A provable secure anonymous proxy signature scheme without random oracles

Rahim Toluee [1,*], Maryam Rajabzadeh Asaar [1], Mahmoud Salmasizadeh [2]

[1]Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran

[2]Electronics Research Institute, Sharif University of Technology, Tehran, Iran

Abstract

In order to protect the proxy signers' privacy, many anonymous proxy signature schemes which are also called proxy ring signatures, have been proposed. Although the provable security in the random oracle model has received a lot of criticism, there is no provable secure anonymous proxy signature scheme without random oracles. In this paper, we propose the first provable secure anonymous proxy signature scheme without random oracles which is the combination of proxy signature and ring signature. For the security analysis, we categorize the adversaries into three types according to different resources they can get and prove in the standard model that, our proposal is anonymous against full key exposure and existential unforgeable against all kinds of adversaries with the computational Diffie–Hellman and the subgroup hiding assumptions in bilinear groups.

Keywords: proxy signature, ring signature, bilinear pairings, provable security, proxy ring signature, anonymous proxy signature;

1 Introduction

The concept of proxy signature was first introduced by Mambo et al. in 1996 [1, 2]. It is useful in cases when an original signer, wishes to delegate his signing rights to the other one, called a proxy signer.

* Corresponding author:

E-mail addresses: rtoluee@ee.sharif.edu (R. Toluee),

asaar@ee.sharif.edu (M.R. Assar),

salmasi@sharif.edu (M. Salmasizadeh).

Proxy signatures can be combined with other special signatures to obtain several variants of proxy signatures such as threshold proxy signatures [3, 4], blind proxy signatures [5, 6], proxy ring signatures [7, 8, 20] and ring proxy signatures [19]. (The proxy ring signature provides the anonymity of the proxy signer while the ring proxy signature provides the anonymity of the original signer.)

The concept of ring signature was first introduced by Rivest et al. [10]. A ring signature provides the anonymity of a signer, which means that the verifier knows the signer is a member of a ring, but he doesn't know exactly who the signer is. There is also no way to revoke the anonymity of the signer. Ring signatures are related to the group signatures which were introduced by Chaum and Heyst [11]. But there are two distinctions between ring signatures and group signatures. First, group signatures have the additional feature that the anonymity of a signer can be revoked by group manager. Second, in group signature schemes, there exists a trusted third party (TTP) or group manager who manages the joining of group members, whereas in ring signature schemes does not exist such trusted party and for n members of a ring, the rest of the $n-1$ members are totally unaware of being involved in the ring. These two properties make ring signatures widely applicable to many cryptographic schemes [12].

Ring signatures can be combined with other special signatures to obtain several variants of ring signatures such as threshold ring signatures [16], Identity-based ring signatures [17] and universal designated verifiable ring signatures [18].

Proxy ring signatures, also called anonymous proxy signatures, are useful in cases when an entity delegates his signing capability to many proxies, called proxy signers group, while it provides anonymity of proxy signers. A choice is using the group signature to solve it (take the group manger as the original entity), but in some applications, unconditional anonymity is necessary. If the proxies hope that nobody (including the original signer) can open their identities, the group signature is not suitable for this situation. So we can use the ring signature to solve this problem and gain the anonymous proxy signature [8, 13, 14, 15] which was first proposed by Zhang et al. in 2003 [20].

Fuchsbauer and Pointcheval, in 2008 proposed a generic construction named "anonymous proxy signatures" [26]. They proposed no concrete scheme and their purpose is different from ours. They defined a general model for consecutive delegations of signing rights with the following properties: "The delegatee actually signing and all intermediate delegators remain anonymous. As for group signatures, in case of misuse, a special authority can open signatures to reveal the chain of delegations and the signer's identity". They gave formal definitions of security and showed them to be satisfiable by constructing an instantiation proven secure under general assumptions in the standard model.

Yu et al., in 2009 proposed an efficient anonymous proxy signature scheme with provable security in the random oracle models [15]. However, provable security in the random oracle model is doubtful when the random oracles are instantiated with hash functions. Hence, in this paper we propose the first provable secure anonymous proxy signature scheme in the standard model without random oracles, based on standard

assumptions. In this way we make use of two signature schemes: the first one is a proxy signature scheme in standard model, which was proposed by Sun et al. in 2011 [21], and the second one is a ring signature scheme without random oracles, which was proposed by Shacham and Waters in 2007[22].

Roadmap: The rest of this paper is organized as follows. Preliminaries and security requirements are given in sections 2 and 3, respectively. Our anonymous proxy signature scheme is proposed in Section 4. In Section 5, security analysis of our proposal is given. Finally, comparison and conclusions are given in Sections 6 and 7, respectively.

2 Preliminaries

2.1 Bilinear Pairing

We make use of bilinear groups of composite order. These were introduced by Boneh et al. [9]. Let n be a composite with factorization $n = pq$. We have:

- G is a multiplicative cyclic group of order n;
- G_p is its cyclic order-p subgroup, and $G_q$is its cyclic order-q subgroup;
- g is a generator of G, while h is a generator of G_q;
- G_T is a multiplicative group of order n;
- $e : G \times G \rightarrow G_T$ is efficiently computable map with the following properties:
  - Bilinear: for all $u, v \in G$ and $a, b \in Z$ we have

$$e(u^a, v^b) = e(u, v)^{ab} \quad (1)$$

  - Non-degenerate: $e(g, g)$ is generator of G_T whenever g is generator of G;

2.2 Computational Diffie-Hellman (CDH) assumption

Given the tuple $(\eta, \eta^a, \eta^b) \epsilon G_p^3$ for random exponents $a, b \epsilon Z_p$, the adversary could solve the CDH problem if he could compute η^{ab}. The CDH assumption states that no probabilistic polynomial-time (PPT) adversary could solve the CDH problem with non-negligible probability.

2.3 Subgroup Hiding (SGH) assumption

As noted in [22], this assumption states that for given w, selected randomly either from G or G_q (with same probabilities), decide whether w is in G_q, is a hard problem.

2.4 Waters signature (WS) in G_p

As noted in [22], the underlying signature scheme is the Waters signature [23]. This signature was adapted for composite order groups by Boyen and Waters [27]. The scheme is as follows.

a. **Setup**

The setup algorithm chooses generators

$$v', v_1, v_2, \dots, v_k \overset{R}{\leftarrow} G_p$$

b. **Key Generation**

η is a generator of G_p. This algorithm picks random exponents $(a, b \overset{R}{\leftarrow} Z_p)$ and sets

$$\eta_1 = \eta^a$$

$$\eta_2 = \eta^b$$

The public key pk is $(\eta_1, \eta_2) \in G_p^2$ and the private key sk is (a, b).

c. **WS Generation**

This algorithm takes as input a message $M \in \{0,1\}^k$ as a bit string $(m_1, m_2, \dots, m_k)$ and then picks a random $r \overset{R}{\leftarrow} Z_p$ and computes

$$\widehat{S_1} \leftarrow \eta^{ab} \left(v' \prod_{j=1}^{k} v_j^{m_j} \right)^r \tag{2}$$

$$\widehat{S_2} \leftarrow \eta^r \tag{3}$$

The WS signature is $\hat{\sigma} = (\widehat{S_1}, \widehat{S_2}) \in G_p^2$.

d. **WS Verification**

A verifier checks that the following equation is satisfied or not

$$e\left(\widehat{S_1}, \eta\right) = e(\widehat{S_2}, v' \prod_{j=1}^{k} v_j^{m_j}) \,.\, e(\eta_1, \eta_2) \tag{4}$$

If it does not hold, the signature will be rejected. Otherwise, it will be accepted.

This signature is secure assuming CDH is hard in G_p, and is used in our reductions.

3 Security Requirements

3.1 Model of anonymous proxy signature schemes

An anonymous proxy signature (APS) scheme consists of the following algorithms.

a. **Setup:** Given the system security parameter, this algorithm outputs system's parameters. For instance, we can assign this part to a trusted third party.

b. **Key Generation:** On input a security parameter, this algorithm generates a personal public-private key pair (pk, sk).

c. **Delegation Generation:** On input the system's parameters, a warrant w and the original signer's private key sk_a, this algorithm generates a delegation signature σ_w on w.

d. **Delegation Verification:** On input the system's parameters, the original signer's public key pk_a and his delegation signature σ_w on the warrant w, this algorithm outputs "accept" if the signature is valid, and "reject" otherwise.

e. **APS Generation:** On input the system's parameters, a message m, the public keys $pk_1, \dots, pk_n$ of the n proxy signers, delegation signature σ_w on w and one proxy signer's secret key, this algorithm generates an APS σ for the message m.

f. **APS Verification:** On input the system's parameters, a message m, an APS σ, the public key pk_a of the original signer, the public keys$pk_1, \dots, pk_n$ of the n proxy signers, the warrant w and the delegation signature σ_w on w this algorithm outputs "accept" if the signature is valid, and "reject" otherwise.

3.2 Anonymity against full key exposure attack in APS schemes

In APS schemes, the anonymity means, informally, that an adversary not be able to find which member of a proxy signers' ring generated a particular signature. Motivated by the work of Bender et al. [24, 25], we provide a formal definition of anonymity against full key exposure attack of an APS scheme as follows.

Given an APS scheme and a PPT adversary, consider the following game:

1. Key pairs $\{(pk_i, sk_i)\}_{i=1}^{l}$ are generated and the set of proxy signers' public keys $S = \{pk_i\}_{i=1}^{l}$ is given to the adversary. In addition, the challenger records the random coins $\{b_i\}$ used in generating each key pair. Here l is a game parameter.

2. The adversary is given access (throughout the entire game) to make APS generation queries of the form (s, σ_w, M, R), where M is the message to be signed, R is a set of public keys, and s is an index such that $pk_s \in R$ holds. The challenger responds with $\sigma = Sig_{sk_s}(\sigma_w, M, R)$.

3. The adversary outputs a message M, distinct indices i_0, i_1, and a ring R for which $pk_{i0}, pk_{i1} \in R$. A random bit b is chosen, and the adversary is given the signature $Sig_{sk_{i_b}}(\sigma_w, M, R)$. In addition, the challenger provides the adversary with the coins $\{b_i\}$ used to generate the keys.

4. The adversary outputs a bit b', and succeeds if $b' = b$.

Definition 1. An APS scheme is anonymous against full key exposure attack if no PPT adversary has advantage non-negligibly greater than $1/2$ of winning in the above game.

3.3 Existential unforgeability in APS schemes

To discuss the unforgeability of APS schemes, we categorize the adversaries into three types [15, 21].

Type1: The adversary only has the public keys of the original signer and proxy signers.

Type2: The adversary has the public keys of the original signer and proxy signers; besides it has the secret key of the original signer.

Type3: The adversary has the public keys of the original signer and proxy signers; besides it has the secret keys of some proxy signers.

It can be found that if an APS scheme is existential unforgeable against Type2 and Type3 adversaries, it is also existential unforgeable against Type1 adversary.

a. Existential unforgeability against adaptive Type2 adversary

The existential unforgeability of an APS scheme against a Type2 adversary means that it is difficult for any entity, including the original signer, other than the proxy signers themselves to generate a valid APS on a message it chooses, even he has obtained the secret key of the original signer and some valid anonymous proxy signatures on messages it chooses. Motivated by the work of Bender et al. [24, 25], we provide a formal definition of existential unforgeability of an APS scheme against Type2 adversary as follows.

Given an APS scheme and a PPT adversary, consider the following game:

1. Key pairs $\{(pk_i, sk_i)\}_{i=1}^{l}$ are generated and the set of public keys $S = \{pk_i\}_{i=1}^{l}$, the public key and secret key of the original signer is given to the adversary. Here l is a game parameter.

2. The adversary is given access (throughout the entire game) to make APS generation queries of the form (s, σ_w, M, R), where M is the message to be signed, R is a set of public keys, and s is an index such that $pk_s \in R$ holds. The challenger responds with $\sigma = Sig_{sk_s}(\sigma_w, M, R)$.

3. The adversary is also given access to a corrupt oracle $Corrupt(.)$, which on input i, returns sk_i.

4. The adversary Outputs $(R^*, \sigma_{w^*}, M^*, \sigma^*)$ and succeeds if $Vrfy_{R^*}(M^*, w^*, \sigma^*) = 1$, it never queried $(\star, \sigma_{w^*}, M^*, R^*)$ and $R^* \subseteq S \backslash C$, where C is the set of corrupted users.

Definition 2. An APS scheme is secure against Type2 adversary if no PPT adversary has a non-negligible advantage in the above game.

b. **Existential unforgeability against adaptive Type3 adversary**

The existential unforgeability of an APS scheme against a Type3 adversary means that it is difficult for an attacker to forge a valid delegation signature on a warrant it chooses, even it has obtained the secret keys of some proxy signers and some other valid delegation signatures on warrants it chooses. We provide a formal definition of existential unforgeability of an APS scheme against Type3 adversary as follows.

Given an APS scheme and a PPT adversary, consider the following game:

1. Key pairs $\{(pk_i, sk_i)\}_{i=1}^{l}$ are generated and the set of public keys $S = \{pk_i\}_{i=1}^{l}$, the public key of the original signer and the secret keys of some proxy signers is given to the adversary. Here l is a game parameter.

2. The adversary is given access (throughout the entire game) to an oracle $Delegationsign(\dots)$ such that $Delegationsign(w)$ returns $Sign_{sk_a}(w)$, where sk_a is the secret key of original signer.

3. The adversary outputs (w^*, σ_{w^*}) and succeeds if $Vrfy(w^*, \sigma_{w^*}) = 1$ and it never queried w^*.

Definition 3. An APS scheme is secure against Type3 adversary if no PPT adversary has a non-negligible advantage in the above game.

Definition 4. An APS scheme is existential unforgeable against adaptive chosen-message attack if it is secure against both Type2 and Type3 adversaries.

4 Proposal

4.1 Setup

The trusted setup algorithm first constructs a group G of composite order $n = pq$ as described in section II. It then chooses random exponents $(a, b_0 \overset{R}{\leftarrow} Z_n)$ and sets

$$A = g^a$$

$$B_0 = g^{b_0}$$

$$A' = h^a$$

Let $H : \{0,1\}^* \rightarrow \{0,1\}^k$ be a collision-resistant hash function. The setup algorithm chooses generators

$$u', u_1, u_2, \dots, u_k \overset{R}{\leftarrow} G$$

Let (G, G_T) be bilinear groups where $|G| = |G_T| = n$, g is the generator of G and e denotes an admissible pairing $G \times G \rightarrow G_T$.

The published common reference string includes a description of the groups G and G_T and of the collision-resistant hash H, along with (A, B_0, A') and $(u', u_1, u_2, \ldots, u_k)$. The factorization of n is not revealed .Note that anyone can use the pairing to verify that the pair (A, A') is properly formed [22].

4.2 Key Generation

Original signer picks random $x_a \overset{R}{\leftarrow} Z_n$ and sets his secret key $sk_a = A^{x_a}$ and his public key $pk_a = g^{x_a}$. In the same way, the proxy ring members compute their private keys and public keys. Note that, we have

$$sk_a = (pk_a)^a \tag{5}$$

4.3 Delegation Generation

Let $w = (w_1, w_2, \ldots, w_k)$ be a k-bit warrant to be signed by the original signer. This algorithm picks a random $r_a \overset{R}{\leftarrow} Z_n$ and computes the delegation $\sigma_w = (\sigma_{w1}, \sigma_{w2})$ and sends it to the proxy ring members, where

$$\sigma_{w1} = sk_a . \left(u' \prod_{i=1}^{k} u_i^{w_i}\right)^{r_a} \tag{6}$$

$$\sigma_{w2} = g^{r_a} \tag{7}$$

4.4 Delegation Verification

Upon receiving $(w, \sigma_{w1}, \sigma_{w2})$, this algorithm checks that the following equation is satisfied or not

$$e(\sigma_{w1}, g) = e(\sigma_{w2}, u' \prod_{i=1}^{k} u_i^{w_i}) e(A, pk_a) \tag{8}$$

If it does not hold, the delegation will be rejected. Otherwise, it will be accepted.

4.5 APS Generation

This algorithm takes as input a message $M \in \{0,1\}^*$, a ring R of public keys of proxy signers, the signature σ_w on w, and a key pair $(pk, sk) \in G^2$, where

$$x_b \overset{R}{\leftarrow} Z_n$$

$$sk = A^{x_b}$$

$$pk = g^{x_b}$$

No key may appear twice in R, and R must include pk .

Let $l = |R|$; it parses the elements of R as $v_i \in G$ for each i, $1 \leq i \leq l$.

Let i^* be the index such that $v_{i^*} = pk$. We define

$$f_i = \begin{cases} 1 & if\ i = i^* \\ 0 & O.w \end{cases}$$

Now for each i, $1 \leq i \leq l$, the algorithm chooses a random exponent $t_i \overset{R}{\leftarrow} Z_n$ and sets

$$C_i \leftarrow \left(\frac{v_i}{B_0}\right)^{f_i} h^{t_i} \tag{9}$$

$$\pi_i \leftarrow \left(\left(\frac{v_i}{B_0}\right)^{2f_i - 1} h^{t_i}\right)^{t_i} \tag{10}$$

$$C \leftarrow \prod_{i=1}^{l} C_i \tag{11}$$

$$t \leftarrow \sum_{i=1}^{l} t_i \tag{12}$$

It computes $(m_1, m_2, \ldots, m_k) \leftarrow H(M, R, w)$. Finally, it chooses $r \overset{R}{\leftarrow} Z_n$ and computes

$$S_1 \leftarrow \sigma_{w1}.sk.\left(u' \prod_{j=1}^{k} {u_j}^{m_j}\right)^r.A'^t \tag{13}$$

$$S_2 \leftarrow g^r \tag{14}$$

$$S_3 \leftarrow \sigma_{w2} \tag{15}$$

The signature is output as σ, where

$$\sigma = \left((S_1, S_2, S_3), \{(C_i, \pi_i)\}_{i=1}^{l}\right) \in G^{2l+3}$$

4.6 APS Verification

The verifier computes $(m_1, m_2, \ldots, m_k) \leftarrow H(M, R, w)$. Let $l = |R|$; it parses the elements of R as $v_i \in G$, for each i, $1 \leq i \leq l$ and checks that no element is repeated in R and rejects otherwise. Then it parses the signature σ as $\left((S_1, S_2, S_3), \{(C_i, \pi_i)\}_{i=1}^{l}\right) \in G^{2l+3}$. (If this parse fails, reject.). The verifier checks first that the $\{\pi_i\}_{i=1}^{l}$ are valid or not

$$e(C_i, C_i/(v_i/B_0)) = e(h, \pi_i) \tag{16}$$

If any of the proofs is invalid, reject. Otherwise, verifier sets $C \leftarrow \prod_{i=1}^{l} C_i$. Accept if the following equation is satisfied

$$e(S_1, g) = e(S_2, u' \prod_{j=1}^{k} u_j^{m_j})\, e(A, pk_a B_0 C) e(S_3, u' \prod_{i=1}^{k} u_i^{w_i}) \tag{17}$$

Note that, there is exactly one non-zero value amongst $\{f_i\}$, and we have

$$B_0 C = (v_{i^*})(h^t) \tag{18}$$

The correctness of the proposal can be verified directly, as following equations

$$e(S_1, g) = e(\sigma_{w1}.sk.\left(u'\prod_{j=1}^{k} u_j^{m_j}\right)^r.A'^t, g)$$

$$= e(\sigma_{w1}, g)e(sk, g)e\left(\left(u'\prod_{j=1}^{k} u_j^{m_j}\right)^r, g\right).e(A'^t, g)$$

$$= e(\left(u'\prod_{i=1}^{k} u_i^{w_i}\right)^{r_a}, g)e(sk_a, g)e(sk, g)e(\left(u'\prod_{j=1}^{k} u\right)^r, g)e(A'^t, g)$$

$$= e(u'\prod_{i=1}^{k} u_i^{w_i}, g^{r_a})e(A, g^{x_a})e(A, g^{x_b})e(u'\prod_{j=1}^{k} u_j^{m_j}, g^r)e(h^{at}, g)$$

$$= e(S_3, u'\prod_{i=1}^{k} u_i^{w_i})e(A, pk_a)e(S_2, u'\prod_{j=1}^{k} u_j^{m_j})e(A, g^{x_b})e(h^{at}, g)$$

$$= e(S_3, u'\prod_{i=1}^{k} u_i^{w_i})e(A, pk_a)e(S_2, u'\prod_{j=1}^{k} u_j^{m_j})e(A, v_{i^*})e(A, h^t)$$

$$= e(S_2, u'\prod_{j=1}^{k} u_j^{m_j})\, e(A, B_0C)e(A, pk_a)e(S_3, u'\prod_{i=1}^{k} u_i^{w_i})$$

$$= e(S_2, u'\prod_{j=1}^{k} u_j^{m_j})\, e(A, pk_aB_0C)e(S_3, u'\prod_{i=1}^{k} u_i^{w_i})$$

5 security analysis

We will analyze the security of our proposal in this section. Our proposal is warrant based and the delegation is original signer's signature on the warrant which contains proxy signers' public key, a period of validity, the restrictions on the messages that the signer can sign and so on. Therefore, this kind of proxy signature can prevent the misuse of the delegation [21]. Some other properties such as distinguishability (distinguishable from normal signatures) and nondeniability can be achieved naturally too. Therefore, we mainly focus on the anonymity and the existential unforgeability of our proposal.

5.1 Anonymity

Theorem 1. Our proposal is anonymous against full key exposure attack if SGH problem is hard.

Proof. Consider a challenger that wants to solve the SGH (subgroup hiding) problem. The group order n, the description of the group G with the generators g of G and h, which in Game 0, h is chosen randomly from Gq and in Game 1, h is chosen randomly from G, is given to the challenger. After receiving a SGH challenge (n, G, g, h), the challenger follows the setup algorithm of Section 4 to obtain system parameters. It then runs $Key\ Generation$ for l times to obtain public-private key pairs $\{(pk_i, sk_i)\}_{i=1}^{l}$ and the randomnesses $\{b_i\}_{i=1}^{l}$ used in each run and sends the public parameters to an adversary and plays the anonymity game with it. The adversary requests a challenge by sending to the challenger the values $(i_0, i_1, R, M, \sigma_w)$. Here i_0 and i_1 are indices such that $pk_{i_0}, pk_{i_1} \in R$. The challenger chooses $b \overset{R}{\leftarrow} \{0,1\}$, creates the signature $\sigma \leftarrow Sig(pk_{i_b}, sk_{i_b}, R, M, \sigma_w)$, and responds to the adversary with σ and the randomnesses $\{b_i\}_{i=1}^{l}$ used to generate the keys. The adversary finally guesses b' for b. If it answers correctly, then the challenger outputs 1, guessing $h \in Gq$; otherwise it outputs 0, guessing $h \in G$.

We define the advantage of the adversary in the Game i, $i = 0, 1$, by $Adv_{\mathcal{A}}^{game\ i}$ and the advantage of the challenger in the SGH game by $Adv_{\mathcal{B}}^{SGH}$. We know $Pr[w \in G] = Pr[w \in Gq] = 1/2$, therefore

$$Adv_{\mathcal{A}}^{game\ 0} - Adv_{\mathcal{A}}^{game\ 1} = Pr[output = 1|game\ 0] - Pr[output = 1|game\ 1] =$$

$$2Pr[output = 1, game\ 0] - 2Pr[output = 1, game\ 1] = 2Adv_{\mathcal{B}}^{SGH} \qquad (19)$$

In Game 1, h is a generator of G, therefore exist $\tau_{i0}, \tau_{i1} \in Z_n$ such that $\left(\frac{v_i}{B_0}\right) h^{\tau_{i1}} = h^{\tau_{i0}}$ and we have

$$(\pi_i|f_i = 1) = \left(\left(\frac{v_i}{B_0}\right) h^{\tau_{i1}}\right)^{\tau_{i1}} = (h^{\tau_{i0}})^{\tau_{i1}} = (h^{\tau_{i1}})^{\tau_{i0}} = \left(h^{\tau_{i0}} \left(\frac{v_i}{B_0}\right)^{-1}\right)^{\tau_{i0}}$$

$$\Rightarrow (\pi_i|f_i = 1) = (\pi_i|f_i = 0) \qquad (20)$$

In other words, the adversary can obtain no information from the pair (C_i, π_i) of the signature, for $1 \leq i \leq l$ in Game 1. Because S_2 and S_3 are independent of the signer choice, therefore the only part of signature may have information for the adversary, is S_1. But, S_1 is the unique value satisfying (17) because of having fixed other parts of the signature σ. Thus the indices i_0 and i_1 are independent of the entire signature σ. It means

$$Adv_{\mathcal{A}}^{game\ 1} = 0 \qquad (21)$$

We define $Adv_{\mathcal{A}}^{anon}$ to be the advantage of the adversary in the anonymity game and $Adv_{\mathcal{B}}^{sgh}$ to be the advantage of the challenger in the SGH problem solving. Because in Game 0, the environment of the adversary is the same as in the anonymity game, we have

$$Adv_{\mathcal{A}}^{game\ 0} = Adv_{\mathcal{A}}^{anon} \qquad (22)$$

Putting equations (19), (21) and (22) together, we obtain

$$Adv_{\mathcal{A}}^{anon} \leq 2Adv_{\mathcal{B}}^{sgh} \tag{23}$$

5.2 Unforgeability

Theorem 2. Our proposal is existential unforgeable against adaptive chosen-message attack if H is collision-resistant and CDH is hard in Gp.

Proof. We need to show that our proposal is secure against Type2 and Type3 adversaries in the standard model.

Theorem 2.1. Our proposal is secure against Type2 adversary if H is collision-resistant and CDH is hard in Gp.

Proof. First we need some requirements as follows

- If d_p satisfies $d_p = 0\ mod\ q$ and $d_p = 1\ mod\ p$, for all $z \in G$ we have

$$z^{d_p} \in G_p,\ z^{d_p} = 1 \text{ iff } z \in G_q \tag{24}$$

Thus we have $C_i^{d_p} = 1$ iff $f_i = 0$, for the pair (C_i, π_i) that satisfies (16).

- If η and h are generators of Gp and Gq respectively, $\eta^{r_1} h^{r_2}$ is a random generator of G for all $r_1 \overset{R}{\leftarrow} Z_p^*$ and $r_2 \overset{R}{\leftarrow} Z_q^*$.
- For all $u \in G_p$ and $v \in G_q$, we have

$$e(u, v) = 1 \tag{25}$$

- For all $u_1, u_2 \in G_p$ and $v_1, v_2 \in G_q$, we have

$$e(u_1 v_1, u_2 v_2) = e(u_1, u_2) e(v_1, v_2) \tag{26}$$

For the proof of Theorem 2.1. we define three types of adversaries as follows:

TypeI: The adversary issues two pairs (M, R) and (M', R') such that $(M, R) \neq (M', R')$ but $H(M, R, w) = H(M', R', w)$.

TypeII: The adversary does not issue a hash collision as above and it forges such that $\sum_i f_i \neq 1$.

TypeIII: The adversary does not issue a hash collision as above and it forges such that $\sum_i f_i = 1$.

We show a challenger which is given the factorization $n = pq$, can break one of our complexity assumptions, based on a forger adversary.

TypeI Adversary. Consider a challenger that wants to solve the collision-resistance problem. The challenger follows the setup algorithm of part 4 to obtain system pa-

rameters. It then runs $Key\ Generation$ for l times to obtain public-private key pairs $\{(pk_i, sk_i)\}_{i=1}^{l}$ and sends the public parameters to an adversary. The challenger also initializes the set C of corrupted users as $C \leftarrow \emptyset$ and continues the unforgeability game with the adversary. The adversary makes signing queries and corruption queries. A signing query is of the form (s, R, M, σ_w), where s is an index such that $pk_s \in R$. When it makes a signing query, the challenger responds with $\sigma \leftarrow Sig(pk_s, sk_s, R, M, \sigma_w)$ and it keeps the pairs $(M^{(i)}, R^{(i)})$ included in signing queries of the adversary. A corruption query is of the form s, where s is an index such that $pk_s \in R$. When the adversary makes a corruption query, the challenger responds with sk_s to it and adds pk_s to C. Eventually, the adversary outputs a tuple $(R^*, M^*, \sigma^*, \sigma_w)$ and wins the game if

- it never made a signing query (s, R^*, M^*, σ_w) for any s;
- $R^* \subseteq \{pk_i\} \setminus C$;
- $(R^*, M^*, \sigma^*, \sigma_w)$ satisfies (17);

In addition, the challenger keeps the pair (M^*, R^*) included in its forgery. In this type of adversary there are two pairs (M, R) and (M', R') such that $H(M, R, w) = H(M', R', w)$. The challenger outputs this pair as a collision and succeeds when the adversary does.

We define $Adv_{\mathcal{A}}^{uf}$ to be the advantage of the adversary in the unforgeability game and $Adv_{\mathcal{B}}^{H\_coll}$ to be the advantage of the challenger in the collision-resistance problem solving, so have

$$Adv_{\mathcal{A}}^{uf} \leq Adv_{\mathcal{B}}^{H\_coll} \tag{27}$$

TypeII Adversary. Consider a challenger that wants to solve the CDH problem. The group order n and its factorization $n = pq$, the description of the group G with η and h the generators of Gp and Gq respectively, and a pair $(\eta^{\alpha}, \eta^{\beta}) \in G_p^2$ is given to the challenger. It's goal is to compute $\eta^{\alpha\beta}$. The challenger chooses a collision-resistant hash function $H: \{0,1\}^* \rightarrow \{0,1\}^k$ and picks $r_1 \overset{R}{\leftarrow} Z_q^*$ and $r_2, r_3 \overset{R}{\leftarrow} Z_q$. Then it sets

$$g \leftarrow \eta h^{r_1}$$

$$A \leftarrow \eta^{\alpha} h^{r_2}$$

$$A' \leftarrow h^{r_2/r_1}$$

$$B_0 \leftarrow \eta^{\beta} h^{r_3}$$

g is a random generator of G. Anyone can use the pairing to verify that the pair (A, A') is properly formed.

$$e(A, h) = e(\eta^{\alpha} h^{r_2}, h) = e(\eta^{\alpha}, h) e(h^{r_2}, h) = e(h^{r_2}, h) = e\left(h^{r_2/r_1}, h^{r_1}\right) = e(A', g) \tag{28}$$

The challenger picks exponents $x', x_1, x_2, \dots, x_k \overset{R}{\leftarrow} Z_n$, and sets $u' \leftarrow g^{x'}$ and $u_j \leftarrow g^{x_j}$ for $1 \le j \le k$. It picks random exponents b_i and sets user keys as $pk_i \leftarrow g^{b_i}$ and $sk_i \leftarrow A^{b_i}$ for $1 \le i \le l$. Then the challenger sends the generators g and h, the parameters (A, A', B_0) and $(u', u_1, \dots, u_k)$, the description of H and the public keys $\{pk_i\}_{i=1}^{l}$ to an adversary. The challenger responds the signing and corruption queries of the adversary as above. Eventually, the adversary outputs a forgery ple $(R^*, M^*, \sigma^*, \sigma_w)$, where $l^* = |R^*|$. We can parse the R^* as $(v_1, v_2, \dots, v_{l^*})$ and assume a mapping $t: [1, l^*] \to [1, l]$ such that $v_i = pk_{t(i)}$, for $1 \le i \le l^*$.

The challenger parses σ^* as $\left((S_1, S_2, S_3), \{(C_i, \pi_i)\}_{i=1}^{l^*}\right) \in G^{2l^*+3}$. (We assume a non-trivial adversary, so this parse must succeed and each pair (C_i, π_i) must satisfy (16)).

After parsing we have for each i, $1 \le i \le l^*$

$$C_i^{d_p} = \left(v_i^{d_p} / B_0^{d_p}\right)^{f_i} = \left((g^{b_{t(i)}})^{d_p} / B_0^{d_p}\right)^{f_i} = \left(\eta^{b_{t(i)}} / \eta^{\beta}\right)^{f_i} \tag{29}$$

Now we define

$$b = \sum_{i=1}^{l^*} f_i \, b_{t(i)} \tag{30}$$

$$f = \sum_{i=1}^{l^*} f_i \tag{31}$$

Putting equations (29-31) together, we have

$$C^{d_p} = \prod_{i=1}^{l^*} C_i^{d_p} = \eta^b / \left(\eta^{\beta}\right)^f \tag{32}$$

Finally, the challenger computes $(m_1, m_2, \dots, m_k) \leftarrow H(M^*, R^*, w)$ and sets

$$x \leftarrow x' + \sum_{i=1}^{k} m_i x_i \tag{33}$$

$$y \leftarrow x' + \sum_{i=1}^{k} w_i x_i \tag{34}$$

We raise both sides of equation (17) to power d_p, by substituting parameters (g, A, A', B_0) as above, using equations (24-26) and equations (32-34), we have

$$e\left(\eta^{\alpha}, \eta^{\beta}.\frac{\eta^b}{(\eta^{\beta})^f}\right) = e\left(S_1^{d_p}, \eta\right).e\left(\left(S_2^{d_p}\right)^{-1}, \eta^x\right).e\left(\left(S_3^{d_p}\right)^{-1}, \eta^y\right) e\left(\eta^{\alpha}, pk_a^{d_p}\right)$$

$$\Rightarrow e(\eta^{\alpha}, \eta^{\beta})^{1-f} = e((\eta^{\alpha})^{-b}.S_1^{d_p}.\left(S_2^{d_p}\right)^{-x}.\left(S_2^{d_p}\right)^{-y}.pk_a^{\alpha d_p}, \eta) \tag{35}$$

So the answer to the CDH problem is

$$\eta^{\alpha\beta} = \left[(\eta^{\alpha})^{-b}.S_1^{d_p}.\left(S_2^{d_p}\right)^{-x}.\left(S_2^{d_p}\right)^{-y}.pk_a^{\alpha d_p}\right]^{1/(1-f)} \tag{36}$$

Remind that in this type of adversary, $f \neq 1$.

The challenger succeeds whenever the adversary does. So by defining $Adv_{\mathcal{B}}^{G_p\_CDH}$ to be the advantage of the challenger in the CDH problem solving, we have

$$Adv_{\mathcal{A}}^{uf} \leq Adv_{\mathcal{B}}^{G_p\_CDH} \tag{37}$$

TypeIII Adversary. We wish to convert a TypeIII adversary to a Waters signature forger in G_p. The proof proceeds as follows.

The group order n and its factorization $n = pq$, the description of the group G with η and h the generators of Gp and Gq respectively, WS public parameters $\hat{u}', \hat{u}_1, \dots, \hat{u}_k$ all in G_p, and a WS public key $(\eta_1, \eta_2) \in G_p^2$ is given to a challenger. The challenger chooses a collision-resistant hash function $H: \{0,1\}^* \to \{0,1\}^k$ and picks $r_1 \overset{R}{\leftarrow} Z_q^*$, $r_2, r_3 \overset{R}{\leftarrow} Z_q$ and $x \overset{R}{\leftarrow} Z_p$. Then it sets

$$g \leftarrow \eta h^{r_1}$$

$$A \leftarrow \eta_1 h^{r_2}$$

$$A' \leftarrow h^{r_2/r_1}$$

$$B_0 \leftarrow \eta^x h^{r_3}$$

The challenger picks random exponents $s', s_1, \dots, s_k \overset{R}{\leftarrow} Z_q$ and sets $u' \leftarrow \hat{u}' h^{s'}$ and $u_j \leftarrow \hat{u}_j h^{s_j}$, for $1 \leq j \leq k$. For generating user keys, it first picks randomly i^* from $\{1, 2, \dots, l\}$, then for each $i \neq i^*$, it chooses a random exponent $b_i \overset{R}{\leftarrow} Z_n$ and sets $pk_i \leftarrow g^{b_i}$ and $sk_i \leftarrow A^{b_i}$ and for i^*, $\mathcal{B}$ picks $r_4 \overset{R}{\leftarrow} Z_q$ and sets $pk_{i^*} \leftarrow \eta_2 h^{r_4}$. It sends the generators g and h, the parameters (A, A', B_0) and $(u', u_1, \dots, u_k)$, the description of H and the public keys $\{pk_i\}_{i=1}^{l}$ to an adversary. The adversary makes signing queries and corruption queries. A corruption query is of the form s, where s is an index such that $pk_s \in R$. When it makes a corruption query, the challenger responds with sk_s unless s equals i^*, in which case the challenger declares failure and exits.

A signing query is of the form (s, R, M, σ_w), where s is an index such that $pk_s \in R$. When the adversary makes a signing query, if $s \neq i^*$, the challenger responds with $\sigma \leftarrow Sig(pk_s, sk_s, R, M, \sigma_w)$. If $s = i^*$, it first requests from WS signing oracle a signature on $(m_1, m_2, \dots, m_k) \leftarrow H(M, R, w)$. The WS signing oracle responds with $\hat{\sigma} = (\widehat{S_1}, \widehat{S_2}) \in G_p^2$. Then the challenger blinds the signature $\hat{\sigma}$ and projects it into G, as follows. Let $l' = |R|$. The challenger parses R as $(v_1, v_2, \dots, v_{l'})$. For $1 \leq i \leq l'$, we define

$$f_i = \begin{cases} 1 & if\ v_i = pk_{i^*} \\ 0 & O.w \end{cases} \tag{38}$$

For each i, $1 \leq i \leq l'$, the challenger picks a random exponent $t_i \overset{R}{\leftarrow} Z_n$ and sets C_i, π_i similar (9) and (10). By defining t, C similar (11) and (12) and choosing $r \overset{R}{\leftarrow} Z_q$, it computes

$$S_1 \leftarrow \sigma_{w1}.\widehat{S_1}.h^{\frac{r_2 r_4}{r_1}}.\left(h^{\frac{s'}{r_1}}\prod_{j=1}^{k}h^{\frac{s_j m_j}{r_1}}\right)^{r}.h^{\frac{r_2 t}{r_1}} \quad (39)$$

$$S_2 \leftarrow \widehat{S_2}.h^{r} \quad (40)$$

$$S_3 \leftarrow \sigma_{w2}$$

By substituting parameters (g, A, A', B_0) as above and using equations (39, 40) and (24-26), we have

$$e(S_1, g) =$$

$$= e(\sigma_{w1}, g)e\left(\hat{S}_1, \eta h^{r_1}\right)e\left(h^{\frac{r_2 r_4}{r_1}}, \eta h^{r_1}\right)e\left(h^{\frac{s'}{r_1}}\prod_{j=1}^{k}h^{\frac{s_j m_j}{r_1}}, \eta h^{r_1}\right)^{r}e\left(h^{\frac{r_2 t}{r_1}}, \eta h^{r_1}\right)$$

$$= e(\sigma_{w1}, g)\, e(\hat{S}_1, \eta)e\left(h^{\frac{r_2 r_4}{r_1}}, h^{r_1}\right)e\left(h^{\frac{s'}{r_1}}\prod_{j=1}^{k}h^{\frac{s_j m_j}{r_1}}, h^{r_1}\right)^{r}e\left(h^{\frac{r_2 t}{r_1}}, h^{r_1}\right)$$

$$= e(\sigma_{w1}, g)e\left(\eta_1, \eta_2\right)e(\hat{S}_2, \hat{u}'\prod_{i=1}^{k}\hat{u}_i^{m_i})e(h^{r_2}, h^{r_4})e(h^{s'}\prod_{j=1}^{k}h^{s_j m_j}, h^{r})e(h^{r_2 t}, h)$$

$$= e(\sigma_{w1}, g)e(A, pk^*)e(\hat{S}_2, \hat{u}'\prod_{i=1}^{k}\hat{u}_i^{m_i})e(h^{s'}\prod_{j=1}^{k}h^{s_j m_j}, h^{r})e(h^{r_2 t}, h)$$

$$= e(\sigma_{w1}, g)e(A, pk^*)e(\hat{S}_2 h^{r}, (\hat{u}'h^{s'})\prod_{j=1}^{k}\left(\hat{u}_j h^{s_j}\right)^{m_j})e(h^{r_2 t}, h)$$

$$= e(\sigma_{w1}, g)\, e(A, pk^*)e(S_2, u'\prod_{j=1}^{k}u_j{}^{m_j})e(h^{r_2 t}, h)$$

$$= e(\sigma_{w1}, g)e(A, CB_0 h^{-t})e(S_2, u'\prod_{j=1}^{k}u_j{}^{m_j})e(h^{r_2 t}, h)$$

$$= e(\sigma_{w1}, g)e(A, CB_0)e(h^{r_2}, h^{-t})e(S_2, u'\prod_{j=1}^{k}u_j{}^{m_j})e(h^{r_2 t}, h)$$

$$= e(S_3, u'\prod_{i=1}^{k}u_i^{w_i})e(A, pk_a B_0 C)e(S_2, u'\prod_{j=1}^{k}u_j{}^{m_j})$$

Above proofs satisfies equation (17). Thus the challenger responds to the adversary with

$$\sigma = \left((S_1, S_2, S_3), \{(C_i, \pi_i)\}_{i=1}^{l}\right) \in G^{2l+3}$$

Eventually, the adversary outputs a forgery tuple $(R^*, M^*, \sigma^*, \sigma_w)$, where $l^* = |R^*|$. We can parse the R^* as $(v_1, v_2, \dots, v_{l^*})$ and assume a mapping $t: [1, l^*] \to [1, l]$ such that $v_i = pk_{t(i)}$, for $1 \le i \le l^*$. Besides, the adversary must not have made a corruption query at any of the indices $\{t(i)\}_{i=1}^{l^*}$.

The challenger parses σ^* as $\left((S_1, S_2, S_3), \{(C_i, \pi_i)\}_{i=1}^{l^*}\right) \in G^{2l^*+3}$. (We assume a non-trivial adversary, so this parse must succeed and each pair (C_i, π_i) must satisfy (16)). Since the adversary is TypeIII, there is exactly one index $s \in \{1, 2, \dots, l^*\}$ such that $f_s = 1$. If $t(s) \neq i^*$, the challenger declares failure and exits, otherwise we have for each i, $1 \le i \le l^*$

$$C_i^{d_p} = \left(\frac{v_i^{d_p}}{B_0^{d_p}}\right)^{f_i}$$

$$\Rightarrow C^{d_p} = \prod_{i=1}^{l^*} C_i^{d_p} = (pk_{i^*})^{d_p} / B_0^{d_p} \tag{41}$$

Finally, the challenger computes $(m_1, m_2, \dots, m_k) \leftarrow H(M^*, R^*, w)$. We raise both sides of equation (17) to power d_p, by substituting parameters (g, A, A', B_0) as above, using equations (24-26) and equation (41), we have

$$e((S_1 \sigma_{w1}^{-1})^{d_p}, \eta).\, e\left(\left(S_2^{d_p}\right)^{-1}, \hat{u}' \prod_{i=1}^{k} \hat{u}_i^{m_i}\right) = e\left(\eta_1, B_0^{d_p} C^{d_p}\right) = e(\eta_1, \eta_2)$$

So $((S_1 \sigma_{w1}^{-1})^{d_p}, S_2^{d_p})$ is a valid WS signature on $H(M^*, R^*, w)$.

We define Adv_B^{WS-uf} to be the advantage of the challenger in the creating a WS forgery. Index i^* is uniformly chosen from the set $\{1,2, \dots, l\}$ and the challenger succeeds in creating a WS forgery whenever the adversary does, so we have

$$Adv_A^{uf} \le l.Adv_B^{WS-uf} \tag{42}$$

Note that WS is unforgeable if CDH is hard in G_p.

Theorem 2.2. Our proposal is secure against Type3 adversary if CDH is hard in G.

Proof. Consider a challenger wants to solve the CDH problem in which a random tuple $(g, g^a, g^b) \epsilon G^3$ and the group order n is given to the challenger and its goal is to compute g^{ab}. The adversary interacts with the challenger in this game.

The challenger picks following exponents

$$x' \overset{R}{\in} Z_n, x_i \overset{R}{\in} Z_n \text{ for each } i,\ 1 \le i \le k$$

Now, the challenger sets the public parameters to be

$$A = g^b$$

$$u' = gA^{x'} \text{ and } u_i = A^{x_i} \text{ for } i = 1, \ldots, k$$

Besides, he sets the original signer's public key to be

$$pk_a = g^a$$

We further define the following function

$$F(w) = x' + \sum_{i=1}^{k} x_i w_i \tag{43}$$

So, we have

$$u' \prod_{i=1}^{k} u_i^{w_i} = gA^{F(w)} \tag{44}$$

Then, the challenger sends the public parameters to the adversary and responds the delegation signature queries of it as follows.

When the challenger receives a of a warrant w, it randomly chooses $r \in Z_n$ and computes

$$\sigma_{w_1} = pk_a^{\frac{-1}{F(w)}} \left(u' \prod_{i=1}^{k} u_i^{w_i} \right)^r$$

$$= pk_a^{\frac{-1}{F(w)}} \left(gA^{F(w)} \right)^r$$

$$= A^a \left(gA^{F(w)} \right)^{\frac{-a}{F(w)}} \left(gA^{F(w)} \right)^r$$

$$= A^a \left(gA^{F(w)} \right)^{r - \frac{a}{F(w)}}$$

By defining $\bar{r} = r - \frac{a}{F(w)}$, we have

$$\sigma_{w_1} = A^a \left(gA^{F(w)} \right)^{\bar{r}} \tag{45}$$

$$\sigma_{w_2} = g^{\bar{r}} = g^{r - \frac{a}{F(w)}} = g^r pk_a^{\frac{-1}{F(w)}} \tag{46}$$

So, the challenger answers as $\sigma_w = (\sigma_{w_1}, \sigma_{w_2})$. Note that the verification equation (8) is satisfied.

The adversary will output a forge signature $\sigma_w^* = (\sigma_{w_1}^*, \sigma_{w_2}^*)$ on a warrant w^* such that w^* has not been queried during the delegation signature queries and σ_w^* is a valid signature of the warrant w^*.

If $F(w^*) \neq 0\ (modp)$, the challenger will abort. Otherwise, we set the signature as

$$(\sigma_{w_1}^*, \sigma_{w_2}^*) = (A^a \left(u' \prod_{i=1}^{k} u_i^{w_i} \right)^r, g^r)$$

So the answer to the CDH problem is

$$g^{ab} = \frac{\sigma_{w_1}^*}{\sigma_{w_2}^*} \tag{47}$$

The challenger succeeds whenever the adversary does, so we have

$$Adv_{\mathcal{A}}^{uf} \leq Adv_{\mathcal{B}}^{G\_CDH} \tag{48}$$

Putting equations (27), (37), (42) and (48) together, we obtain

$$Adv_A^{uf} \leq Adv_B^{H_{coll}} + Adv_B^{G_{p_{CDH}}} + l.Adv_B^{WS-uf} + Adv_{\mathcal{B}}^{G\_CDH} \tag{49}$$

6 Comparisons

In this section, we will compare our proposal with others from the computational costs, construction and model of provable security of schemes viewpoints. Since the pairing computation is the most time consuming, we compare the computational costs of schemes based on the number of pairing operations used. To generate an APS σ on a message M with a warrant w and by using l proxy signers, we summarize the comparisons in the Table 1:

	Number of Pairing Operations Used	Definition of APS	Model of Provable Security
Zhang's Scheme [20]	$4l + 1$	proxy signer's privacy protection	random oracle model
Yu's Scheme [15]	$l + 1$	proxy signer's privacy protection	random oracle model
Fuchsbauer's Scheme [26]	no concrete scheme	anonymity for delegatee and delegators (Group Signature-Based)	without random oracles
Our Proposal	$2l + 7$	proxy signer's privacy protection	without random oracles

Table 1. Comparison of our proposal construction and model of provable security with existing schemes

7 Conclusions

In this paper, we proposed the first provable secure anonymous proxy signature scheme without random oracles which is the combination of proxy signature and ring

signature. This proposal avoids the criticisms of the random oracle model and becomes needful whenever proxy signer wants to sign message on behalf of the original signer providing anonymity.

For the security analysis, we categorized the adversaries into three types according to different resources they can get and showed that, our proposal to be anonymous against full key exposure attack and existential unforgeable against all kinds of adversaries in the standard model with the computational Diffie–Hellman and the subgroup hiding assumptions in bilinear groups.

The signature in our proposal is of size $2l + 3$ group elements for l members of proxy signers and requires $2l + 7$ pairings to verify.

References

[1] M. Mambo, K. Usuda, E. Okamoto, Proxy signatures for delegating signing operation, 3rd ACM Conference on Computer and Communications Security (1996) 48-57.

[2] M. Mambo, K. Usuda, E. Okamoto, Proxy signature: Delegation of the power to sign messages, IEICE Transactions on Fundamentals, Vol. E79-A, No. 9 (1996) 1338-1353.

[3] C. Ma, J. Ao, Group-based proxy re-encryption scheme secure against chosen ciphertext attack, International Journal of Network Security, Vol. 8, No. 3 (2009) 266-270.

[4] K. Zhang, Threshold proxy signature schemes, Proceedings of the First International Workshop on Information Security (1997) 282-290.

[5] W.D. Lin, J.K. Jan, A security personal learning tools using a proxy blind signature scheme, Proceedings of International Conference on Chinese Language Computing, Illinois, USA (July 2000) 273-277.

[6] G.K. Verma, A proxy blind signature scheme over braid groups, International Journal of Network Security, Vol. 9, No. 3 (2009) 214-217.

[7] A.K. Awasthil, S. Lal, ID-based ring signature and proxy ring signature schemes from bilinear pairings, International Journal of Network Security, Vol. 4, No. 2 (Mar 2007) 187-192.

[8] J. Li, T. H. Yuen, X. F. Chen, et al, Proxy ring signature: Formal definitions, efficient construction and new variant, Proceedings of International Conference of Computational Intelligence and Security, Vol. 2 (2006) 1259-1264.

[9] D. Boneh, E.J. Goh, K. Nissim, Evaluating 2-DNF formulas on cipher texts, In J. Kilian, editor, Proceedings of TCC 2005, number 3378 in LNCS, Springer-Verlag (Feb 2005) 325-341.

[10] R. Rivest, A. Shamir, Y. Tauman, How to leak a secret, AsiaCrypt'01, LNCS 2248, Springer-Verlag (2001) 552-565.

[11] D. Chaum, E. Heyst, Group signature, EUROCRYPT 1991, LNCS 547, Springer-Verlag (1991) 257-265.

[12] R. Rivest, A. Shamir, Y. Tauman, How to leak a secret: Theory and applications of ring signatures, Essays in Theoretical Computer Science: in Memory of Shimon Even, LNCS 3895, Springer-Verlag (2006) 164-186.
[13] A.K. Awasthil, S. Lal, A new proxy ring signature scheme, Proceeding of RMS 2004, Agra, INDIA (2004).
[14] H. Xiong, Z. Qin, F. Li, A Certificateless Proxy Ring Signature Schemewith Provable Security, International Journal of Network Security, Vol. 12, No. 2 (Mar 2011) 92-106.
[15] Y. Yu, C. Xu, X. Huang,Y. Mu, An efficient anonymous proxy signature scheme with provable security, Computer Standards & Interfaces 31 (2009) 348–353.
[16] E. Bresson, J. Stern, M. Szydlo, Threshold ring signatures and applications to ad-hoc groups, In CRYPTO 2002, Proceedings, Volume 2442 of Lecture Notes in Computer Science, Springer (2002) 465–480.
[17] S. Chow, S. Yiu, L. Hui, Efficient identity based ring signature, In ACNS 2005, Proceedings, Volume 3531 of Lecture Notes in Computer Science, Springer (2005).
[18] J. Li , Y. Wang, Universal designated verifier ring signature (proof) without random oracles, In Emerging Directions in Embedded and Ubiquitous Computing 2006, Proceedings, Volume 4097 of Lecture Notes in Computer Science, Springer (2006) 332–341.
[19] W. Baodian, Z. Fangguo, C. Xiaofeng, Ring Proxy Signatures, Journal of Electronics (China), Vol. 25 No. 1 (January 2008).
[20] F. Zhang, R. Safavi-Naini, C. Lin, New Proxy Signature, Proxy Blind Signature and Proxy Ring Signature Schemes from Bilinear Pairings, IACR Cryptology ePrint Archive 2003: 104 (2003).
[21] Y. Sun, C. Xu, Y. Yu, Y. Mu, Strongly unforgeable proxy signature scheme secure in the standard model, The Journal of Systems and Software 84 (2011) 1471–1479.
[22] H. Shacham, B. Waters, Efficient Ring Signatures Without Random Oracles, In T. Okamoto and X. Wang, eds., Proceedings of PKC 2007, Vol. 4450 of LNCS, Springer-Verlag (Apr 2007) 166–180.
[23] B. Waters, Efficient identity-based encryption without random oracles, In R. Cramer, editor, Proceedings of Eurocrypt 2005, Vol. 3494 of LNCS, Springer-Verlag (May 2005) 114- 127.
[24] A. Bender, J. Katz, R. Morselli, Ring signatures: Stronger definitions, and constructions without random oracles, Cryptology ePrint Archive, Report 2005/304 (2005), http://eprint.iacr.org
[25] A. Bender, J. Katz, R. Morselli, Ring signatures: Stronger definitions, and constructions without random oracles, In S. Halevi and T. Rabin, editors, Proceedings of TCC 2006, Volume 3876 of LNCS, Springer-Verlag (Mar 2006) 60-79.
[26] G. Fuchsbauer, D. Pointcheval, Anonymous Proxy Signatures, In The 6th Conference on Security in Communication Networks (SCN '08), Springer, Vol. 5229, Amalfi, Italy (2008) 201-217.
[27] X. Boyen, B. Waters, Compact group signatures without random oracles, In S. Vaudenay, editor, Proceedings of Eurocrypt 2006, Volume 4004 of LNCS, Springer-Verlag (May 2006) 427-444.