftarrow g^{\alpha'_i}$and $A_\ell = \sum_{i=1, j=1, i\neq j}^{n-1} \alpha_j \epsilon_\ell^i$, then the property(Equation (2)) still holds:

$$F'(\epsilon_\ell) = g^{-A_\ell} \prod_{i=0}^{n} \beta_i{}^{\epsilon_\ell^i} = g^{-A_\ell} g^{\sum_{i=0}^{n} \alpha'_i \epsilon_\ell^i} = 1 \bmod p \tag{3}$$

In order to construct a group public key, the group manager picks a random number $\gamma \in_R Z_q$, then computes its inverse γ^{-1} and parameters $\rho_\ell \leftarrow -\gamma A_\ell \bmod q$ for member ℓ. The *group public key* is defined as an $n+2$ tuple, $\{\beta_0, ..., \beta_{n+1}\} \leftarrow \{\beta_0, ..., \beta_n, g^{\gamma^{-1}}\}$. The manager keeps γ and all $\{\alpha_i\}$ secret and gives ϵ_ℓ and ρ_ℓ to a group member ℓ who then uses ϵ_ℓ and ρ_ℓ as her *group private key pair*.

Distributed Encryption and Decryption. If Alice wants to send a message m securely to a designated group G, she picks the encryption key, k, computes $w = hash(m)$, and then encrypts m to obtain the ciphertext $c = (c_1, c_2)$ as follows:

$$c_1 \leftarrow \{a_0, ..., a_{n+1}\} \leftarrow \{g^k \beta_0^w, \beta_1^w, ..., \beta_{n+1}^w\}, \qquad c_2 = m g^k \bmod p.$$

Let Bob be the group member and his group private key pair be (ϵ_b, ρ_b). Bob does the following:

$$c'_1 \leftarrow a_0 (\prod_{i=1}^{n} a_i^{\epsilon_b^i}) a_{n+1}^{\rho_b} = g^k (\prod_{i=0}^{n} g^{w \alpha_i \epsilon_b^i}) = g^k g^{w f(\epsilon_b)} = g^k \bmod p$$

After computing c'_1 using his group private key pair (ϵ_b, ρ_b), Bob can decrypt ciphertext $c = (c_1, c_2)$ from c_2/c'_1. Any member who has his group private key pair in G can then decrypt the ciphertext c to obtain m. Once m is obtained, Bob verifies the correctness of the encryption by checking whether $c_1 = \{g^k \beta_0^w, \beta_1^w, ..., \beta_{n+1}^w\}$ using g^k and w.

3 Existing Distributed Signcryption

This section explains the existing distributed signcryption along with its extension, then considers their security and efficiency [MV00]. The group is constructed as described in Section 2 and some of notations are also the same as in Section 2.

3.1 Distributed Signcryption

Assume that Alice is the sender who signcrypts a message m and sends the message to a designated group. Bob is a member of the designated group and he unsigncrypts the message.

The signcryption. Alice does the follows and sends to Bob or the designated group the signcrypted message (c_1, c_2, r, s).

Choose $z \in_R Z_q$ and compute $k = g^z \mathrm{mod} p$
Split k into k_1 and k_2
Compute $r = hash_{k_2}(m)$
Compute $s = z(k \cdot r + x_a)^{-1} \mathrm{mod}\ q$ if SDSS1
$s = z(k + x_a \cdot r)^{-1} \mathrm{mod}\ q$ if SDSS2
Compute $w = hash(m)$
The encrypted message is as follows:

$$c_1 \leftarrow \{a_0, ..., a_n, a_{n+1}\} \leftarrow \begin{cases} \{g^{kr}\beta_0^w, \beta_1^w, ..., \beta_{n+1}^w\} & (SDSS1) \\ \{g^k\beta_0^w, \beta_1^w, ..., \beta_{n+1}^w\} & (SDSS2) \end{cases}$$

$$c_2 = E_{k_1}(m)$$

The unsigncryption. Bob who is one of the designated group members and has his group private key pair (ϵ_b, ρ_b) can unsigncrypt the signcrypted message by discovering the secret session key k as follows:

For SDSS1

$$k \leftarrow (y_a a_0 (\prod_{i=1}^n a_i^{\epsilon_b^i}) a_{n+1}^{\rho_b})^s = (y_a g^{rk} \prod_{i=0}^n g^{w\alpha_i \epsilon_b^i})^s$$
$$= (y_a g^{rk} g^{wf(\epsilon_b)})^s = g^z \mathrm{mod}\ p$$

For SDSS2

$$k \leftarrow (y_a^r a_0 (\prod_{i=1}^n a_i^{\epsilon_b^i}) a_{n+1}^{\rho_b})^s = (y_a^r g^k \prod_{i=0}^n g^{w\alpha_i \epsilon_b^i})^s$$
$$= (y_a^r g^k g^{wf(\epsilon_b)})^s = g^z \mathrm{mod}\ p$$

Then Bob splits k into k_1 and k_2 as agreed earlier and verifies $m? = D_{k_1}(c_2)$

Its extension to a group signcryption. The distributed signcryption can be extended to a group signcryption, which enables a member of group to signcrypt a message on behalf of the group and send it to another member in another group. Consider two designated group G_A and G_B, then assume that Alice belongs to the group G_A and wants to send a signcrypted message m to the group G_B which Bob belongs to. Let's introduce Alice's ϵ_a that is one of his group private key pair satisfying $f(\epsilon_a) = 0 \mod p$ (defined in section 2.2). Bob also has a counterpart key ϵ_b. The group public keys for G_A and G_B are $\{\overline{\beta}_0, \overline{\beta}_1, ..., \overline{\beta}_{n+1}\}$ and $\{\beta_0, \beta_1, ..., \beta_{n+1}\}$, respectively. Both public keys have the same form as those given earlier. In order to signcrypt the message, Alice needs to do the following and keeps (z, k, w) secret:

Choose $z \in_R Z_q$ and compute $k = g^z \bmod p$
Split k into k_1 and k_2
Compute $w = hash(m)$
Compute the signing commitment, $u_j \leftarrow \overline{\beta}_j{}^w$ for $j = 1, ..., n$
Compute $r = hash_{k_2}(m)$
Compute $s_j = z(\epsilon_a^j - ru_j) \bmod q$ for $j = 1, ..., n$

The signcrypted message is as follows:

$c_1 \leftarrow \{a_0, ..., a_n, a_{n+1}\} \leftarrow \{g^{kr}\beta_0^w, \beta_1^w, ..., \beta_{n+1}^w\}$
$c_2 \leftarrow \{\overline{a}_0, u_1, ..., u_n, \overline{a}_{n+1}\} \leftarrow \{g^{z-rk}\overline{\beta}_0{}^w, u_1, ..., u_n, \overline{\beta}_{n+1}{}^{w\rho_a}\}$
$c_3 = E_{k_1}(m)$

Sends the signcrypted tuple $(c_1, c_2, c_3, r, s_1, ..., s_n)$ to G_B.

Then The verification process by Bob belonging to G_B includes session key recovery as in the following step:

Recover session key k

$$\begin{aligned} k &= [a_0(\textstyle\prod_{i=1}^n a_i^{\epsilon_b^i})a_{n+1}^{\rho_b}][\overline{a}_0(\prod_{j=1}^n u_j^{ru_j}\overline{\beta}_j{}^{s_j})\overline{a}_{n+1}] \\ &= [g^{kr}\beta_0^w(\textstyle\prod_{i=1}^n \beta_i^{w\epsilon_b^i})\beta_{n+1}^{\rho_b}][g^{z-rk}\overline{\beta}_0^w(\prod_{j=1}^n \overline{\beta}_j{}^{w\epsilon_a^j})\overline{\beta}_{n+1}^{w\rho_a}] \\ &= [g^{rk}\textstyle\prod_{i=0}^n \beta_i^{w\epsilon_b^i}][g^{z-rk}\prod_{j=0}^n \overline{\beta}_j{}^{w\epsilon_a^j}] \\ &= g^z \bmod p \end{aligned}$$

Then Bob verifies the correctness of a_0 and $\overline{a}_0$ using the recovered session key.

3.2 Consideration

Distributed signcryption and its extension can be useful in a distributed environment where an entity signcrypts a message and a group unsigncrypts the message. However, it can not provide sender ID confidentiality and its extension is unsuitable for a group signcryption. Thus, the current study mainly focuses on dealing with these two problems.

Distributed signcryption. Distributed signcryption cannot provide sender ID confidentiality, as in the unsigncrypting procedure, the verifier cannot verify the signcryption message (c_1, c_2, r, s) without the signer's public key y_a as follows:

For SDSS1	For SDSS2
$k \leftarrow (y_a a_0(\prod_{i=1}^n a_i^{\epsilon_b^i})a_{n+1}^{\rho_b})^s$	$k \leftarrow (y_a^r a_0(\prod_{i=1}^n a_i^{\epsilon_b^i})a_{n+1}^{\rho_b})^s$

In this case, either the sender must give his certificate to the designated group in advance or someone who belongs to the designated group and wants to verify the signcryption must know the sender's public key, y_a based on their own efforts. This, of course, is another overhead in this protocol. Therefore, distributed signcryption cannot be used in applications requiring sender ID confidentiality such as electronic voting scheme and registration step in group signature. In electronic voting, sender information should be hided to other voters except administrators and in group signature, anyone who wants to join a group registers his commitment that represents himself to group manager without revealing his secret information.

Its unsuitable extension to group signcryption. Group signcryption should include the properties of both group signature and signcryption. The following outlines the weakness as regard group signcryption.

First, when initially constructing a group, the group manager computes and creates group private key pairs $\{(\epsilon_i, \rho_i)\}_{i=1}^{n}$ for all the members. In this case, if an adversary successfully attacks the manager, all secret information about the group members will be compromised. For example, assume that a member's group private key $(\epsilon_\ell, \rho_\ell)$ is revealed after an attack on the group manager. The attacker can impersonate the member and forge a signcryption message. As such, this kind of situation is very dangerous in the protocol related to groups. Furthermore, a malicious group manager can forge one of his group member easily. Because the above reasons, most of the group signatures avoid revealing the member's secret information to the manager [AT00,BS01,CS97,LW02].

Second, in unsigncryption, the verifier needs to know which group member sent the signcrypted message in advance, thereby it creates more overhead in the signcryption protocol, where all messages are encrypted by symmetric and asymmetric encryption.

Finally, the overall protocol is too complicated as regards the modular arithmetic to satisfying some requirements such as anonymity and traceability.

4 The Proposed Schemes

This section presents a new distributed signcryption scheme with sender ID confidentiality and extends it to an efficient distributed signcryption as a group signcryption by solving the above mentioned weakness.

4.1 A Distributed Signcryption with Sender ID Confidentiality

The group manager initializes a group, as in Section 2.2, then remainder of the situation is that same as in the existing scheme.

Signcryption. Alice does the following and sends to Bob who belongs to a designated group the message (c_1, c_2). Where, $Cert_a$ is Alice's certificate including

her public key, $x||y$ means the concatenation of x and y and the rest of notations are the same as in Section 3.

Choose $z \in_R Z_q$ and compute $k = g^z \text{mod} p$
Split k into k_1 and k_2
Compute $r = hash_{k_2}(m)$
Compute $s = z(r + x_a)^{-1} \text{mod } q$ if SDSS1
$s = z(1 + x_a \cdot r)^{-1} \text{mod } q$ if SDSS2
Compute $w = hash(m)$
The encrypted message is as follows:
$c_1 \leftarrow \{a_0, ..., a_n, a_{n+1}\} \leftarrow \{k\beta_0^w, \beta_1^w, ..., \beta_{n+1}^w\}$
$c_2 = E_{k_1}(m||r||s||Cert_a)$

Instead of using the session key, k, for computing s, it is used in computing c_1 in the form of $k\beta_0^w$ like ElGamal encryption [ElGa]. This method make it possible to recover the session k without knowing r, s and insert r, s, and $Cert_a$ in c_2 encryption. Therefore, the property of sender ID confidentiality is provided in the proposed scheme.

The unsigncryption. Bob or any one of the designated group members can unsigncrypt the signcrypted message using his (ϵ_b, ρ_b) based on discovering the secret session key k as follows:

$k \leftarrow a_0(\prod_{i=1}^n a_i^{\epsilon_b^i}) a_{n+1}^{\rho_b} = g^z \prod_{i=0}^n g^{w\alpha_i \epsilon_b^i} = g^z g^{wf(\epsilon_b)} = g^z \text{mod } p$
Split k into k_1 and k_2
Decrypt $D_{k_1}(c_2) = m||r||s||Cert_a$
Verify $r? = hash_{k_2}(m)$
$g^z? = (y_a \cdot g^r)^s$ if SDSS1 or $g^z? = (g \cdot y_a^r)^s$ if SDSS2

Instead of using only the session key recovery in unsigncryption, signature verification follows the session key recovery. With this two step, sender ID confidentiality can be provided. It is obvious that the recipient can unsigncrypt the message by following process above. The only way to decrypt is to have one of group member's secret key pair, ϵ_b, along with ρ_b, which gives $f(\epsilon_b) = 0 \text{ mod } q$. Alice has to use the designated group public key, otherwise no one in the group can unsigncrypt the message. Alice also has to use her private key, x_a, to compute s, since his public key is used to verify the signcryption. If s doesn't embed x_a, x_a cannot be removed by using public key y_a in the verification.

4.2 An Efficient Extension to Group Signcryption

Next, the proposed scheme is extended to a efficient group signcryption. There are five procedures involved in group signcryption: setup, join, signcryption, unsigncryption, and tracing or open, which means opening the identity of the

group member who issued the signcryption together with a proof of this fact, enabling a member of one group to signcrypt a message on behalf of the group and send it to another member in another group with anonymity.

Setup. To derive the information related a group, the group manager computes the following values:

- p, q, and g are the same as the above and $h \in Z_p^*$ is newly generated.
- G_A group manager's RSA signature key denotes d_A , verification key e_A, and n_A a large RSA modulus with two random prime factors of approximately equal length. It satisfies $e_A \cdot d_A \equiv 1 \bmod \phi(n_A)$, where $\phi(n_A)$ is Euler phi function.

The manager keeps d_A as his secret signature key and opens (p, q, g, h, n_A, e_A) as the system parameters.

Join. Each entity who wants to join a group generates his own group private key ϵ_ℓ and computes $\tau_\ell (= h^{\epsilon_\ell} \bmod p)$ as *group membership key.* Then he transfers τ_ℓ to the group manager through secure channel and proves to the manager that he knows the discrete logarithm of τ to the base h. ϵ_ℓ should be kept secret by the entity. The group manager doesn't need to keep and generate his group member's all secret information. It can enhance the strength of security. Each group manager generates $v_\ell (= \tau_\ell^{d_A} \bmod n_A)$ as membership certificate like [CS97]. Using RSA signature is simple and efficient for verifying whether τ_ℓ is valid or not. In order to setup a group, the manager computes the coefficients of following polynomial:

$$f(x) = \prod_{i=1}^{n} (x - \tau_i) = \sum_{i=0}^{n} \alpha_i x^i \tag{4}$$

Let $\{\alpha_i'\}$ and $\{\beta_i\}$ be define like Section 2.2, and define $A_\ell = \sum_{i=1, j=1, i \neq j}^{n-1} \alpha_j \tau_\ell^i$, then Equation (4) has the following property:

$$F'(\tau_\ell) = g^{-A_\ell} \prod_{i=0}^{n} \beta_i{}^{\tau_\ell^i} = g^{-A_\ell} g^{\sum_{i=0}^{n} \alpha_i' \tau_\ell^i} = g^{f(\tau_\ell)} = 1 \bmod p \tag{5}$$

In order to make a group public key, the manager picks a random number γ and computes its inverse and ρ_ℓ as in the distributed encryption. The group public key is defined as $\{\beta_0, ..., \beta_{n+1}\} \leftarrow \{\beta_0, ..., \beta_n, g^{\gamma^{-1}}\}$. Then the manager keeps γ, all $\{\alpha_i\}$, and $\{\tau_\ell\}$ secret and gives v_ℓ and ρ_ℓ to group member ℓ.

Signcryption. After the above group construction, consider two designated groups, G_A and G_B , and assume that Alice belongs to G_A and Bob is one of recipients belonging to G_B. In order to signcrypt the message m, Alice needs to do the following using his ϵ_a, τ_a, and v_a :

Choose z and $t \in_R Z_q$ and compute $k = g^z \text{mod} p$
Split k into k_1 and k_2
Compute $r = hash_{k_2}(m)$
Compute $s = z(r + \epsilon_a \cdot t)^{-1} \text{mod } q$ if SDSS1
$s = z(1 + \epsilon_a \cdot r \cdot t)^{-1} \text{mod } q$ if SDSS2
Compute $w = hash(m)$
Compute $\lambda_a = (t^{e_A} \cdot \tau_a \bmod n_A) \text{mod } q$, $\delta_a = g^{\epsilon_a t}$, and $\theta_a = t \cdot v_a \bmod n_A$
The encrypted message is as follows:
$c_1 \leftarrow \{a_0, ..., a_{n+2}, \} \leftarrow \{k\beta_0^{w\tau_a}, \beta_1^{w\tau_a}, ..., \beta_{n+1}^{w\tau_a}, g^{\lambda_a}\}$
$c_2 = E_{k_1}(ID_{G_A}||m||r||s||\delta_a||\theta_a)$

Where, ID_{G_A} is identity of group G_A. It also includes its group manager's public information (n_A, e_A) . The rest notations are the same as in previous section.

Unsigncryption. Bob or any member of G_B can unsigncrypt the signcrypted message using his (τ_b, ρ_b) based on discovering the secret session key k as follows:

$k \leftarrow a_0(\prod_{i=1}^{n} a_i^{\tau_b^i}) a_{n+1}^{\rho_b} = g^z \prod_{i=0}^{n} g^{w\alpha_i \tau_b^i} = g^z g^{wf(\tau_b)} = g^z \text{mod } p$
Split k into k_1 and k_2
Decrypt $D_{k_1}(c_2) = ID_{G_A}||m||r||s||\delta_a||\theta_a$
Compute $\lambda'_a = ({\theta_a}^{e_A} \bmod n_A) \bmod q$
Verify $r? = hash_{k_2}(m)$
$g^z? = (\delta_a \cdot g^r)^s$ if SDSS1 or $g^z? = (g \cdot \delta_a^r)^s$ if SDSS2
$a_{n+2}? = g^{\lambda'_a}$

It is obvious that the recipient can unsigncrypt the signcrypted message by the above process. The only way to decrypt is to have a group membership key, τ_b, along with ρ_b, which gives $f(\tau_b) = 0 \bmod q$. Alice has to use the designated group public key, otherwise no one in the group can unsigncrypt the signcrypted message. Alice also has to use her valid group keys, ϵ_a and τ_a, and membership certificate v_a to compute the signcryption (c_1, c_2), since only valid key and certificate can be accepted in the verification of the signcryption. Even though a group manager may release the group anonymous key, τ_a or τ_b, by accident, no malicious attacker can impersonate any group member without knowing the valid pair, (ϵ_a, v_a) or (ϵ_b, v_b).

Tracing or Open. In case of disputes, Bob forwards the c_1 and $w(= hash(m))$ to group G_A's manager after decrypting c_2 message and knowing the group G_A's identity. Then, only the manager can find the group member, Alice, who issued this signcryption by testing $\{a_i \ ? = (\beta_i^w)^{\tau_\ell}\}_{i=1}^{n+1}$ for all his group members' τ_ℓ in G_A. After this procedure, disputes can be solved by the group manager. But it causes large computational loads for very large group, so there has more rooms for improving it as a future work.

5 Analysis

This section presents an analysis of the security and computational load of the proposed schemes compared with the existing schemes.

5.1 Analysis of the Signcryption with Sender ID Confidentiality

Security. For the signcryption schemes to be secure, following conditions must be satisfied [Zhe1].

- Unforgeability: A dishonest verifier is in the best position to forge signcrypted text because he knows the original message m and corresponding signature r, s. Thus, it is shown that even the dishonest verifier can not succeed to forge the signcrypted text. For successful forging attack, a dishonest verifier must find another message m' where the hash value is r or another valid signcrypted pair, $m'||r'||s'||Cert'_a$. Considering the former, it is impossible for the attacker to find other message m' with the hash value r because the keyed hash function behaves random function. Regarding the later case, even if a dishonest verifier can generate m' and its hash value r', he cannot generate corresponding s' as he doesn't know the secret key of the signer, x_a.

- Non-repudiation: When a kind of dispute occurs, the recipient forwards a decrypted signcryption pair $m||r||s||Cert_a$ to a trusted third party. Then the third party can settle the dispute by verifying the following:

  recover key $k = (y_a \cdot g^r)^s$ if SDSS1 or $k = (g \cdot y_a^r)^s$ if SDSS2
  verify $r? = hash_{k_2}(m)$

- Confidentiality: The whole signcrypted message (c_1, c_2) is encrypted by symmetric and asymmetric encryption. Therefore, it has a stronger or same confidentiality compared with [Zhe1] and [MV00]. Plus, sender ID confidentiality is also satisfied because sender's ID is included in the encrypted message, c_2. Thus, an adversary can not discover any information without decrypting the ciphertext c_2.

Computational cost. The computational cost was considered as regards modular arithmetic, including modular exponentiation, modular multiplication, and modular inverse, then compared with the existing scheme, as shown in Table 1, since most of the computational time is spent on these modular arithmetic. In Table 1, I_q is denoted as the number of inverse in mod q, $M_q(M_p)$ as the number of multiplication in mod $q(p)$, and $E_p(E_q)$ as the number of exponentiation in mod $p(q)$. As result, the proposed scheme had almost the same computational cost compared to the existing scheme, yet the proposed scheme was more efficient in signcryption. However, when the size n increased, the cost was almost the same.

Table 1. Cost comparison between the proposed scheme and the existing scheme.

Cost		Signcryption		Unsigncryption	
		SDSS1	SDSS2	SDSS1	SDSS2
The existing scheme	I_q	1	1	-	-
	M_q	3	2	-	-
	M_p	1	1	$n+2$	$n+2$
	E_q	-	-	$n-1$	$n-1$
	E_p	$n+4$	$n+4$	$n+2$	$n+3$
The proposed scheme	I_q	1	1	-	-
	M_q	1	2	-	-
	M_p	1	1	$n+2$	$n+2$
	E_q	-	-	$n-1$	$n-1$
	E_p	$n+3$	$n+3$	$n+3$	$n+3$

5.2 Analysis of the Extension to a Group Signcryption

Security. For the group signcryption schemes which include properties of signcryption and group signature at the same time to be secure, following conditions must be satisfied [AT00,BS01,CS97,LW02,Zhe1].

- Correctness: This means that the signcryption produced by a group member must be accepted by the unsigncryption, which can be shown by inspection of the protocol.

- Unforgeability: Only valid group members are able to signcrypt a message on behalf of the group, which is similar to distributed signcryption with sender ID confidentiality. $hash_k(\cdot)$, keyed hash function, behaves as a random function, while the group private key, ϵ_a, can not be revealed to anyone, making the protocol unforgeable.

- Anonymity: With a valid decrypted message, identifying the individual who signcrypt the message is computationally hard for anyone but the group manager. As the sender's information is hidden in the form of $\delta_a (= g^{\epsilon_a t})$ and $\theta_a (= t \cdot v_a)$, and t is changed at every session, no information about sender ID is revealed by $ID_{G_A}||m||r||s||\delta_a||\theta_a$ in the proposed scheme.

- Unlinkability: Deciding if two valid decrypted messages ($ID_{G_A}||m||r||s||\delta_a||\theta_a$) and ($ID_{G_A}||\hat{m}||\hat{r}||\hat{s}||\hat{\delta_a}||\hat{\theta_a}$) were computed by the same group member is computationally hard. As for anonymity, the problem of linking two decrypted message reduces to decide whether δ_a and $\hat{\delta_a}$ are released from same member or θ_a and $\hat{\theta_a}$ are released from same member. This is related to discrete logarithm and finding random number changed at every session.

- Exculpability: Neither a group member nor the group manager can signcrypt on behalf of other group members. No one can obtain any information about

a group private key ϵ_i from $\delta_i (= g^{\epsilon_i t})$. Thus, the value ϵ_i is computationally hidden. Moreover, a group manager can not signcrypt on behalf of his group member because computing discrete logarithms is assumed to be unfeasible.

– Traceability: It was shown in above protocol.

– Coalition-resistance: This means that a colluding subset of group members cannot generate a valid signcryption that the group manager is unable to link to one of the colluding group members. Each group manager signs his members' group membership key τ_i by an RSA signature and transfer it, v_i, to each member. As such, no colluding subset can generate a valid correlated ϵ_i, τ_i, and v_i in the signcryption without the help of the right member and the manager.

– Confidentiality: It is the same as the confidentiality of Section 5.1

Computational cost. The computational cost was also considered as regards modular arithmetic, as shown in Table 2. The situations and notations were almost the same as those in Table 1, M_{n_A} was the number of multiplication in mod n_A and E_{n_A} was the number of RSA exponentiation in mod n_A. In our scheme, a minor RSA signature computation was used to avoid an impersonation attack and minor inverse in mod q was used for computing s. As result, the proposed scheme was more efficient than the existing scheme. Especially, with group size n growing, our scheme was more efficient in both signcryption and unsigncryption than the existing one.

Table 2. Cost comparison between the proposed extension and the existing extension.

Cost		Signcryption		Unsigncryption	
		SDSS1	SDSS2	SDSS1	SDSS2
The existing extension	M_q	**2n+3**	**2n+3**	**n**	**n**
	M_p	**1**	**1**	**3n+3**	**3n+3**
	E_q	**n-1**	**n-1**	**n-1**	**n-1**
	E_p	**n+7**	**n+7**	**3n+1**	**3n+1**
The proposed extension	I_q	1	1	-	-
	M_q	**n+4**	**n+5**	**-**	**-**
	M_p	**1**	**1**	**n+2**	**n+2**
	E_q	**-**	**-**	**n-1**	**n-1**
	E_p	**n+5**	**n+5**	**n+4**	**n+4**
	M_{n_A}	2	2	-	-
	E_{n_A}	1	1	1	1

6 Conclusion

The current study proposed a new distributed signcryption including confidentiality of sender ID that does not involve any additional computational cost, plus an extension for efficient group signcryption. When compared with the extension of the existing scheme, the proposed scheme is more efficient in computational cost and provides additional advantages. Especially the group manager doesn't need to keep and generate his member's secret information. It can enhance the strength of security in the protocol. The new scheme has potential applications in electronic commerce and other areas.

References

[AT00] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," in *Advanced in Cryptology – CRYPTO '2000*, LNCS Vol. 1880, pages 255–270, Springer Verlag, 2000.

[BD98] F. Bao and R. H. Deng, "A signcryption scheme with signature directly verifiable by public key,", in *Public Key Cryptography – PKC '98*, LNCS Vol. 1431, pages 55–59, Springer Verlag, 1998.

[BS01] E. Bresson and J. Stern, "Efficient revocation in group signatures," in *Public Key Cryptography – PKC '2001*, LNCS Vol. 1992, pages 190–206, Springer Verlag, 2001.

[CS97] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," in *Advances in Cryptography – CRYPTO '97*, LNCS Vol. 1294, pages 410–424, Springer Verlag, 1997.

[DSS] National Institute of Standards and Technology, "Digital Signature Standard," Federal Information Processing Standards Publication FIPS PUB 186 U.S. Department of Commerce, 1994.

[ElGa] T. ElGamal, "A public-key cryptosystmes and a signature scheme based on discrete logarithms," in *IEEE Transactions on Information Theory*, Vol. IT-31(4), pages 469–472, 1985.

[KM02] D. Kwak, J. Ha, H. Lee, H. Kim, and S. Moon, "A WTLS handshake protocol with user anonymity and forward secrecy,", in *CDMA International Conference – CIC '2002*, LNCS Vol. 2524, pages 219–230, Springer Verlag, 2002.

[LM00] K. Lee and S. Moon, "AKA protocol for mobile communications,", in *Australasian Conference Informations Security and Privacy – ACISP '2000*, LNCS Vol. 1841, pages 400–411, Springer Verlag, 2000.

[LW02] Y. Lyuu and M. Wu, "Convertible Group Undeniable Signatures,", in *International Conference on Information Security and Cryptology – ICISC '2002*, LNCS Vol. 2587, pages 48–61, Springer Verlag, 2002.

[MN99] Y. Mu, V. Varadharajan, and K. Q. Nguyen, "Delegated decryption," in *Cryptography and Coding '99*, LNCS Vol. 1746, pages 258–269, Springer Verlag, 1999.

[MOV] A. J. Manezes, P. C. van Oorshot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.

[MV00] Y. Mu and V. Varadharajan, "Distributed signcryption," in *Advanced in Cryptology – INDOCRYPT '2000 Proceedings*, LNCS Vol. 1977, pages 155–164, Springer Verlag, 2000.

[RSA] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," in *Communications of the ACM*, Vol. 21, No.2, pages 120–126, 1978.

[Zhe1] Y. Zheng, "Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption),"in *Advanced in Cryptology – CRYPTO '97 Proceedings*, LNCS Vol. 1294, pages 165–179, Springer Verlag, 1997.

[Zhe2] Y. Zheng, "Signcryption and its application in efficient public key solutions,", in *Information Security Workshop – ISW '97*, LNCS Vol. 1396, pages 291–312, Springer Verlag, 1997.