Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 373690, 10 pages
http://dx.doi.org/10.1155/2014/373690

Research Article

Certificateless Proxy Signature from RSA

Lunzhi Deng,[1] Jiwen Zeng,[2] and Yunyun Qu[1]

[1] *School of Mathematics and Computer Science, Guizhou Normal University, Guiyang 550001, China*
[2] *School of Mathematical Sciences, Xiamen University, Fujian 361005, China*

Correspondence should be addressed to Lunzhi Deng; denglunzhi@163.com

Received 31 January 2014; Revised 28 May 2014; Accepted 29 May 2014; Published 23 June 2014

Academic Editor: Kwok-Wo Wong

Although some good results were achieved in speeding up the computation of pairing function in recent years, it is still interesting to design efficient cryptosystems with less bilinear pairing operation. A proxy signature scheme allows a proxy signer to sign messages on behalf of an original signer within a given context. We propose a certificateless proxy signature (CLPS) scheme from RSA and prove its security under the strongest security model where the Type I/II adversary is a super Type I/II adversary.

1. Introduction

Public key cryptography is an important technique to realize network and information security. Traditional public key infrastructure requires a trusted certification authority to issue a certificate binding the identity and the public key of an entity. Hence, the problem of certificate management arises. To solve the problem, Shamir defined a new public key paradigm called identity-based public key cryptography [1]. However, identity-based public key cryptography needs a trusted PKG to generate a private key for an entity according to its identity. So we are confronted with the key escrow problem. Fortunately, the two problems in traditional public key infrastructure and identity-based public key cryptography can be prohibited by introducing certificateless public key cryptography (CLPKC) [2], which can be conceived as an intermediate between traditional public key infrastructure and identity-based cryptography.

1.1. Certificateless Cryptography. In 2003, Al-Riyami and Paterson [2] introduced the notion of certificateless public key cryptography. Its goal is to remove the key escrow property from identity-based cryptography and has attracted a great extent of attention lately [3–11]. Certificateless cryptography not only eliminates the key escrow property but also removes certificates. It lets a semitrusted KGC issue a user partial key to a user with respect to his/her identity. By possessing both the user partial key and a self-generated user secret key, the user is able to carry out predefined cryptographic operations. Typically there are two types of attacks to consider in certificateless cryptography. One is called KGC Attack in which the KGC is malicious and targets forge signatures from its knowledge about the user's partial key. The other one is called Key Replacement Attack in which the user's public/secret key pair could be replaced by a third party but this user's partial key issued by the KGC is not revealed. Au et al. [12] further investigated the types of malicious activities that the semitrusted KGC may be allowed to perform in practice and proposed a new strong security model called Malicious-but-Passive KGC Attack to replace original KGC Attack. A Malicious-but-Passive KGC may generate system parameters and the master key pair without following the scheme specification. Several certificateless signature schemes have been found vulnerable to this attack.

1.2. Cryptography from RSA. In 1985, Shamir [1] proposed the first identity-based signature scheme from the RSA primitive. In 1990, Guillou and Quisquater [13] proposed a similar RSA identity-based signature scheme, which is constructed from a zero-knowledge identification protocol. Herranz [14] proposed identity-based ring signatures from RSA whose security is based on the hardness of the RSA problem. After initial schemes, the following breakthrough result in the area of identity-based cryptography came in 2003, when Boneh and Franklin [15] designed an efficient identity-based public

key encryption scheme. In the design, they used as a tool bilinear pairings, a kind of maps which can be constructed on some elliptic curves. Since the appearance of this work, a lot of cryptography schemes have been proposed for encryption, signature, key agreement, and so forth and they all employ such bilinear pairings. However, it is still desirable to find cryptography schemes which do not need to employ bilinear pairings.

1.3. Proxy Signature. The concept of proxy signatures was first introduced by Mambo et al. [16]. Based on the delegation type, they classified proxy signature schemes into three types: full delegation, partial delegation, and delegation by warrant. In a full delegation scheme, the original signer's private key is given to the proxy signer. Hence, the proxy signer has the same signing right as the original signer. Obviously, such schemes are impractical and insecure for most of real-world settings. In a partial delegation scheme, a proxy signer has a new key, called proxy private key, which is different from the original's private key. Although proxy signatures generated by using proxy private key are different from the original signers standard signatures, the proxy signer is not limited on the range of messages he can sign. This weakness is eliminated in delegation by warrant schemes. One of the main advantages of the use of warrants is that it is possible to include any type of security policy (that specifies what kinds of messages are delegated and may contain other information, such as the identities of the original signer, the proxy signer, the delegation period, etc.) in the warrant to describe the restrictions under which the delegation is valid. Therefore, proxy signature scheme which uses the method of this approach attracts a great interest, and it is often expected that new proxy signature schemes will implement the functionality of warrants.

In order to adapt different situations, many proxy signature variants are produced, such as one-time proxy signature, proxy blind signature, and multiproxy signature. Since the proxy signature appears, it attracts many researchers' great attention. Using bilinear pairings, people proposed many new ID-based proxy signature (IBPS) schemes [17–31] and certificateless proxy signature (CLPS) [32–38] schemes. All the above schemes are very practical, but they are based on bilinear pairings and the pairing is regarded as the most expensive cryptography primitive. The relative computation cost of a pairing is much higher than that of the scalar multiplication over elliptic curve group. Therefore, CLPS scheme without less bilinear pairing operations would be more appealing in terms of efficiency.

1.4. Motivations and Our Contributions. Although some good results were achieved in speeding up the computation of pairing function in recent years, it is still interesting to design cryptographic scheme without pairing operations.

In this paper, we propose a certificateless proxy signature (CLPS) scheme, which has the following features.

(i) The proposed scheme is security under the strongest security model. Namely, in the scheme, the super Type I/II adversary can obtain the valid signatures for the replaced public key, without additional submission.

(ii) The proposed scheme not only enjoys a high security level but is also very efficient. The scheme does not need pairing operation. To the best of authors' knowledge, our scheme is the first certificateless proxy signature scheme from RSA.

2. Preliminaries

Definition 1. Let $N = pq$, where p and q are two k-bit prime numbers. Let b be a random prime number, greater than 2^l for some fixed parameter l, such that $\gcd(b, \varphi(N)) = 1$. Let Y be a random element in Z_N^*. We say that an algorithm $\mathscr{C}$ solves the RSA problem if it receives as input the tuple (N, b, Y) and outputs an element X such that $X^b = Y \bmod N$.

Definition 2. Given a generator g of a group G of prime order b, and an element $g^x \in G$, the discrete logarithm problem (DLP) is to compute x.

2.1. Model of Certificateless Proxy Signature Scheme. A certificateless proxy signature scheme consists of the following eight algorithms: setup, partial private key extraction, secret value setting, user public key generation, delegate, delegation verify, proxy sign, and proxy signature verify:

(i) *Setup*. This algorithm takes as input a security parameter k and returns params (system parameters) and a randomly chosen master secret key msk. After the algorithm is performed, the KGC publishes the system parameters params and keeps the master key msk secret.

(ii) *Partial Private Key Extract*. This algorithm takes as input params, msk, and an identity $\mathrm{ID} \in \{0,1\}^*$ of an entity and returns a partial private key D_{ID}. The KGC carries out the algorithm to generate the partial private key D_{ID} and sends D_{ID} to the corresponding owner ID via a secure channel.

(iii) *Secret Value Set.* This algorithm takes the params, an identity ID, as input and outputs a secret value t_{ID}. This algorithm is run by the identity ID for itself.

(iv) *User Public Key Generate*. This algorithm takes the params, an identity ID and the identity's secret value t_{ID} as input. It outputs the public key $\mathrm{PK}_{\mathrm{ID}}$ for the identity ID. This algorithm is run by the identity ID for itself.

(v) *Delegate.* This algorithm takes as input the params, original signer's full private key (t_o, D_o), a warrant m_w, and outputs the delegation π.

(vi) *Delegation Verify.* This algorithm takes as input params, π, and verifies whether π is a valid delegation from the original signer.

(vii) *Proxy Sign.* This algorithm takes as input the params, proxy signer's full private key (t_p, D_p), delegation π, a message m, and outputs the proxy signature σ.

(viii) *Proxy Signature Verify.* This algorithm takes as input the params, original signer's identity/public key $\mathrm{ID}_o/\mathrm{PK}_o$, proxy signer's identity/public key $\mathrm{ID}_p/\mathrm{PK}_p$, a proxy signature σ, and outputs 1 if the proxy signature is valid or 0 otherwise.

Definition 3. A certificateless proxy signature scheme (CLPS) is said to be existentially unforgeable against adaptive chosen message attacks (EUF-CLPS-CMA) if no polynomially bounded adversary has a nonnegligible advantage in the following two games against Type I and Type II adversaries.

Game I. Now we illustrate the first game performed between a challenger $\mathscr{C}$ and a Type I adversary $\mathscr{A}_I$ for a certificateless proxy signature scheme.

Initialization. $\mathscr{C}$ runs the setup algorithm to generate a master secret key msk and the public system parameters params. $\mathscr{C}$ keeps msk secret and gives params to $\mathscr{A}_I$. We should bear in mind that $\mathscr{A}_I$ does not know msk.

Queries. $\mathscr{A}_I$ performs a polynomially bounded number of queries. These queries may be made adaptively; that is, each query may depend on the answers to the previous queries.

(i) Create user: on inputting an identity $\mathrm{ID} \in \{0,1\}^*$, if ID has already been created, nothing is to be carried out. Otherwise, $\mathscr{C}$ runs the algorithms partial private key extract, secret value set, and user public key generate to obtain the partial private key D_{ID}, secret value t_{ID}, and public key $\mathrm{PK}_{\mathrm{ID}}$. In this case, ID is said to be created and $\mathrm{PK}_{\mathrm{ID}}$ is returned.

(ii) Partial private key extract: on inputting an identity ID, it returns the partial private key D_{ID} if ID has been created. Otherwise, returns 0.

(iii) Public key replace: on inputting an identity ID and a user public key $\mathrm{PK}_{\mathrm{ID}}$, the original user public key of ID is replaced with $\mathrm{PK}_{\mathrm{ID}}$ if ID has been created. Otherwise, no action will be taken.

(iv) Secret value set: on inputting an identity ID, it returns the corresponding user secret key t_{ID} if ID has been created. Otherwise, returns 0. Note that t_{ID} is the secret value associated with the original public key $\mathrm{PK}_{\mathrm{ID}}$. $\mathscr{A}_I$ cannot query the secret value for ID whose public key has been replaced.

(v) Delegate: when $\mathscr{A}_I$ submits original signer's identity/public key ID_o/P_o and a warrant m_w to the challenger, $\mathscr{C}$ responds by running the delegate algorithm on the warrant m_w and the original signer's full private key (t_o, D_o).

(vi) Proxy sign: when $\mathscr{A}_I$ submits a delegation π and a message m to the challenger, $\mathscr{C}$ responds by running the proxy sign algorithm on the delegation π, message m, and the proxy signer's full private key (t_p, D_p).

Forge. $\mathscr{A}_I$ outputs a tuple

$$(\pi^*, \mathrm{ID}_o, \mathrm{PK}_o) \quad \text{or} \quad \left(m^*, m_w^*, \sigma^*, \mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p\right). \tag{1}$$

$\mathscr{A}_I$ wins the game, if one of the following cases is satisfied:

(i) Case 1: The final output is $(\pi^*, \mathrm{ID}_o, \mathrm{PK}_o)$ and it satisfies

(1) π^* is a valid delegation.

(2) π^* is not generated from the delegation query on $(\mathrm{ID}_o, \mathrm{PK}_o)$.

(3) $\mathscr{A}_I$ does not query the original signer ID_o's partial private key.

(4) $\mathscr{A}_I$ cannot query the secret value for any identity if the corresponding public key has already been replaced.

(ii) Case 2: The final output is $(m^*, m_w^*, \sigma^*, \mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p)$ and it satisfies

(1) σ^* is a valid proxy signature.

(2) σ^* is not generated from the proxy signature query.

(3) The tuple $(\mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p, m_w^*)$ does not appear in delegation query.

(4) $\mathscr{A}_I$ does not query the original signer ID_o's partial private key.

(5) $\mathscr{A}_I$ cannot query the secret value for any identity if the corresponding public key has already been replaced.

(iii) Case 3: The final output is $(m^*, m_w^*, \sigma^*, \mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p)$ and it satisfies

(1) σ^* is a valid proxy signature.

(2) σ^* is not generated from the proxy signature query.

(3) $\mathscr{A}_I$ does not query the proxy signer ID_p's partial private key.

(4) $\mathscr{A}_I$ cannot query the secret value for any identity if the corresponding public key has already been replaced.

The advantage of $\mathscr{A}_I$ is defined as $\mathrm{Adv}_{\mathscr{A}_I}^{\text{EUF-CLPS-CMA}} = \Pr[\mathscr{A}_I \quad \text{win}]$.

Game II. A Type II adversary $\mathscr{A}_{II}$ plays the second game with a challenger $\mathscr{C}$ as follows.

Initialization. $\mathscr{C}$ runs the setup algorithm to obtain a master secret key msk and public system parameters params. $\mathscr{C}$ gives params and msk to $\mathscr{A}_{II}$. We should bear in mind that $\mathscr{A}_{II}$ know msk.

Queries. $\mathscr{A}_{II}$ may adaptively make a polynomially bounded number of queries as in Game I.

Forge. $\mathscr{A}_{II}$ outputs a tuple

$$(\pi^*, \mathrm{ID}_o, \mathrm{PK}_o) \quad \text{or} \quad \left(m^*, m_w^*, \sigma^*, \mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p\right). \tag{2}$$

$\mathcal{A}_{II}$ wins the game, if one of the following cases is satisfied

(i) Case 1: The final output is $(\pi^*, \mathrm{ID}_o, \mathrm{PK}_o)$ and it satisfies

(1) π^* is a valid delegation.
(2) π^* is not generated from the delegation query on $(\mathrm{ID}_o, \mathrm{PK}_o)$.
(3) $\mathcal{A}_{II}$ does not replace the original signer ID_o's public key.
(4) $\mathcal{A}_{II}$ does not query the original signer ID_o's secret value.
(5) $\mathcal{A}_{II}$ cannot query the secret value for any identity if the corresponding public key has already been replaced.

(ii) Case 2: The final output is $(m^*, m_w^*, \sigma^*, \mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p)$ and it satisfies

(1) σ^* is a valid proxy signature.
(2) σ^* is not generated from the proxy signature query.
(3) The tuple $(\mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p, m_w^*)$ does not appear in delegation query.
(4) $\mathcal{A}_{II}$ does not replace the original signer ID_o's public key.
(5) $\mathcal{A}_{II}$ does not query the original signer ID_o's secret value.
(6) $\mathcal{A}_{II}$ cannot query the secret value for any identity if the corresponding public key has already been replaced.

(iii) Case 3: The final output is $(m^*, m_w^*, \sigma^*, \mathrm{ID}_o, \mathrm{PK}_o, \mathrm{ID}_p, \mathrm{PK}_p)$ and it satisfies

(1) σ^* is a valid proxy signature.
(2) σ^* is not generated from the proxy signature query.
(3) $\mathcal{A}_{II}$ does not replace the proxy signer ID_p's public key.
(4) $\mathcal{A}_{II}$ does not query the proxy signer ID_p's secret value.
(5) $\mathcal{A}_{II}$ cannot query the secret value for any identity if the corresponding public key has already been replaced.

The advantage of $\mathcal{A}_{II}$ is defined as $\mathrm{Adv}_{\mathcal{A}_{II}}^{\text{EUF-CLPS-CMA}} = \Pr[\mathcal{A}_{II} \text{ win}]$.

3. Our Certificateless Proxy Signature Scheme

(i) Setup: given the security parameter of the system k, the KGC generates two random k-bit prime numbers p and q. Then it computes $N = pq$. For some fixed parameter l (for example $l = 160$), it chooses at random a prime number b satisfying $2^l < b < 2^{l+1}$ and $\gcd(b, \varphi(N)) = 1$. Then it chooses group G of prime order b, a generator g of G, and computes $a = b^{-1} \bmod \varphi(N)$. Furthermore, KGC chooses five cryptographic hash functions described as follows: $H_0 : \{0,1\}^* \to Z_N^*$, $H_1, H_2, H_3, H_4 : \{0,1\}^* \to Z_b^*$. Finally, KGC outputs the set of public parameters: $\text{params} = \{N, b, G, g, H_1, H_2, H_3\}$; the master secret key is (p, q, a).

(ii) Partial private key extract: for an identity $\mathrm{ID} \in \{0,1\}^*$ his private key is $D_{\mathrm{ID}} = Q_{\mathrm{ID}}^a$, $Q_{\mathrm{ID}} = H_0(\mathrm{ID})$. The KGC sends D_{ID} to the user ID via a secure channel.

(iii) Set secret value: the user with identity $\mathrm{ID} \in \{0,1\}^*$ randomly chooses $t_{\mathrm{ID}} \in Z_b^*$.

(iv) User public key generation: the user with identity $\mathrm{ID} \in \{0,1\}^*$ computes his public key $P_{\mathrm{ID}} = g^{t_{\mathrm{ID}}}$.

(v) Delegate: m_w is the warrant consisting of the identities/public keys of original signer and proxy signer, the delegation duration, and so on. On inputting the warrant m_w, the original signer, whose identity/public key is ID_o/P_o, performs the following steps.

(vi) Randomly selects $c \in Z_b^*$, $A \in Z_N^*$, computes $T_1 = g^c$, $T_2 = A^b \bmod N$, $h_1 = H_1(m_w, T_1, T_2)$, $h_2 = H_2(m_w, T_1, T_2)$.

(vii) Computes $r = c + t_o h_1 \bmod b$, $R = AD_o^{h_2} \bmod N$.

(viii) Outputs $\pi = (m_w, T_1, T_2, r, R)$ as the delegation.

(ix) Delegation verify: to verify a delegation $\pi = (m_w, T_1, T_2, r, R)$ for an identity/public key ID_o/P_o, the verifier performs the following steps.

Computes $h_1 = H_1(m_w, T_1, T_2)$, $h_2 = H_2(m_w, T_1, T_2)$.

Checking whether $g^r = T_1 P_o^{h_1}$, $R^b = T_2 Q_o^{h_2} \bmod N$, if both of equalities hold, accept the delegation. Otherwise, reject.

(x) Proxy sign: for a message m, the proxy signer (whose identity/public key is ID_p/P_p) who owns the delegation $\pi = (m_w, T_1, T_2, r, R)$ does the following.

(1) Randomly selects $d \in Z_b^*$, $B \in Z_N^*$, computes $S_1 = g^d$, $S_2 = B^b \bmod N$, $k_1 = H_3(m, m_w, T_1, T_2, S_1, S_2)$, $k_2 = H_4(m, m_w, T_1, T_2, S_1, S_2)$.
(2) Computes $z = r + d + t_p k_1$, $Z = RBD_p^{k_2} \bmod N$.
(3) Outputs the signature $\sigma = (m, m_w, T_1, T_2, S_1, S_2, z, Z)$.

(xi) Proxy signature verify: to verify the validity of a proxy signature (where the original singer's identity/public key is ID_o/P_o, the proxy singer's identity/public key is ID_p/P_p), a verifier first checks whether the original signer and proxy signer conform to m_w and then performs the following steps.

(1) Computes $h_1 = H_1(m_w, T_1, T_2)$, $h_2 = H_2(m_w, T_1, T_2)$.
(2) Computes $k_1 = H_3(m, m_w, T_1, T_2, S_1, S_2)$, $k_2 = H_4(m, m_w, T_1, T_2, S_1, S_2)$.

(3) Checking whether $g^z = T_1 S_1 P_o^{h_1} P_p^{k_1}$, $Z^b = T_2 S_2 Q_o^{h_2} Q_p^{k_2} \bmod N$, if both of equalities hold, outputs 1. Otherwise, outputs 0.

(xii) On correctness, we have

$$g^z = g^{r+d+t_p k_1} = g^r g^d g^{t_p k_1} = T_1 P_o^{h_1} S_1 P_p^{k_1}$$
$$Z^b = \left(RBD_p^{k_2}\right)^b = R^b B^b D_p^{k_2 b} = T_2 S_2 Q_o^{h_2} Q_p^{k_2}. \quad (3)$$

4. Security Results of Scheme 1

Theorem 4. *In the random oracle model, if there is an adversary $\mathscr{A}_I$ that can win the EUF-CLPS-CMA Game I with advantage ε and within time T, after making at most q_{H_0} H_0 queries, q_{H_1} H_1 queries, q_{H_2} H_2 queries, q_{H_3} H_3 queries, q_{H_4} H_4 queries, q_U create user queries, q_K partial private key extraction queries, q_S set secret value queries, q_R user public key replacement queries, q_D delegation queries, and q_P proxy signature queries, the RSA problem can be solved with probability ε/q_{H_0} within time $T + (q_{H_0} + 2q_D + 2q_P)T_m + (q_U + q_D + q_P)T_e$, where T_m denotes the time for a modular operation and T_e denotes the time for a exponentiation in G.*

Proof. Suppose the challenger $\mathscr{C}$ receives a random instance (N, b, Y) of the RSA problem and has to find an element $X \in Z_N^*$ such that $X^b = Y$. $\mathscr{C}$ will run $\mathscr{A}_I$ as a subroutine and act as $\mathscr{A}_I$'s challenger in the EUF-CLPS-CMA game I. □

Setup. At the beginning of the game, $\mathscr{C}$ runs the setup program with the parameter k and gives $\mathscr{A}$ the system parameters: params = $\{N, b, G, g, H_0, H_1, H_2, H_3, H_4\}$.

Queries. Without loss of generality, we assume that all the queries are distinct and $\mathscr{A}_I$ will make H_0 query and create user query for ID before ID is used in any other queries.

(i) H_0 queries: $\mathscr{C}$ maintains the list L_0 of tuple (ID_i, A_i). The list is initially empty. When $\mathscr{A}_I$ makes a query $H_0(\mathrm{ID}_i)$, $\mathscr{C}$ responds as follows.

At the jth H_0 query, $\mathscr{C}$ sets $H_0(\mathrm{ID}^*) = Y$. For $i \neq j$, $\mathscr{C}$ randomly picks a value $A_i \in Z_N^*$ and sets $H_0(\mathrm{ID}_i) = A_i^b$. Then, the query and the answer will be stored in the list L_0.

(ii) H_1 queries: $\mathscr{C}$ maintains the list L_1 of tuple (α_i, h_i). The list is initially empty. When $\mathscr{A}_I$ makes a query $H_1(\alpha_i)$, $\mathscr{C}$ randomly picks a value $h_i \in Z_b^*$ and sets $H_1(\alpha_i) = h_i$; the query and the answer will then be stored in the list L_1.

(iii) H_2 queries: $\mathscr{C}$ maintains the list L_2 of tuple (α_i, h_i). The list is initially empty. When $\mathscr{A}_I$ makes a query $H_2(\alpha_i)$, $\mathscr{C}$ randomly picks a value $h_i \in Z_b^*$ and sets $H_2(\alpha_i) = h_i$; the query and the answer will then be stored in the list L_2.

(iv) H_3 queries: $\mathscr{C}$ maintains the list L_3 of tuple (β_i, k_i). The list is initially empty. When $\mathscr{A}_I$ makes a query $H_3(\beta_i)$, $\mathscr{C}$ randomly picks a value $k_i \in Z_b^*$ and sets $H_3(\beta_i) = k_i$; the query and the answer will then be stored in the list L_3.

(v) H_4 queries: $\mathscr{C}$ maintains the list L_4 of tuple (β_i, k_i). The list is initially empty. When $\mathscr{A}_I$ makes a query $H_4(\beta_i)$, $\mathscr{C}$ randomly picks a value $k_i \in Z_b^*$ and sets $H_4(\beta_i) = k_i$; the query and the answer will then be stored in the list L_4.

(vi) Create user queries: $\mathscr{C}$ maintains the list L_U of tuple $(\mathrm{ID}_i, t_i, D_i, P_i)$. $\mathscr{A}_I$ makes creating user query for identity ID_i and $\mathscr{C}$ first makes query $H_0(\mathrm{ID}_i)$ and gets (ID_i, A_i) from list L_0, then randomly chooses $t_i \in Z_b^*$, sets $P_i = g^{t_i}$. If $\mathrm{ID}_i = \mathrm{ID}^*$, $\mathscr{C}$ sets $D_i = 0$, otherwise sets $D_i = A_i$. Then it sends the P_i to $\mathscr{A}_I$; the $(\mathrm{ID}_i, t_i, D_i, P_i)$ will be stored in the list L_U.

(vii) Partial private key extract: $\mathscr{C}$ maintains the list L_K of tuple (ID_i, D_i). $\mathscr{A}_I$ makes partial private key extraction query for identity ID_i. If $\mathrm{ID}_i = \mathrm{ID}^*$, $\mathscr{C}$ fails and stops. Otherwise, $\mathscr{C}$ finds the tuple $(\mathrm{ID}_i, t_i, D_i, P_i)$ in list L_U and responds with the partial private key D_i; the (ID_i, D_i) will be stored in the list L_K.

(viii) User public key replace: $\mathscr{C}$ maintains the list L_R of tuple $(\mathrm{ID}_i, P_i, P_i')$. $\mathscr{A}_I$ makes user public key replacement request for identity ID_i with a new valid public key value P_i'. $\mathscr{C}$ replaces the current public key value P_i with the value P_i' and tuple $(\mathrm{ID}_i, P_i, P_i')$ will be stored in the list L_R.

(ix) Set secret value: $\mathscr{C}$ maintains the list L_S of tuple (ID_i, t_i). $\mathscr{A}_I$ makes setting secret value query for identity ID_i. $\mathscr{C}$ finds the tuple $(\mathrm{ID}_i, t_i, D_i, P_i)$ in list L_U and responds with the secret value t_i; the (ID_i, t_i) will be stored in the list L_S. (Note: $\mathscr{A}_I$ cannot query the secret value for ID whose public key has been replaced.)

(x) Delegate: $\mathscr{A}_I$ submits ID_o/P_o, ID_p/P_p, and m_w to challenger. $\mathscr{C}$ outputs a delegation as follows.

If $\mathrm{ID}_o \neq \mathrm{ID}^*$ and $\mathrm{ID}_o \notin L_R$, $\mathscr{C}$ gives a delegation by calling the delegate algorithm. Otherwise, $\mathscr{C}$ does as follows.

(1) Randomly selects $A \in Z_N^*$ and $r, h_1, h_2 \in Z_b^*$.
(2) Computes $T_1 = P_o^{r-h_1}$, $T_2 = A^b Q_o^{-h_2}$ and $R = A$.
(3) Stores the relations $h_1 = H_1(m_w, T_1, T_2)$ and $h_2 = H_2(m_w, T_1, T_2)$. If collision occurs, repeats the step (1)–(3).
(4) Outputs $\pi = (m_w, T_1, T_2, r, R)$ as the delegation.

(xi) Proxy sign: $\mathscr{A}_I$ submits a delegation $\pi = (m_w, T_1, T_2, r, R)$ message m to the challenger. $\mathscr{C}$ outputs a certificateless proxy signature as follows (where original signer's identity/public key is ID_o/P_o, proxy signer's identity/public key is ID_p/P_p).

If $\mathrm{ID}_p \neq \mathrm{ID}^*$ and $\mathrm{ID}_p \notin L_R$, $\mathscr{C}$ gives a signature by calling the proxy sign algorithm. Otherwise, $\mathscr{C}$ does as follows.

(1) Randomly selects $B \in Z_N^*$ and $y, k_1, k_2 \in Z_b^*$.
(2) Computes $S_1 = P_p^{y-k_1}$, $S_2 = B^b Q_p^{-k_2}$, $z = r + y$, and $Z = RB$.
(3) Stores the relations $k_1 = H_3(m, m_w, T_1, T_2, S_1, S_2)$ and $k_2 = H_4(m, m_w, T_1, T_2, S_1, S_2)$. If collision occurs, repeats the step (1)–(3).
(4) Outputs the proxy signature $\sigma = (m, m_w, T_1, T_2, S_1, S_2, z, Z)$.

Forge. $\mathscr{A}_I$ outputs a tuple

$$\{\pi^* = (m_w, T_1, T_2, r, R), \mathrm{ID}_o, P_o\}$$
$$\text{or} \quad \{\sigma^* = (m, m_w, T_1, T_2, S_1, S_2, z, Z), \mathrm{ID}_o, P_o, \mathrm{ID}_p, P_p\}. \tag{4}$$

If $\mathscr{A}_I$'s output satisfies none of the three cases in EUF-CLPS-CMA game I, $\mathscr{C}$ aborts; Otherwise, $\mathscr{C}$ can solve the RSA problem as follows.

Case 1. The final output is $\{\pi^* = (m_w, T_1, T_2, r, R), \mathrm{ID}_o, P_o\}$ and the output satisfies the requirement of Case 1 as defined in EUF-CLPS-CMA game I. In fact, π^* is the signature for m_w by ID_o. By the forking lemma for generic signature scheme, for the resemble construction we can get two delegations: (m_w, T_1, T_2, r, R) and (m_w, T_1, T_2, r, R'), where $h_1 = h_1' = H_1(m_w, T_1, T_2)$, $h_2 = H_2(m_w, T_1, T_2)$, $h_2' = H_2'(m_w, T_1, T_2)$, and $h_2 \neq h_2'$. If $\mathrm{ID}_o = \mathrm{ID}^*$, we can solve RSA problem as follows. The relation becomes $(R'R^{-1})^b = Y^{h_2'-h_2} \bmod N$. Since $h_2, h_2' \in Z_b$, we have that $|h_2' - h_2| < b$. By the element b is a prime number. So $\gcd(b, h_2' - h_2) = 1$. This means that there exist two integers c and d such that $cb + d(h_2' - h_2) = 1$. Finally, the value $X = (R'R^{-1})^d Y^c \bmod N$ is the solution of the given instance of the RSA problem. In effect, we have $X^b = (R'R^{-1})^{bd} Y^{bc} = Y^{d(h_2'-h_2)} Y^{bc} = Y^{cb+d(h_2'-h_2)} = Y$.

Probability of Success. The probability that $\mathscr{C}$ does not fail during the queries is $(q_{H_0} - q_K)/q_{H_0}$. The probability that $\mathrm{ID}_o = \mathrm{ID}^*$ is $1/(q_{H_0} - q_K)$. So the combined probability is $((q_{H_0} - q_K)/q_{H_0}) \cdot (1/(q_{H_0} - q_K)) = 1/q_{H_0}$. Therefore, the probability of $\mathscr{C}$ to solve the RSA problem is ϵ/q_{H_0}.

Case 2. The final output is $\{\sigma^* = (m, m_w, T_1, T_2, S_1, S_2, z, Z), \mathrm{ID}_o, P_o, \mathrm{ID}_p, P_p\}$ and the output satisfies the requirement of Case 2 as defined in EUF-IBPS-CMA game I. By the forking lemma for generic signature scheme, for the resemble construction we can get two proxy signatures: $(m, m_w, T_1, T_2, S_1, S_2, z, Z)$ and $(m, m_w, T_1, T_2, S_1, S_2, z, Z')$, where

$$\begin{aligned} k_1 = k_1' &= H_3(m, m_w, T_1, T_2, S_1, S_2), \\ k_2 = k_2' &= H_4(m, m_w, T_1, T_2, S_1, S_2) \\ h_1 = h_1' &= H_1(m_w, T_1, T_2), \\ h_2 &= H_2(m_w, T_1, T_2), \\ h_2' &= H_2'(m_w, T_1, T_2), \\ h_2 &\neq h_2'. \end{aligned} \tag{5}$$

If $\mathrm{ID}_o = \mathrm{ID}^*$, we can solve RSA problem as follows. The relation becomes $(Z'Z^{-1})^b = Y^{h_2'-h_2} \bmod N$. Since $h_2, h_2' \in Z_b$, we have that $|h_2' - h_2| < b$. By the element b is a prime number. So it holds $\gcd(b, h_2' - h_2) = 1$. This means that there exist two integers c and d such that $cb + d(h_2' - h_2) = 1$. Finally, the value $X = (Z'Z^{-1})^d Y^c \bmod N$ is the solution of the given instance of the RSA problem. In effect, we have

$$X^b = (Z'Z^{-1})^{bd} Y^{bc} = Y^{d(h_2'-h_2)} Y^{bc} = Y^{cb+d(h_2'-h_2)} = Y. \tag{6}$$

Probability of success is the same as the probability in Case 1.

Case 3. The final output is $\{\sigma^* = (m, m_w, T_1, T_2, S_1, S_2, z, Z), \mathrm{ID}_o, P_o, \mathrm{ID}_p, P_p\}$ and the output satisfies the requirement of Case 3 as defined in EUF-IBPS-CMA game I. By the forking lemma for generic signature scheme, for the resemble construction we can get two proxy signatures: $(m, m_w, T_1, T_2, S_1, S_2, z, Z)$ and $(m, m_w, T_1, T_2, S_1, S_2, z, Z')$, where

$$\begin{aligned} h_1 = h_1' &= H_1(m_w, T_1, T_2), \\ h_2 = h_2' &= H_2(m_w, T_1, T_2), \\ k_1 = k_1' &= H_3(m, m_w, T_1, T_2, S_1, S_2) \\ k_2 &= H_4(m, m_w, T_1, T_2, S_1, S_2), \\ k_2' &= H_4'(m, m_w, T_1, T_2, S_1, S_2), \\ k_2 &\neq k_2'. \end{aligned} \tag{7}$$

If $\mathrm{ID}_p = \mathrm{ID}^*$, we can solve RSA problem as follows. The relation becomes $(Z'Z^{-1})^b = Y^{k_2'-k_2} \bmod N$. Since $k_2, k_2' \in Z_b$, we have that $|k_2' - k_2| < b$. By the element b is a prime number. So it holds $\gcd(b, k_2' - k_2) = 1$. This means that there exist two integers c and d such that $cb + d(k_2' - k_2) = 1$. Finally, the value $X = (Z'Z^{-1})^d Y^c \bmod N$ is the solution of the given instance of the RSA problem. In effect, we have

$$X^b = (Z'Z^{-1})^{bd} Y^{bc} = Y^{d(k_2'-k_2)} Y^{bc} = Y^{cb+d(k_2'-k_2)} = Y. \tag{8}$$

Probability of success is the same as the probability in Case 1.

Theorem 5. *In the random oracle model, if there is an adversary $\mathscr{A}_{II}$ that can win the EUF-CLPS-CMA game II with advantage ε and within time T, after making at most q_{H_0} H_0 queries, q_{H_1} H_1 queries, q_{H_2} H_2 queries, q_{H_3} H_3 queries, q_{H_4} H_4 queries, q_U create user queries, q_K partial private key extraction queries, q_S set secret value queries, q_R user public key replacement queries, q_D delegate queries, q_P proxy signature*

queries, the discrete logarithm problem DLP can be solved with probability ε/q_U *within time* $T + (q_{H_0} + 2q_D + 2q_P)T_m + (q_U + q_D + q_P)T_e$, *where* T_m *denote the time for a modular operation and* T_e *denote the time for an exponentiation in* G.

Proof. Suppose the challenger $\mathscr{C}$ receives a random instance (g^x, g) of the DLP and has to compute the value of x. $\mathscr{C}$ will run $\mathscr{A}_{II}$ as a subroutine and act as $\mathscr{A}_{II}$'s challenger in the EUF-CLPS-CMA game II. □

Setup. At the beginning of the game, $\mathscr{C}$ runs the setup program with the parameter k and gives $\mathscr{A}$ the system parameters: params $= \{N, b, G, g, H_0, H_1, H_2, H_3, H_4\}$ and master secret key (p, q, a).

Queries. Without loss of generality, we assume that all the queries are distinct and $\mathscr{A}_{II}$ will make H_0 query and create user query for ID before ID is used in any other queries.

(i) H_0 queries: $\mathscr{C}$ maintains the list L_0 of tuple (ID_i, A_i). The list is initially empty. When $\mathscr{A}_{II}$ makes a query $H_0(\mathrm{ID}_i)$, $\mathscr{C}$ randomly picks a value $A_i \in Z_N^*$ and sets $H_0(\mathrm{ID}_i) = A_i$. Then, (ID_i, A_i) will be stored in the list L_0.

(ii) H_1, H_2, H_3, H_4 queries: same as that in the proof of Theorem 4.

(iii) Create user: $\mathscr{C}$ maintains the list L_U of tuple $(\mathrm{ID}_i, t_i, D_i, P_i)$. The list is initially empty. When $\mathscr{A}_{II}$ makes creating user query for ID, $\mathscr{C}$ responds as follows.

> At the jth create user query, $\mathscr{C}$ first makes query $H_0(\mathrm{ID}^*)$, gets $(\mathrm{ID}^*, A_{\mathrm{ID}^*})$ from list L_0, sets $D_{\mathrm{ID}^*} = A^a_{\mathrm{ID}^*}$ and $P_{\mathrm{ID}^*} = g^x$. $i \neq j$, $\mathscr{C}$ first makes query $H_0(\mathrm{ID}_i)$, gets (ID_i, A_i) from list L_0, sets $D_i = A_i^a$, then, randomly chooses $t_i \in Z_b^*$, sets $P_i = g^{t_i}$, then sends P_i to the $\mathscr{A}_{II}$; the query and the answer will be stored in the list L_U.

(iv) Partial private key extract: Since $\mathscr{A}_{II}$ knows master secret key (p, q, a), he can compute partial private key for any identity by himself. Hence, $\mathscr{C}$ does not need making partial private key query.

(v) User public key replace: $\mathscr{C}$ maintains the list L_R of tuple $(\mathrm{ID}_i, P_i, P_i')$. $\mathscr{A}_{II}$ makes user public key replacement request for identity ID_i with a new valid public key value P_i'. $\mathscr{C}$ replaces the current public key value P_i with the value P_i' and tuple $(\mathrm{ID}_i, P_i, P_i')$ will be stored in the list L_R.

(vi) Set secret value: $\mathscr{C}$ maintains the list L_S of tuple (ID_i, t_i). $\mathscr{A}_{II}$ makes partial private key query for identity ID_i. If $\mathrm{ID}_i = \mathrm{ID}^*$, $\mathscr{C}$ fails and stops. Otherwise, $\mathscr{C}$ finds the tuple $(\mathrm{ID}_i, t_i, D_i, P_i)$ in list L_U and responds with the secret value t_i; the (ID_i, t_i) will be stored in the list L_S. (Note: $\mathscr{A}_{II}$ cannot query the secret value for ID whose public key has been replaced.)

(vii) Delegate and proxy sign: Same as that in the proof of Theorem 4.

Forge. $\mathscr{A}_{II}$ outputs a tuple

$$\begin{aligned}&\{\pi^* = (m_w, T_1, T_2, r, R), \mathrm{ID}_o, P_o\}\\ &\text{or } \{\sigma^* = (m, m_w, T_1, T_2, S_1, S_2, z, Z), \mathrm{ID}_o, P_o, \mathrm{ID}_p, P_p\}.\end{aligned} \tag{9}$$

If $\mathscr{A}_{II}$'s output satisfies none of the three cases in EUF-CLPS-CMA game II, $\mathscr{C}$ aborts; otherwise, $\mathscr{C}$ can solve the DLP in G as follows.

Case 1. The final output is $\{\pi^* = (m_w, T_1, T_2, r, R), \mathrm{ID}_o, P_o\}$ and the output satisfies the requirement of Case 1 as defined in EUF-CLPS-CMA game II. In fact, π^* is the signature for m_w by ID_o. By the forking lemma for generic signature scheme, for the resemble construction we can get two delegations: (m_w, T_1, T_2, r, R) and (m_w, T_1, T_2, r', R), where

$$\begin{aligned}h_1 &= H_1(m_w, T_1, T_2),\\ h_1' &= H_1'(m_w, T_1, T_2),\\ h_2 = h_2' &= H_2(m_w, T_1, T_2),\\ h_1 &\neq h_1'.\end{aligned} \tag{10}$$

If $\mathrm{ID}_o = \mathrm{ID}^*$, we can solve DLP as follows: $x = (r - r')(h_1 - h_1')^{-1} \bmod b$.

Probability of Success. The probability that $\mathscr{C}$ does not fail during the queries is $(q_U - q_S)/q_U$. The probability that $\mathrm{ID}_o = \mathrm{ID}^*$ is $1/(q_U - q_S)$. So the combined probability is $((q_U - q_S)/q_U) \cdot (1/(q_U - q_S)) = 1/q_U$. Therefore, the probability of $\mathscr{C}$ to solve the DLP is ϵ/q_U.

Case 2. The final output is $\{\sigma^* = (m, m_w, T_1, T_2, S_1, S_2, z, Z), \mathrm{ID}_o, P_o, \mathrm{ID}_p, P_p\}$ and the output satisfies the requirement of Case 2 as defined in EUF-CLPS-CMA game II. By the forking lemma for generic signature scheme, for the resemble construction we can get two proxy signatures: $(m, m_w, T_1, T_2, S_1, S_2, z', Z)$ and $(m, m_w, T_1, T_2, S_1, S_2, z', Z)$, where

$$\begin{aligned}h_1 &= H_1(m_w, T_1, T_2),\\ h_1' &= H_1'(m_w, T_1, T_2),\\ h_2 = h_2' &= H_2(m_w, T_1, T_2),\\ h_1 &\neq h_1'\\ k_1 = k_1' &= H_3(m, m_w, T_1, T_2, S_1, S_2),\\ k_2 = k_2' &= H_4(m, m_w, T_1, T_2, S_1, S_2).\end{aligned} \tag{11}$$

If $\mathrm{ID}_o = \mathrm{ID}^*$, we can solve DLP as follows: $x = (z - z')(h_1 - h_1')^{-1} \bmod b$.

Probability of success is same as the probability in Case 1.

Case 3. The final output is $\{\sigma^* = (m, m_w, T_1, T_2, S_1, S_2, z, Z), \mathrm{ID}_o, P_o, \mathrm{ID}_p, P_p\}$ and the output satisfies the requirement

of Case 3 as defined in EUF-CLPS-CMA game II. By the forking lemma for generic signature scheme, for the resemble construction we can get two proxy signatures: $(m, m_w, T_1, T_2, S_1, S_2, z, Z)$ and $(m, m_w, T_1, T_2, S_1, S_2, z', Z)$, where

$$\begin{aligned} h_1 = h_1' &= H_1(m_w, T_1, T_2), \\ h_2 = h_2' &= H_2(m_w, T_1, T_2), \\ k_2 = k_2' &= H_4(m, m_w, T_1, T_2, S_1, S_2) \\ k_1 &= H_3(m, m_w, T_1, T_2, S_1, S_2), \\ k_1' &= H_3'(m, m_w, T_1, T_2, S_1, S_2), \\ k_1 &\neq k_1'. \end{aligned} \tag{12}$$

If $\mathrm{ID}_p = \mathrm{ID}^*$, we can solve DLP as follows: $x = (z - z')(k_1 - k_1')^{-1} \bmod b$.

Probability of success is same as the probability in Case 1.

5. Efficiency

Although some good results were achieved in speeding up the computation of pairing function in recent years, it is still desirable to find cryptography schemes which do not need to employ bilinear pairings. In this section, we compare the performance of our scheme with several CLPS schemes in Table 2; we define some notations as follows.

P: a pairing operation.

E_G a pairing-based scalar multiplication operation.

E_S: a scalar multiplication operation.

E_N: modular exponent in Z_N.

Cao et al. [39] obtained the running time for cryptographic operations through a PIV 3 GHZ processor with 512 M bytes memory and the Windows XP operating system. For the pairing-based scheme, to achieve the 1024-bit RSA level security, a supersingular curve E over a finite field F_p, with $p = 512$ bits and a large prime order $q = 160$ bits, was used. For the ECC-based schemes, to achieve the same security level, the ECC group on Koblitz elliptic curve $y^2 = x^3 + ax^2 + b$ was used which is defined on $F_{2^{163}}$ with $a = 1$ and b is a 163-bit random prime. The running times are listed in Table 1.

To evaluate the computation efficiency of different schemes, we use the simple method from [39]. For example, in Li et al. [35] scheme, eleven pairing operations and seven pairing-based scalar multiplication operation are needed. So the resulting computation time is $20.01 \times 11 + 6.38 \times 7 = 265.87$. The detailed comparison results of several different CLPS schemes are illustrated in Table 2.

Table 1: Cryptographic operation time (in milliseconds).

P	E_G	E_S	E_N
20.01	6.38	0.83	11.20

Table 2: Comparison of several CLPS schemes.

Scheme	Delegate and verify	Sign and verify	Execution time
Li et al. [35]	$4P + 3E_G$	$7P + 4E_G$	265.87
Lu et al. [34]	$4P + 3E_G$	$7P + 4E_G$	265.87
Choi and Lee [32]	$4P + 3E_G$	$3P + 3E_G$	179.05
Seo et al. [36]	$3P + 3E_G$	$4P + 4E_G$	185.43
Zhang et al. [38]	$4P + 3E_G$	$5P + 2E_G$	212.89
Ours	$3E_S + 4E_N$	$3E_S + 4E_N$	94.58

6. Conclusion

We proposed a certificateless proxy signature scheme and prove that our scheme is unforgeable under the strongest security model where the Type I/II adversary is a super Type I/II adversary. The analysis shows our scheme is more efficient than the related schemes. To the best of authors' knowledge, our scheme is the first certificateless proxy signature scheme from RSA. Due to the good properties of our schemes, it is very useful for practical application.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors are grateful to the anonymous referees for their helpful comments and suggestions. This research is supported by the National Natural Science Foundation of China (no. 11261060), the Dr. Research Foundation of Guizhou Normal University of Guizhou Province, China, under Grant 2013, and the Science and Technology Foundation of Guizhou Province, China, under Grant LKS[2013]02.

References

[1] A. Shamir, "Identity-based cryptosystem and signature scheme," in *Advances in Cryptology*, vol. 196 of *Lecture Notes in Computer Science*, pp. 47–53, Springer, Berlin, Germany, 1985.

[2] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Advances in Cryptology—Asiacrypt 2003*, C. S. Laih, Ed., vol. 2894 of *Lecture Notes in Computer Science*, pp. 452–473, Springer, Berlin, Germany, 2003.

[3] H. Du and Q. Wen, "Efficient and provably-secure certificateless short signature scheme from bilinear pairings," *Computer Standards and Interfaces*, vol. 31, no. 2, pp. 390–394, 2009.

[4] S. Duan, "Certificateless undeniable signature scheme," *Information Sciences*, vol. 178, no. 3, pp. 742–755, 2008.

[5] B. C. Hu, D. S. Wong, Z. Zhang, and X. Deng, "Key replacement attack against a generic construction of certificateless signature," in *Proceedings of the 11th Australasian Conference on Information Security and Privacy (ACISP '06)*, vol. 4058 of *Lecture Notes in Computer Science*, pp. 235–246, Springer, Melbourne, Australia, July 2006.

[6] X. Huang, W. Susilo, Y. Mu, and F. Zhang, "On the security of certificateless signature schemes from Asiacrypt 2003," in *Proceedings of the 4th International Conference on Cryptology and Network Security (CANS '05)*, vol. 3810 of *Lecture Notes in Computer Science*, pp. 13–25, Springer, Xiamen, China, December 2005.

[7] Y. Long and K. Chen, "Certificateless threshold cryptosystem secure against chosen-ciphertext attack," *Information Sciences*, vol. 177, no. 24, pp. 5620–5637, 2007.

[8] K. A. Shim, "Breaking the short certificateless signature scheme," *Information Sciences*, vol. 179, no. 3, pp. 303–306, 2009.

[9] F. Wang and Y. Zhang, "A new provably secure authentication and key agreement mechanism for SIP using certificateless public-key cryptography," *Computer Communications*, vol. 31, no. 10, pp. 2142–2149, 2008.

[10] L. Wang, Z. Cao, X. Lia, and H. Qian, "Simulatability and security of certificateless threshold signatures," *Information Sciences*, vol. 177, no. 6, pp. 1382–1394, 2007.

[11] Z. Zhang, D. S. Wong, J. Xu, and D. Feng, "Certificateless public-key signature: security model and efficient construction," in *Proceedings of the 4th International Conference on Applied Cryptography and Network Security (ACNS '06)*, vol. 3989 of *Lecture Notes in Computer Science*, pp. 293–308, Springer, Singapore, June 2006.

[12] M. H. Au, J. Chen, J. K. Liu, Y. Mu, D. S. Wong, and G. Yang, "Malicious KGC attacks in certificateless cryptography," in *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security (ASIACCS '07)*, pp. 302–311, Singapore, March 2007.

[13] L. C. Guillou and J. J. Quisquater, "A paradoxical identity-based signature scheme resulting from zero-knowledge," in *Advances in Cryptology—CRYPTO' 88*, vol. 403 of *Lecture Notes in Computer Science*, pp. 216–231, Springer, New York, NY, USA, 1990.

[14] J. Herranz, "Identity-based ring signatures from RSA," *Theoretical Computer Science*, vol. 389, no. 1-2, pp. 100–117, 2007.

[15] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[16] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: delegation of the power to sign messages," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 79, no. 9, pp. 1338–1353, 1996.

[17] A. Boldyreva, A. Palacio, and B. Warinschi, "Secure proxy signature schemes for delegation of signing rights," *Journal of Cryptology*, vol. 25, no. 1, pp. 57–115, 2012.

[18] C. Gu and Y. Zhu, "Provable security of ID-based proxy signature schemes," in *Proceedings of the 3rd International Conference on Computer Network and Mobile Computing (ICCNMC '05)*, X. Lu and W. Zhao, Eds., vol. 3619 of *Lecture Notes in Computer Science*, pp. 1277–1286, Springer, Zhangjiajie, China, August 2005.

[19] C. Gu and Y. Zhu, "An efficient ID-based proxy signature scheme from pairings," in *Proceedings of the 3rd SKLOIS Conference on Information Security and Cryptology (INSCRYPT '07)*, vol. 4990 of *Lecture Notes in Computer Science*, pp. 40–50, Springler, Xining, China, September 2007.

[20] D. B. He, J. H. Chen, and J. Hu, "An ID-based proxy signature schemes without bilinear pairings," *Annales des Télécommunications*, vol. 66, no. 11-12, pp. 657–662, 2011.

[21] H. Ji, W. Han, L. Zhao, and Y. Wang, "An identity-based proxy signature from bilinear pairings," in *Proceedings of the WASE International Conference on Information Engineering (ICIE '09)*, pp. 14–17, Shanxi, China, July 2009.

[22] S. Kim, S. Park, and D. Won, "Proxy signatures, revisited," in *Proceedings of the 1st International Conference on Information and Communication Security (ICICS '97)*, Y. Han and S. Quing, Eds., vol. 1334 of *Lecture Notes in Computer Science*, pp. 223–232, Springer, Heidelberg, Germany, 1997.

[23] B. Lee, H. Kim, and K. Kim, "Strong proxy signature and its applications," in *Proceedings of the Symposium on Cryptography and Information Security (SCIS '01)*, pp. 603–608, Oiso, Japan, January 2001.

[24] B. Lee, H. Kim, and K. Kim, "Secure mobile agent using strong non designated proxy signature," in *Proceedings of the 6th Australasian Conference on Information Security and Privacy (ACISP '01)*, V. Varadharajan and Y. Mu, Eds., vol. 2119 of *Lecture Notes in Computer Science*, pp. 474–486, Springer, Sydney, Australia, July 2001.

[25] J. Y. Lee, J. H. Cheon, and M. Kim SJoye, "An analysis of proxy signatures: is a secure channel necessary?" in *Proceedings of the RSA Conference on the Cryptographers' Track (CT-RSA '03)*, vol. 2612 of *Lecture Notes in Computer Science*, pp. 68–79, Springer, San Francisco, Calif, USA, April 2003.

[26] T. Malkin, S. Obana, and M. Yung, "The hierarchy of key evolving signatures and a characterization of proxy signatures," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT '04)*, C. Cachin and J. L. Camenisch, Eds., vol. 3027 of *Lecture Notes in Computer Science*, pp. 306–322, Springer, Interlaken, Switzerland, May 2004.

[27] T. Okamoto, A. Inomata, and E. Okamoto, "A proposal of short proxy signature using pairing," in *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC '05)*, pp. 631–635, IEEE Computer Society Press, Los Alamitos, Calif, USA, April 2005.

[28] T. Okamoto, M. Tada, and E. Okamoto, "Extended proxy signatures for smart cards," in *Proceedings of the 2nd International Workshop on Information Security (ISW '99)*, Y. Zheng and M. Mambo, Eds., vol. 1729 of *Lecture Notes in Computer Science*, pp. 247–258, Springer, Kuala Lumpur, Malaysia, November 1999.

[29] G. Wang, F. Bao, J. Zhou, and R. H. Deng, "Security analysis of some proxy signatures," in *Proceedings of the 6th International Conference on Information Security and Cryptology (ICISC '03)*, J.-I. Lim and D.-H. Lee, Eds., vol. 2971 of *Lecture Notes in Computer Science*, pp. 305–319, Springer, Seoul, Republic of Korea, November 2004.

[30] W. Wu, Y. Mu, W. Susilo, J. Seberry, and X. Y. Huang, "Identity-based proxy signature from pairings," in *Proceedings of the 4th International Conference on Autonomic and Trusted Computing (ATC '07)*, vol. 4610 of *Lecture Notes in Computer Science*, pp. 22–31, Springer, Hong Kong, July 2007.

[31] J. Xu, Z. Zhang, and D. Feng, "ID-based proxy signature using bilinear pairings," in *Parallel and Distributed Processing and Applications—ISPA 2005 Workshops*, G. Chen, Y. Pan, M. Guo, and J. Lu, Eds., vol. 3759 of *Lecture Notes in Computer Science*, pp. 359–367, Springer, Heidelberg, Germany, 2005.

[32] K. Choi and D. Lee, "Certificateless proxy signature scheme," in *Proceedings of the 3rd International Conference on Multimedia, Information Technology and Its Applications (MITA '07)*, pp. 437–440, Manila, Philippines, August 2007.

[33] Y. C. Chen, C. L. Liu, G. Horng, and K. C. Chen, "A provably secure certificateless proxy signature scheme," *International*

Journal of Innovative Computing, Information & Control, vol. 7, no. 9, pp. 5557–5569, 2011.

[34] R. Lu, D. He, and C. Wang, "Cryptanalysis and improvement of a certificateless proxy signature scheme from bilinear pairings," in *Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD '07)*, pp. 285–290, IEEE Computer Society, Qingdao, China, August 2007.

[35] X. Li, K. Chen, and L. Sun, "Certificateless signature and proxy signature schemes from bilinear pairings," *Lithuanian Mathematical Journal*, vol. 45, no. 1, pp. 76–83, 2005.

[36] S. H. Seo, K. Y. Choi, J. Y. Hwang, and S. Kim, "Efficient certificateless proxy signature scheme with provable security," *Information Sciences*, vol. 188, pp. 322–337, 2012.

[37] H. Xiong, F. G. Li, and Z. G. Qin, "A provably secure proxy signature scheme in certificateless cryptography," *Informatica*, vol. 21, no. 2, pp. 277–294, 2010.

[38] L. Zhang, F. T. Zhang, and Q. H. Wu, "Delegation of signing rights using certificateless proxy signatures," *Information Sciences*, vol. 184, no. 1, pp. 298–309, 2012.

[39] X. Cao, W. Kou, and X. Du, "A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges," *Information Sciences*, vol. 180, no. 15, pp. 2895–2903, 2010.