Bull. Korean Math. Soc. **53** (2016), No. 4, pp. 1051–1069
http://dx.doi.org/10.4134/BKMS.b150516
pISSN: 1015-8634 / eISSN: 2234-3016

CRYPTANALYSIS AND IMPROVEMENT OF A PROXY SIGNATURE WITH MESSAGE RECOVERY USING SELF-CERTIFIED PUBLIC KEY

Manoj Kumar Chande and Cheng-Chi Lee

Abstract. Combining the concept of self-certified public key and message recovery, Li-Zhang-Zhu (LZZ) gives the proxy signature scheme with message recovery using self-certified public key. The security of the proposed scheme is based on the discrete logarithm problem (DLP) and one-way hash function (OWHF). Their scheme accomplishes the tasks of public key verification, proxy signature verification, and message recovery in a logically single step. In addition, their scheme satisfies all properties of strong proxy signature and does not use secure channel in the communication between the original signer and the proxy signer. In this paper, it is shown that in their signature scheme a malicious signer can cheat the system authority (SA), by obtaining a proxy signature key without the permission of the original signer. At the same time malicious original signer can also cheat the SA, he can also obtain a proxy signature key without the permission of the proxy signer. An improved signature scheme is being proposed, which involves the remedial measures to get rid of security flaws of the LZZ et al.'s. The security and performance analysis shows that the proposed signature scheme is maintaining higher level of security, with little bit of computational complexity.

1. Introduction

The public key cryptosystem (PKC), is synonymously known as asymmetric key cryptosystem. In this a pair of keys are used, one for encryption known as public key, which encrypts data, and the other one is a private key for decryption. The first important contribution in the development of PKC was given by Diffie and Hellman [2]. They defined PKC and it's associated components like, OWHF and trapdoor information. The other popular PKC's are RSA [15], ElGamal [3], and ECC [6, 10]. In the traditional PKC, a certification is required by the certification authority (CA), to bind a user's identity and its public key. In real time scenario, if the number of user increases, then it

Received June 30, 2015; Revised January 15, 2016.
2010 *Mathematics Subject Classification.* 94A60.
Key words and phrases. discrete logarithm, digital signature, proxy signature, message recovery.

is difficult to manage this certification process. To overcome with this situation, Shamir [17], proposed an identity-based (ID-based) PKC. His approach employs the user's identity as his/her public key, therefore the certification of public key is not required, which reduces the amount of storage, communication and computation. This ID-based approach effectively solves the problem of public key verification, but the disadvantage is that CA knows secret keys of all users after registration. Therefore, CA may masquerade as any legitimate user by generating a valid key pair for the user without being detected. This creates a problem for public key verification process.

The problem of public key verification stands until, Girault [4], gives solution in form of self-certified public keys. In this system each user's public key is signed by CA using private key of CA himself. This key system has the following features- First the secret key can be determined by the user himself/ herself or jointly by the user and CA, and does not known to CA. Secondly the user can use his/her own secret key to verify the authenticity of the self-certified public key issued by CA, and thus no other certification is required. Another important feature is the task of public key verification can be further accomplished with subsequent cryptographic application in a logically single step. This is the reason for the self-certified approach to be more cost efficient as compared to the certificate-based and the ID-based approaches. This approach also helps to resists the active attacks on public keys, in which an adversary (Adv) looking to replace or modify an original public key by a fake public key of his choice.

The security objectives of PKC like, confidentiality, integrity, message authentication and non-repudiation, can be achieve through one of the most important cryptographic tool known as digital signature. Applications for digital signatures range from secure electronic communication, legal signing of contracts to licensed software updates. Digital signature provide a method to assure that, a message is in fact originates from the person who claims to have generated the message. The commonly used digital signatures are RSA [15], ElGamal [3], DSA [11], and ECDSA [1]. These signature schemes do not have the message recovery feature. In 1994, Nyberg and Rueppel [12], gave digital signature schemes allowing message recovery. In this kind of schemes the message can be conveyed within the signature and can be recovered at the verifier's site. The message need not be hashed or sent along with the signature which saves storage space and communication bandwidth. The security of their signature scheme is based DLP.

Suppose a top official, of any workplace needs to move out of station. In this circumstances he/she is not able to sign routine official documents. So, he/she delegate his/her authority to some subordinate (known as proxy signer), who perform this task in his/her behalf. To help out in this circumstances, Mambo [9], introduced the concept of proxy signature. This variant of signature scheme allows an original signer to delegate his/her signing power to a different

signer, called proxy signer. The proxy signer can stand proxy for the original signer to generate signatures, referred to as proxy signatures.

There arises a natural question that, is it possible to design a signature scheme with the merits of self-certified public key system, proxy signature and message recovery signature scheme. In the year 2004 Hsu and Wu [5], gives efficient proxy signature scheme using self-certified public keys. Shao [18], shows that the Hsu and Wu [5], scheme is not secure. In their scheme it is possible that a malicious signer or dishonest original signer can cheat the CA. An attacker can do attack with the CA, to obtain a proxy signature key without the permission of the original signer. In the year 2005, LZZ [7], has given a new design of proxy signature scheme with message recovery using self-certified public key. The security of the proposed scheme is based on well-known, DLP and OWHF. Their scheme accomplishes the tasks of public key verification, proxy signature verification, and message recovery in a logically single step. In addition they claim that their scheme satisfies all properties of strong proxy signature and does not use secure channel in the communication between the original signer and the proxy signature signer.

In succession variants of the proxy signature with message recovery feature are proposed by different researcher's. In the same year 2005 Lu and Cao [8], proposed a designated verifier proxy signature scheme and its security is based on ECDLP. Another variant is given in the year 2009, by Wu and Hsu [21], they give the first multi-proxy signature schemes, their schemes are based on DLP and ECDLP respectively. In the year 2012, Xie [22], shows that Wu et al's scheme is not secure against proxy warrant revision attack. In the same year, the first identity-based proxy signature scheme with message recovery using bilinear pairing is proposed by Singh and Verma [19]. Tian, Huang and Yang [20], via two concrete attacks showed that Singh's scheme is not secure. Padhey and Tiwari [13], claim that they proposed the first certificateless proxy signature with message recovery, whose security is based on ECDLP. They also claim that their signature scheme is secure against existential forgery under adaptive chosen message, ID attacks, and furthermore, it is more efficient than Singh and Verma [19], scheme for practical applications.

In this paper, we focus on the signature scheme given by LZZ [7]. It is shown that their signature scheme is not secure. A malicious signer can cheat the system authority (SA), during proxy key extraction. Without the permission of the original signer, the malicious signer is able to extract proxy key. On the other hand, a malicious original signer can cheat the SA, into extracting a proxy signature key without the permission of the proxy signer. To overcome this security flaw, an improved signature scheme is proposed. This paper organized as follows: In Section 2, preliminaries are given. Section 3, gives the brief review of LZZ et al.'s signature scheme. In Section 4, an attack on LZZ scheme is given and an improved scheme is given in Section 5. The detailed analysis of computation and performance of the scheme is being done in Section 6. The

security analysis of the proposed signature scheme is given in Section 7. Last section concludes the work done in this paper.

2. Preliminaries

This section has two subsection, one is about the intractable mathematical problem on which the security of the proposed signature rely and the other is about syntax and security requirements.

2.1. Discrete Logarithm Problem (DLP)

Let g be an element of a finite group G, and order of g is n. The discrete logarithm problem is to find the smallest non-negative integer x such that $g^x = y$. It is easy to compute discrete exponentiation $y = g^x \bmod n$, for given g, x and n, but it is infeasible to determine x, for given y, g and n, when n is large.

2.2. Syntax and security requirements for the proposed signature scheme

(1) Setup
In this phase SA generates the system parameters (p, q, g, β), chooses a one-way hash function $h()$, which are public and keeps his private key as a secret.
(2) User Registration
Each user should get registered who wants to participate in this signature process. For this the user sends some parameters with his identity information to SA, then SA sends him partial secret and a parameter. The user checks the validity of these parameters and finally accepts (x_i, y_i), as his private and public key pair.
(3) Proxy Key Generation
In this phase all the three parties, the original signer U_o, the proxy signer U_{P} and SA exchange a pair of parameter and identity with other two. After this they all compute proxy public key and U_{P} computes his proxy secret key X_{P}.
(4) Proxy Signature Generation
The proxy signer U_{P}, using his secret key generates the proxy signature $(r, s, \Psi, m_w, ID_o, Y_{\mathrm{P}}, ID_{\mathrm{P}})$.
(5) Message Recovery And Signature Verification
The receiver of the signature or signature verifier, using the available public parameters verifies the signature and recovers the signed message in a single step.

The security requirement for proxy signature, that it should satisfy the following properties:

(i) Distinguishability: The proxy signature must be distinguished from the regularly used normal signature.

(ii) Identifiability: One can determine the identity of the original signer U_o and authorized proxy signer U_{P} from a proxy signature.
(iii) Non-repudiation: The signer in signature generation cannot deny after having the signature.
(iv) Prevention of misuse: It can be assured that the only purpose of proxy key pair is to produce proxy signature, which conforms to delegation information. The proxy signer is responsible for any kind of misuse of proxy key pair.
(v) Unforgeability: U_{P}, the proxy signer can only able to produce a valid proxy signature for the original signer. Even the original signer U_o cannot produce it.
(vi) Verifiability: The verifier/receiver of the signature should be able to verify the proxy signature in the same manner as the original signature did verification.

3. Brief review of LZZ signature scheme

In this section the review of LZZ [7], signature scheme is given. Their scheme divided into five steps namely: (1) System Setup (2) User Registration (3) Proxy Key Generation (4) Proxy Signature Generation and (5) Message Recovery And Verification.

(1) System Setup
The SA, randomly selects two large prime numbers p, q such that $q \mid p-1$, a generator g, with order q over $GF(p)$, and a secure OWHF $h()$. SA generates a pair of secret and public key ($\gamma \in Z_q^*$, $\beta = g^{\gamma} \bmod p$). After this, SA publishes, p, q, g, β and $h()$, while keeping γ secret. Let U_o, be the original signer, with an identity ID_o, who wants to delegate his/her signing power to some proxy signer U_{P} with an identity ID_{P}.

(2) User Registration
Suppose that U_o and U_{P} want to register with SA. The procedure for user registration is as follows:

(i) U_o, randomly selects an integer, $t_o \in Z_q^*$ and U_{P} also randomly selects an integer, $t_{\mathrm{P}} \in Z_q^*$, as their master key and computes

$$v_o = g^{h(t_o \| ID_o)} \bmod p,$$
$$v_{\mathrm{P}} = g^{h(t_{\mathrm{P}} \| ID_{\mathrm{P}})} \bmod p,$$

then sends (ID_o, v_o) and $(ID_{\mathrm{P}}, v_{\mathrm{P}})$ to SA. Then SA, saves the identity of all the users in a log file.

(ii) Upon receiving (ID_o, v_o) and $(ID_{\mathrm{P}}, v_{\mathrm{P}})$, SA randomly selects time-variant integers $z_o, z_{\mathrm{P}} \in Z_q^*$ and computes public key's

$$y_o = v_o \cdot g^{z_o} - h(ID_o) \bmod p, \tag{3.1}$$

$$y_{\mathrm{P}} = v_{\mathrm{P}} \cdot g^{z_{\mathrm{P}}} - h(ID_{\mathrm{P}}) \bmod p, \tag{3.2}$$

and their witness's

$$w_o = z_o + \gamma \cdot (y_o + h(ID_o)) \bmod q,$$
$$w_{\mathrm{P}} = z_{\mathrm{P}} + \gamma \cdot (y_{\mathrm{P}} + h(ID_{\mathrm{P}})) \bmod q,$$

then sends (y_o, w_o) and $(y_{\mathrm{P}}, w_{\mathrm{P}})$, to U_o and U_{P} respectively.

(iii) Upon receiving (y_o, w_o) and $(y_{\mathrm{P}}, w_{\mathrm{P}})$, U_o and U_{P} respectively computes their secret key's

$$x_o = w_o + h(t_o \| ID_o) \bmod q,$$
$$x_{\mathrm{P}} = w_{\mathrm{P}} + h(t_{\mathrm{P}} \| ID_{\mathrm{P}}) \bmod q,$$

and verifies the authenticity of the public key's y_o and y_{P}, by checking

$$g^{x_o} = (y_o + h(ID_o)) \cdot \beta^{y_o + h(ID_o)} \bmod p, \tag{3.3}$$

$$g^{x_P} = (y_{\mathrm{p}} + h(ID_{\mathrm{P}})) \cdot \beta^{y_{\mathrm{P}} + h(ID_{\mathrm{P}})} \bmod p. \tag{3.4}$$

Theorem 3.1. *The secret key's x_o, x_P and the corresponding public key's y_o, y_P satisfies equations* (3.3)*, and* (3.4) *respectively.*

Proof. Substituting value of w_i, into x_i, where $i = o, \mathrm{P}$. We have

$$x_i = z_i + \gamma \cdot (y_i + h(ID_i)) + h(t_i \| ID_i) \bmod q \tag{3.5}$$

raising both sides of the equation (3.5), to exponent to base g, using equation (3.1) and (3.2), for $i = o, \mathrm{P}$, it gives

$$\begin{aligned} g^{x_i} &= g^{z_i + \gamma \cdot (y_i + h(ID_i)) + h(t_i \| ID_i)} \bmod p \\ &= v_i \cdot g^{z_i} \cdot \beta^{y_i + h(ID_i)} \bmod p \\ &= (y_i + h(ID_i)) \cdot \beta^{y_i + h(ID_i)} \bmod p \end{aligned}$$

which implies theorem holds. □

(3) Proxy Key Generation
U_o randomly selects $k \in Z_q^*$ and computes

$$K = g^k \bmod p,$$
$$\sigma = x_o \cdot h(K) + k \bmod q,$$

and sends (ID_o, K, σ) to U_{P}. Then U_{P} accepts (ID_o, K, σ), if the equation

$$g^{\sigma} = [(y_o + h(ID_o)) \cdot \beta^{y_o + h(ID_o)}]^{h(K)} \cdot K \bmod p$$

holds. Then U_{P} computes the proxy signature key

$$\sigma' = \sigma + x_{\mathrm{P}} \bmod q.$$

(4) Proxy Signature Generation
Suppose U_{P} wants to sign a message m, where m contains redundancy for later verification, when it is recovered. U_{P} randomly selects $w \in Z_q^*$ and computes

$$r = m \cdot g^{-w} \bmod p, \tag{3.6}$$

$$s = w - \sigma' \cdot h(r) \bmod q, \tag{3.7}$$

then send proxy signature $(r, s, K, ID_o, ID_{\mathrm{P}})$ to a verifier V.

(5) Message Recovery And Signature Verification
Upon receiving proxy signature $(r, s, K, ID_o, ID_{\mathrm{P}})$, the verifier V can recover message m by the equation

$$m = r g^s [\{(y_o + h(ID_o))\beta^{y_o + h(ID_o)}\}^{h(K)} \{(y_{\mathrm{P}} + h(ID_{\mathrm{P}}))\beta^{y_{\mathrm{P}} + h(ID_{\mathrm{P}})} K\}]^{h(r)} \bmod p.$$

The verifier V, verifies the validity of the recovered message by the embedded redundancy information. The correctness of the scheme is shown below.

Theorem 3.2. *The message m can be recovered correctly from the proxy signature (r, s, K, ID_o, ID_P), at the same time, the public key y_o and y_P are also verified indirectly.*

Proof. From the equation (3.6) and (3.7), we have

$$\begin{aligned} g^w &= g^s \cdot g^{\sigma' \cdot h(r)} \bmod p \\ &= g^s \cdot g^{(\sigma + x_{\mathrm{P}}) \cdot h(r)} \bmod p \\ &= g^s \cdot g^{(x_o \cdot h(K) + k + x_{\mathrm{P}}) \cdot h(r)} \bmod p \\ &= g^s \cdot [\{(y_o + h(ID_o))\beta^{(y_o + h(ID_o)}\}^{h(K)} \\ &\quad \cdot \{y_{\mathrm{P}} + h(ID_{\mathrm{P}})\beta^{y_{\mathrm{P}} + h(ID_{\mathrm{P}})} K\}]^{h(r)} \bmod p. \end{aligned}$$

Thus message m can be recovered from $(r, s, K, ID_o, ID_{\mathrm{P}})$ and verified by checking redundancy information, which implies for $i = o, \mathrm{P}$

$$g^{x_i} = (y_i + h(ID_i)) \cdot \beta^{y_i + h(ID_i)} \bmod p$$

this equation holds and the public keys y_o, y_{P}, are also verified simultaneously. □

4. Cryptanalysis of LZZ signature scheme

Let a malicious user U_{P}, with identity ID_{P}, wants to cheat the SA, into extracting a proxy signature key without the permission of the original signer U_o. According to the registration stage U_o has the private and public keys (x_o, y_o), such that U_o should be responsible for the self-certified public key y_o, as per equation (3.3).

The malicious user U_{P} chooses an integer $t_{\mathrm{P}} \in Z_q^*$ randomly and computes

$$v_{\mathrm{P}} = g^{h(t_{\mathrm{P}} \| ID_{\mathrm{P}})} \cdot \{(y_o + h(ID_o)) \cdot \beta^{y_o + h(ID_o)}\}^{-1} \bmod p \tag{4.1}$$

User Registration		
U_i		SA
$t_i \in Z_q^*$ $v_i = g^{h(t_i \| ID_i)} \bmod p$ $x_i = w_i + h(t_i \| ID_i) \bmod p$ Check $g^{x_i} = \{y_i + h(ID_i)\} \cdot \beta^{\{y_i + h(ID_i)\}} \bmod p$	$\xrightarrow{(v_i, ID_i)}$ $\xleftarrow{(y_i, w_i)}$	$z_i \in Z_q^*$ $y_i = v_i \cdot g^{z_i} - h(ID_i) \bmod p$ $w_i = z_i + \gamma \cdot \{y_i + h(ID_i)\} \bmod q$
U_o		SA
Proxy Key Generation		
$K = g^k \bmod p$ $\sigma = x_o \cdot h(K) + k \bmod q$	$\xrightarrow{(ID_o, K, \sigma)}$	Check $g^{\sigma} = \left[\{y_o + h(ID_o)\} \cdot \beta^{\{y_o + h(ID_o)\}}\right]^{h(K)} \cdot K \bmod q$ $\sigma' = \sigma + x_{\mathrm{P}} \bmod q$
Proxy Signature Generation By U_{P}		
$\alpha \in Z_q^*$ Compute $r = m \cdot g^{-w} \bmod p$ $s = w - \sigma' \cdot h(r) \bmod q$ Send Proxy Signature $(r, s, K, ID_o, ID_{\mathrm{P}})$ To Verifier V		
Message Recovery And Signature Verification By Verifier / Receiver		
$m = r \cdot g^s \left[\left\{ (y_o + h(ID_o)) \cdot \beta^{\{(y_o + h(ID_o\}} \right\}^{h(K)} \cdot \left\{ (y_{\mathrm{P}} + h(ID_{\mathrm{P}})) \cdot \beta^{\{(y_{\mathrm{P}} + h(ID_{\mathrm{P}}\}} \cdot K \right\} \right]^{h(r)} \bmod p$		

FIGURE 1. Process flow diagram for LZZ signature scheme

and sends $(v_{\mathrm{P}}, ID_{\mathrm{P}})$ to the SA. Then SA chooses $z_{\mathrm{P}} \in Z_q^*$ and computes

$$y_{\mathrm{P}} = v_{\mathrm{P}} \cdot g^{z_{\mathrm{P}}} - h(ID_{\mathrm{P}}) \bmod p, \tag{4.2}$$

$$w_{\mathrm{P}} = z_{\mathrm{P}} + (y_{\mathrm{P}} + h(ID_{\mathrm{P}})) \cdot \gamma \bmod q, \tag{4.3}$$

then sends $(y_{\mathrm{P}}, w_{\mathrm{P}})$ to U_{P}, then U_{P} computes

$$x_{\mathrm{P}} = w_{\mathrm{P}} + h(t_{\mathrm{P}} \| ID_{\mathrm{P}}) \bmod q \tag{4.4}$$

and verifies its authenticity by the equation

$$\left[\left\{\beta^{y_o + h(ID_o)} \cdot \left(y_o + h(ID_o)\right)\right\} \cdot \left\{\beta^{y_{\mathrm{P}} + h(ID_{\mathrm{P}})} \cdot \left(y_{\mathrm{P}} + h(ID_{\mathrm{P}})\right)\right\}\right] = g^{x_{\mathrm{P}}} \bmod p. \tag{4.5}$$

If this equation holds, the U_{P} accepts $(x_{\mathrm{P}}, y_{\mathrm{P}})$ as his private and public key. After the completion of registration phase, SA publishes y_{P} the public key of U_{P}.

Now U_{P} randomly chooses an integer $k \in Z_q^*$ and computes

$$K = g^k \bmod p, \tag{4.6}$$

$$\sigma' = x_{\mathrm{P}} \cdot h(K) + k \bmod p. \tag{4.7}$$

U_{P} checks its validity as

$$g^{\sigma'} = [\beta^{(y_o + h(ID_o)} (y_o + h(ID_o) \beta^{(y_{\mathrm{P}} + h(ID_{\mathrm{P}})} (y_{\mathrm{P}} + h(ID_{\mathrm{P}}))]^{h(K)} K \bmod p \tag{4.8}$$

thereafter U_{P} can use σ' to sign message on behalf of U_o.

For signing message m on behalf of the original signer U_o, U_{P} randomly chooses an integer $w \in Z_q^*$ and computes (r, s) as follows

$$r = m \cdot g^{-w} \bmod p, \tag{4.9}$$

$$s = w - \sigma' \cdot h(r) \bmod q, \tag{4.10}$$

the proxy signature of m is (r, s) and on receiving this signature any verifier V can recover message m by the equation

$$m = r g^s [\{(y_o + h(ID_o)) \beta^{y_o + h(ID_o)}\}^{h(K)} \{(y_{\mathrm{P}} + h(ID_{\mathrm{P}})) \beta^{y_{\mathrm{P}} + h(ID_{\mathrm{P}})}\} K]^{h(r)} \bmod p. \tag{4.11}$$

Theorem 4.1. *The malicious user U_P, can attack and forge the signature (r, s, K, ID_o, ID_P), by following the procedure mentioned above, which always verifies the equation* (4.11).

Proof. It is sufficient to show that the proxy signature key satisfies the equation (4.8).

$$\begin{aligned}
g^{\sigma'} &= g^{x_{\mathrm{P}} \cdot h(K) + k} \bmod p \\
&= g^{\{w_{\mathrm{P}} + h(t_{\mathrm{P}} \| ID_{\mathrm{P}})\} \cdot h(K) + k} \bmod p \\
&= g^{\{z_{\mathrm{P}} + (y_{\mathrm{P}} + h(ID_{\mathrm{P}})) \cdot \gamma + h(t_{\mathrm{P}} \| ID_{\mathrm{P}})\} \cdot h(K) + k} \bmod p \\
&= \{g^{z_{\mathrm{P}}} \cdot g^{(y_{\mathrm{P}} + h(ID_{\mathrm{P}})) \cdot \gamma} \cdot g^{h(t_{\mathrm{P}} \| ID_{\mathrm{P}})}\}^{h(K)} K \bmod p \\
&= [\{((y_{\mathrm{P}} + h(ID_{\mathrm{P}})) v_{\mathrm{P}}^{-1} \beta^{y_{\mathrm{P}} + h(ID_{\mathrm{P}})} v_{\mathrm{P}} ((y_o + h(ID_o)) \beta^{y_o + h(ID_o)})\}]^{h(K)} K \bmod p \\
&= [\{(y_{\mathrm{P}} + h(ID_{\mathrm{P}}))\} \beta^{y_{\mathrm{P}} + h(ID_{\mathrm{P}})} \{(y_o + h(ID_o))\} \beta^{y_o + h(ID_o)}]^{h(K)} K \bmod p.
\end{aligned}$$

So in this way the forge signature key has the same property as that of the proxy signature key generated by the cooperation between the original signer and the proxy signer. □

5. Improvement of LZZ signature scheme

The following notations are used to demonstrate the proposed signature scheme.

Notation	Description
p, q	Large prime numbers, such that $q \mid p-1$.
m	Message to be signed.
m_{w}	Message warrant.
g	Generator of order q, over a finite field.
$h()$	One way hash function.
γ	Private key of SA.
β	Public key of SA, where $\beta = g^{\gamma} \bmod p$.
U_i	Original and proxy signer, for $i = o, \mathrm{P}$.
ID_i	Identity of original signer and proxy signer, for $i = o, \mathrm{P}$.
x_i	Private key of original signer and proxy signer, for $i = o, \mathrm{P}$.
y_i	Public key of original signer and proxy signer, for $i = o, \mathrm{P}$.
V	The signature verifier.

This improved signature scheme also has the same stages as LZZ scheme.

(1) System Setup
The SA, randomly selects two large prime numbers p, q such that $q \mid p-1$, a generator g with order q, over a finite field and a secure OWHF $h() : \{0,1\}^* \to Z_q$. SA generates a pair of secret and public key ($\gamma \in Z_q^*, \beta = g^{\gamma} \bmod p$). After this, SA publishes p, q, g, β and $h()$, while keeping γ secret. Let U_o, be the original signer, with an identity ID_o, who wants to delegate his/her signing power to some proxy signer U_{P} with an identity ID_{P}.

(2) User Registration
Let the user U_o, randomly chooses a master key $t_o \in Z_q^*$ and computes

$$\mathrm{v_o} = g^{h(t_o \| ID_o)} \bmod p \tag{5.1}$$

then sends $(\mathrm{v_o}, ID_o)$ to SA. After receiving $(\mathrm{v_o}, ID_o)$, SA chooses $z_o \in Z_q^*$ and computes

$$y_o = \mathrm{v_o} \cdot g^{z_o} - h(ID_o) \bmod p, \tag{5.2}$$

$$w_o = z_o + \{(y_o + h(ID_o))\gamma\} \bmod q, \tag{5.3}$$

then sends (y_o, w_o) to U_o. After this, U_o computes

$$x_o = w_o + h(t_o \parallel ID_o) \bmod q \tag{5.4}$$

and checks its validity as

$$\begin{aligned} g^{x_o} &= g^{w_o + h(t_o \| ID_o)} \bmod p \\ &= g^{w_o} \cdot v_o \bmod p \\ &= g^{z_o + \{(y_o + h(ID_o)) \cdot \gamma\}} \cdot \{(y_o + h(ID_o)\} \cdot g^{-z_o} \bmod p \\ &= \{(y_o + h(ID_o)\} \cdot g^{(y_o + h(ID_o)) \cdot \gamma} \bmod p \\ &= (y_o + h(ID_o)) \cdot \beta^{y_o + h(ID_o)} \bmod p \end{aligned}$$

if it holds, then U_o accepts (x_o, y_o) as his private and public key. After the registration process is over, SA publishes y_o, as the public key of U_o. In the similar way the proxy signer is also registered and his key pair is $(x_{\mathrm{P}}, y_{\mathrm{P}},)$. For the future use, SA saves triplet $(\mathrm{v_i}, ID_i, z_i)$, where $i = o, \mathrm{P}$, so that he can check the authenticity of the registered user.

(3) Proxy Key Generation

Let U_o, wants to delegate his signing authority to the designated U_{P}. U_o randomly selects an integer s_{P} and computes

$$u_{\mathrm{P}} = g^{h(s_{\mathrm{P}} \| ID_o)} \bmod p \tag{5.5}$$

after this sending $(u_{\mathrm{P}}, m_{\mathrm{w}})$ to SA and U_{P}. The message warrant m_{w}, contains the original signer's identity, the proxy signer's identity and the delegation period, etc., which also authenticates the designated proxy signer. Then U_{P} randomly chooses $k_{\mathrm{P}} \in Z_q^*$, and computes

$$v_{\mathrm{P}} = g^{h(k_{\mathrm{P}} \| ID_{\mathrm{P}})} \bmod p \tag{5.6}$$

and sends $(v_{\mathrm{P}}, ID_{\mathrm{P}})$ to SA and U_o. Then SA randomly selects an integer $r_{\mathrm{P}} \in Z_q^*$ and computes

$$w_{\mathrm{P}} = g^{h(r_{\mathrm{P}} \| m_{\mathrm{w}})} \bmod p \tag{5.7}$$

after this sending w_{P} to U_{P} and U_o.

As U_o, U_{P} and SA respectively receives $(u_{\mathrm{P}}, v_{\mathrm{P}}, w_{\mathrm{P}}, m_{\mathrm{w}})$, they all compute the proxy public key of U_{P} as

$$Y_{\mathrm{P}} = v_{\mathrm{P}} \cdot u_{\mathrm{P}} \cdot w_{\mathrm{P}} \bmod p. \tag{5.8}$$

After this U_o computes

$$\phi = h(y_o \parallel Y_{\mathrm{P}} \parallel m_{\mathrm{w}}) \cdot x_o + h(y_o \parallel Y_{\mathrm{P}} \parallel ID_{\mathrm{P}}) \cdot h(s_{\mathrm{P}} \parallel ID_o) \bmod q \tag{5.9}$$

and then sends it to U_{P} via a secure channel. As U_{P} receives ϕ, he checks its authenticity as

$$g^{\phi} = \beta^{(y_o + h(ID_o))} \cdot (y_o + h(ID_o))\}^{h(y_o \| Y_{\mathrm{P}} \| m_{\mathrm{w}})} \cdot u_P^{h(y_o \| Y_{\mathrm{P}} \| ID_{\mathrm{P}})} \bmod q \tag{5.10}$$

simultaneously, SA computes

$$\Phi = \{h(y_o \parallel ID_o \parallel Y_{\mathrm{P}} \parallel ID_{\mathrm{P}}) - (y_o + h(ID_o)) \cdot h(y_o \parallel Y_{\mathrm{P}} \parallel m_{\mathrm{w}})\} \cdot \gamma + h(y_o \parallel Y_{\mathrm{P}} \parallel ID_{\mathrm{P}}) \cdot h(r_{\mathrm{P}} \parallel m_{\mathrm{w}}) \bmod q \tag{5.11}$$

and sends it to the proxy signer U_{P}. After receiving Φ, U_{P} checks its authenticity as

$$g^{\Phi} = \beta^{h(y_o \| ID_o \| Y_{\mathrm{P}} \| ID_{\mathrm{P}}) - (y_o + h(ID_o)) \cdot h(y_o \| Y_{\mathrm{P}} \| m_{\mathrm{w}})} \cdot w^{h(y_o \| Y_{\mathrm{P}} \| ID_o)}. \tag{5.12}$$

At last U_{P} computes his proxy private key as

$$X_{\mathrm{P}} = \phi + \Phi + h(y_o \parallel Y_{\mathrm{P}} \parallel ID_{\mathrm{P}}) \cdot h(k_{\mathrm{P}} \parallel ID_{\mathrm{P}}) \bmod q. \tag{5.13}$$

U_{P} verifies his private and public key as

$$g^{X_{\mathrm{P}}} = \beta^{h(y_o \| ID_o \| Y_{\mathrm{P}} \| ID_{\mathrm{P}})} (y_o + h(ID_o))^{h(y_o \| y_{\mathrm{P}} \| m_{\mathrm{w}})} Y_{\mathrm{P}}{}^{h(y_o \| Y_{\mathrm{P}} \| ID_{\mathrm{P}})} \bmod p. \tag{5.14}$$

If this verification holds, then U_{P} can use X_{P}, for signing on behalf of U_o.

(4) Proxy Signature Generation
To sign the message m, U_{P} selects randomly $\alpha \in Z_q^*$ and does the following computation

$$r = \{m \| h(m)\} \cdot g^{-\alpha} \bmod p, \tag{5.15}$$

$$\Psi = h(m \parallel r), \tag{5.16}$$

$$s = \alpha - \Psi \cdot X_{\mathrm{P}} \bmod q. \tag{5.17}$$

So ultimately the proxy signature for the message m, is $(r, s, \Psi, m_{\mathrm{w}}, y_o, ID_o, Y_{\mathrm{P}}, ID_{\mathrm{P}})$.

(5) Message Recovery And Signature Verification
To verify the proxy signature , the verifier V does the following computation

$$m \| h(m) = r \cdot g^s \cdot [\beta^{h(y_o \| ID_o \| Y_{\mathrm{P}} \| ID_{\mathrm{P}})} \cdot (y_o + h(ID_o))^{h(y_o \| Y_{\mathrm{P}} \| m_{\mathrm{w}})} \cdot Y_{\mathrm{P}}{}^{h(y_o \| Y_P \| ID_{\mathrm{P}})}]^{\Psi} \bmod p \tag{5.18}$$

and verifies whether $\Psi = h(m \| r)$, holds or not.

Theorem 5.1. *The message m, can be recovered correctly from the proxy signature $(r, s, \Psi, m_{\mathrm{w}}, y_o, ID_o, Y_P, ID_P)$, at the same time, the public key's and identities are also verified indirectly.*

Proof. The steps for signature verification are as follows:

$$\begin{aligned} m \| h(m) &= r \cdot g^s \cdot [\beta^{h(y_o \| ID_o \| Y_{\mathrm{P}} \| ID_{\mathrm{P}})} \cdot (y_o + h(ID_o))^{h(y_o \| Y_{\mathrm{P}} \| m_{\mathrm{w}})} \\ &\quad \cdot Y_{\mathrm{P}}{}^{h(y_o \| Y_{\mathrm{P}} \| ID_{\mathrm{P}})}]^{\Psi} \bmod p && \text{by (5.18)} \\ &= r \cdot g^s \cdot \{g^{X_{\mathrm{P}}}\}^{\Psi} \bmod p \\ &= r \cdot g^{s + X_{\mathrm{P}} \cdot \Psi} \bmod p \\ &= r \cdot g^{\alpha} \bmod p && \text{by (5.17)} \end{aligned}$$

U_o	U_{P}	SA
Proxy Key Generation		
$s_{\mathrm{P}} \in Z_q^*$	$\xleftarrow{(v_{\mathrm{P}},\ ID_{\mathrm{P}})}$ $k_{\mathrm{P}} \in Z_q^*$, $v_{\mathrm{P}} = g^{h(k_{\mathrm{P}}\|ID_{\mathrm{P}})}$ $\xrightarrow{(v_{\mathrm{P}},\ ID_{\mathrm{P}})}$	$r_{\mathrm{P}} \in Z_q^*$
$u_{\mathrm{P}} = g^{h(s_{\mathrm{P}}\|ID_o)} \bmod p$	$\xrightarrow[To\ U_{\mathrm{P}}\ \&\ SA]{(u_{\mathrm{P}},\ m_w)}$ $\xleftarrow[To\ U_o\ \&\ U_{\mathrm{P}}]{(w_{\mathrm{P}})}$	$w_{\mathrm{P}} = g^{h(r_{\mathrm{P}}\|m_w)} \bmod p$
U_o, U_{P} And SA Compute $Y_{\mathrm{P}} = v_{\mathrm{P}} \cdot u_{\mathrm{P}} \cdot w_{\mathrm{P}} \bmod p$		
$\phi = h(y_o\|Y_{\mathrm{P}}\|m_w) \cdot x_o$ $+h(y_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}}) \cdot h(s_{\mathrm{P}}\|ID_o) \bmod q$	$\xrightarrow{\phi}$ Check $\xleftarrow{\Phi}$	$\Phi = h(y_o\|ID_o\|y_{\mathrm{P}}\|ID_{\mathrm{P}})$ $-(y_o + h(ID_o)) \cdot h(y_o\|Y_{\mathrm{P}}\|m_w) \cdot \gamma$ $+h(y_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}}) \cdot h(r_{\mathrm{P}}\|m_w) \bmod p$
$g^{\phi} = \{\beta^{(y_o + h(ID_o))} \cdot (y_o + h(ID_o))\}^{h(y_o\|Y_{\mathrm{P}}\|m_w)} \cdot {u_{\mathrm{P}}}^{h(y_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}})}$		
$g^{\Phi} = \beta^{h(y_o\|ID_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}}) - (y_o + (ID_o)) \cdot h(y_o\|Y_{\mathrm{P}}\|m_w)} \cdot w^{h(y_o\|Y_{\mathrm{P}}\|ID_o)}$		
Compute		
$X_{\mathrm{P}} = \phi + \Phi + h(y_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}}) \cdot h(k_{\mathrm{P}}\|ID_{\mathrm{P}}) \bmod q$		
$g^{X_{\mathrm{P}}} = \beta^{h(y_o\|ID_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}})} \cdot (y_o + h(ID_o))^{h(y_o\|Y_{\mathrm{P}}\|m_w)} \cdot Y_{\mathrm{P}}^{h(y_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}})} \bmod p$		
Proxy Signature Generation By Proxy Signer		
$\alpha \in Z_q^*$		
$r = \{m\|h(m)\} \cdot g^{-\alpha}$		
$\Psi = h(m\|r)$		
Proxy Signature Is $(r, s, \Psi, m_w, y_o, ID_o, Y_{\mathrm{P}}, ID_{\mathrm{P}})$		
Verify X_{P}		
Message Recovery & Signature Verification By Signature Receiver / Verifier		
$m\|h(m) = r \cdot g^s \cdot \left[\beta^{h(y_o\|ID_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}})} \cdot (y_o + h(ID_o))^{h(y_o\|Y_{\mathrm{P}}\|m_w)} \cdot Y_{\mathrm{P}}^{h(y_o\|Y_{\mathrm{P}}\|ID_{\mathrm{P}})}\right]^{\Psi} \bmod p$		

FIGURE 2. Our improved scheme

$$= r \cdot m\|h(m) \cdot r^{-1} \bmod p \qquad \text{by (5.15)}$$
$$= m\|h(m).$$

□

6. Performance and computational analysis

The notations used to elaborate computational load and performance of the proposed scheme are as follows:

h – Hashing of some value.
e – Exponentiation operation.
m – Multiplication of two quantity.
i – Inversion under modulo operation.
a – Simple addition or subtraction.

On the basis of time estimates given in [14], we roughly estimate the time taken for different phases of the proposed signature scheme. The computational load for addition/subtraction is not taken into consideration because it takes negligible CPU time.

First we give the computational detail of the LZZ [7], scheme. This scheme is not too much complex as far as mathematical calculations are concern. Table 1 given below shows that the computational load and timings is not that much huge for this scheme.

TABLE 1. Algebraic Operations & Computation Time (in ms) For LZZ [7], Scheme (For 128 bit).

Phase	U_o	U_{P}	SA	Verifier	Totel Operations	Timing
Reg.	2h + 3e + m +2a	2h + 3e + m + 2a	2h + 2e + 4m + 6a		6h + 8e + 6m + 10a	35.368428
Proxy Gen	h + e + m + a	h + 3e + 2m + 2a			3h + 4e + 3m + 3a	17.684214
Proxy Sig		h + e + 2m + i + a			h + e + 2m + i + a	09.330075
Ver.				4h + 5e + 6m + 2a	4h + 5e + 6m + 2a	25.921546
Total	3h + 4e + 2m + 3a	5h + 7e + 5m + i + 5a	2h + 2e + 4m + 6a	4h + 5e + 6m + 2a	14h + 18e + 17m + i + 1 6a	88.304263

In next Table 2, the computational load and roughly estimated timings are given for the proposed scheme. This scheme is little bit complex than LZZ [7], due to improvements regarding security concern. In our scheme the proxy signer U_{P}, is having maximum load of computation.

TABLE 2. Algebraic Operations & Computation Time (in ms) For Proposed Scheme (For 128 bit).

Phase	U_o	U_{P}	SA	Verifier	Total Operations	Timing
Reg.	2h + 3e + m + 2m		h + e + 2m + 3a		3h + 4e + 3m + 5a	17.684214
Proxy Gen	3h + e + 4m + a	7h + 11e + 10m + 3a	5h + e + 5m + i + 2a		15h + 13e + 19m + i + 6a	82.174803
Proxy Sig		2h + e + 2m + i + a			2h + e + 2m + i + a	10.843801
Ver.				6h + 5e + 4m + a	6h + 5e + 4m + a	25.893134
Total	5h + 4e + 5m + 3a	9h + 12e + 12m + i + 4a	6h + 2e + 7m + i + 5a	6h + 5e + 4m + a	26h + 23e + 28m + 2i + 13a	136.595952

From Table 1 and Table 2, it can be observed that the computational load and their timings for user registration, proxy signature generation and verification phases are approximately, 50.00%, 116.00% and almost equal to the counterparts of LZZ [7], respectively. But the calculation of the proxy key generation, shows the complexity of the proposed scheme. In this phase the load is roughly 4.6 times, than the LZZ [7], scheme. This computation contributes the major portion of overall computational timings. Overall the proposed scheme

has an extra computational load due to higher level of security, which exceeds roughly 55.00%, than the previous scheme. The next section is about the security analysis, which shows the advantage of our scheme as far as the cryptographic security is concerned.

7. Security analysis of the proposed signature scheme

This section has two subsection. One shows that how all the fundamental properties are fulfilled by the proposed signature. The other is about the strength of the proposed signature, that how it is secure from different threats and attacks.

7.1. Security properties

(1) Distinguishability: The proxy signature $(r, s, \Psi, m_{\mathrm{w}}, y_o, ID_o, Y_{\mathrm{P}}, ID_{\mathrm{P}})$, consists of message warrant m_w and proxy signer's public key Y_{P}. This proxy public key is generated by contribution of U_o, U_{P} and SA, through the equation (5.8). In this way the signature is distinguished by the normal signature.

(2) Identifiability: The proxy signature contains ID_o and ID_{P}, which are identities of original signer U_o and U_{P}, so anyone can identify them. Proxy signer's public key involves v_{P}, u_{P}, which can be calculated respectively using, ID_o and ID_{P}. So in this way anyone can identify original and proxy signer.

(3) Nonrepudiation: In the verification step s, is used, which involves proxy private key X_{P}. Proxy public key Y_{P} and identity ID_{P}, is also used so the proxy signer cannot deny that he has signed the message. X_{P} involves ϕ and Φ, which are calculated only by U_o and SA respectively, so they are also not in a position of denial.

(4) Prevention of Misuse: The proxy key pair misuse is prevented through message warrant m_{w}, because, it contains identifiers of original and proxy signer, message type to be signed by the proxy signer, delegation period, etc. So proxy key pair cannot be used for any other purpose or to sign some other message.

(5) Unforgeability: Original signer or some attacker pretend to be as proxy signer to sign illegally the message m. For this they need private key X_{P} of U_{P} and to get this X_{P}, is infeasible, because of the unknown parameter $t_{\mathrm{P}} \in Z_q^*$, which is known to U_{P} only. So it difficult to forge the proxy signature.

(6) Verifiability: Verifier or any user can make sure from the proxy signature $(r, s, \Psi, m_{\mathrm{w}}, y_o, ID_o, Y_{\mathrm{P}}, ID_{\mathrm{P}})$, that U_o agrees with the signed message, since the proxy public key Y_{P} has a component u_{P}. This u_{P}, is calculated by original signer using a random number s_{P}, and which is known to him only.

7.2. Security strength

(1) Security Measures To Resist Forgery Attacks on Private Key's.
   (i) Security of Private Key (γ) of SA.
   To obtain private key γ, of SA from his public key β, is difficult due to DLP, since $\beta = g^{\gamma} \bmod p$. γ, is used to produce the public information w_o. This w_o, involves random number $z_o \in Z_q^*$ selected by SA, so again, it is infeasible to get γ, from w_o. In all, it is difficult to obtain γ from all available public parameters.
   (ii) Security of The Master Key (t_o), of The Authorized Original Signer.
   Description of the registration stage, shows that the secret key of the legitimate original signer U_o is computed from the equation (5.4). This secret key is secure through the random value $h(t_o \| ID_o)$, and this random value can be obtained by equation (5.1), if the solution of DLP is possible. Even if the secret key (x_o), is revealed to someone, still the master key (t_o) will remain safe due to irreversible OWHF.
   (iii) Security of Secret Key (x_o), of A Legitimate Original Signer.
   The safety of the secret key x_o, is mainly due to t_o. Safety of t_o, is already discussed perviously.
   (iv) Security of Proxy Secret Key (X_{P}), of a Proxy Signer.
   Suppose an adversary wants to expose proxy private key X_{P}, using an intercepted proxy signature $(r, s, \Psi, m_{\mathrm{w}}, y_o, ID_o, Y_{\mathrm{P}}, ID_{\mathrm{P}})$ generated by proxy signer. The proxy secret key X_{P} is calculated through equation (5.13), and due to unknown values ϕ, Φ and t_{P}, it is not feasible to calculate X_{P} from the information available in public domain. So an adversary don't have enough information regarding secure parameters to calculate proxy secret key X_{P}.

(2) Security Against Some Possible Attacks.
   (i) Active Attack Through Re-registration of An Existing Authorized User.
   Let an adversary tries to re-register the identity information which is already registered with SA for U_o and U_{P}, in such a way that, SA will provide another valid self-certified public key to masqueraded as U_o and U_{P}. If the adversary succeeds in this attack, then he can universally impersonate U_o and U_{P}. The subsequent proxy signature is also valid without being detected the forgery, because the adversary has corresponding secret key associated to the reproduced public key. The best way to prevent from this attack is that save all information regarding identities of the registered users in a log file. SA should be careful and check if the information regarding identity, submitted by the original signer and

the proxy signer already exists in the log file then SA stops the proceedings and avoids this kind of attack.

(ii) Active Attack Through Generating Key Pair For Non-existing Users.
An adversary may attempt to generate a valid pair of secret and public key, by its own, through supplying fake information regarding identity, without taking the assistance of SA. So the pseudo key pair (x_i, y_i) and identity information ID_i, must satisfy the equation

$$g^{x_i} = \{y_i + h(ID_i)\} \cdot \beta^{\{y_i + h(ID_i)\}}$$

otherwise they are of no use to produce proxy signature. To validate the above equation with (x_i, y_i) and ID_i, the adversary has to encounter difficulty of DLP and OWHF.

(iii) Security Against Signature Forgery Attack.
From the different phases of the proposed signature scheme, it is clear that it combines the two well know signature schemes [12] and [16]. To sign some message, without having knowledge of proxy signer's private key, an adversary has to encounter DLP and OWHF. So it not possible to forge proxy signature scheme.

(iv) Security Against Public Key substitution Attack.
A dishonest original signer, attempts a public key substitution attack by modifying his public key using randomly selecting necessary parameters. But it is again difficult for him to find such substitution of his public key, which satisfies equation (5.18), and it is obviously due to DLP.

8. Conclusion

In this paper, it is shown that the LZZ et al.'s signature scheme is vulnerable to the cheat attacks. In addition, an improved scheme is also proposed. The proposed scheme has merit that the original signer and the proxy signer's public key can simultaneously be authenticated in verifying proxy signature process, which make the proposed scheme withstand the cheat attacks. Computational load and complexity of the signature scheme is little bit increased, but ultimately security matters.

References

[1] ANSI X9.62-1999, *The elliptic curve digital signature algorithm* (*ECDSA*), Technical report, American Bankers Association, 1999.

[2] W. Diffie and M. E. Hellman, *New directions in cryptography*, IEEE Trans. Inform. Theory **IT-22** (1976), 644–654.

[3] T. ElGamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Trans. Inform. Theory **IT-31** (1985), 469–472.

[4] M. Girault, *Self-certified public keys*, Advances in Cryptology Eurocrypt' 91, 491–497, Berlin: Springe-Verlag, 1991.

[5] C. L. Hsu and T. S. Wu, *Efficient proxy signature schemes using self-certified public keys*, Appl. Math. Comput. **152** (2004), no. 3, 807–820.
[6] N. Koblitz, *Elliptic curve cryptosystems*, Math. Comp. **48** (1987), no. 177, 203–209.
[7] J. G. Li, Y. C. Zhang, and Y. L. Zhu, *A new proxy signature scheme with message recovery using self-certified public key*, Wuhan Univ. J. Nat. Sci. **10** (2005), no. 1, 219–222.
[8] R. Lu and Z. Cao, *Designated verifier proxy signature scheme with message recovery*, Appl. Math. Comput. **169** (2005), no. 2, 1237–1246.
[9] M. Mambo, K. Usuda, and E. Okamoto, *Proxy signatures*: *delegation of the power to sign messages*, IEICE Trans. Fundam. **E79-A** (1996), no. 9, 1338–1354.
[10] V. S. Miller, *Use of elliptic curves in cryptography*, Advances in cryptologyCRYPTO '85 (Santa Barbara, Calif., 1985), 417–426, Lecture Notes in Comput. Sci., 218, Springer, Berlin, 1986.
[11] National Institute of Standards and Technology, *A proposed federal information processing standard for digital signature standard* (*DSS*), Federal Register **56** (1991), no. 169, 42980–42982.
[12] K. Nyberg and A. R. Rueppel, *Message recovery for signature schemes based on the discrete logarithm problem*, Advances in cryptologyEUROCRYPT '94 (Perugia), 182–193, Lecture Notes in Comput. Sci., 950, Springer, Berlin, 1995.
[13] S. Padhye and N. Tiwari, *ECDLP-based certificateless proxy signature scheme with message recovery*, Trans. Emerging Tel. Tech. **26** (2015), 346–354.
[14] R. Rajaram Ramasamy and M. Amutha Prabakar, *Digital signature scheme with message recovery using knapsack-based ECC*, Int. J. Network Security **12** (2011), no. 1, 7–12.
[15] R. L. Rivest, A. Shamir, and L. M. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, Comm. ACM **21** (1978), no. 2, 120–126.
[16] C. P. Schnorr, *Efficient signature generation by smart cards*, J. Cryptol. **3** (1991), no. 3, 161–174.
[17] A. Shamir, *Identity-based cryptosystems and signature schemes*, Advances in cryptology (Santa Barbara, Calif., 1984), 47–53, Lecture Notes in Comput. Sci., 196, Springer, Berlin, 1985.
[18] Z. Shao, *Improvement of efficient proxy signature schemes using self-certified public keys*, Appl. Math. Comput. **168** (2005), no. 1, 222–234.
[19] H. Singh and G. K. Verma, *ID-based proxy signature scheme with message recovery*, J. Sys. Software **85** (2012), 209–214.
[20] M. Tian, L. Huang, and W. Yang, *Cryptanalysis of an ID-based proxy signature scheme with message recovery*, Appl. Math. Inf. Sci. **6** (2012), no. 3, 419–422.
[21] T. S. Wu, C. L. Hsu, and H. Y. Lin, *Self-certified multi-proxy signature schemes with message recovery*, J. Zhejiang Univ. Sci. A **10** (2009), no. 2, 290–300.
[22] Q. Xie, *Provably secure self-certified multi-proxy signature with message recovery*, J. Networks **7** (2012), no. 10, 1616–1623.

Manoj Kumar Chande
School of Studies in Mathematics
Pt. Ravishankar Shukla University
Raipur, 492010, Chhattisgarh, India
and
Shri Shankaracharya Institute of Professional Management & Technology
Raipur, 492015, Chhattisgarh, India
E-mail address: manojkumarchande@gmail.com

Cheng-Chi Lee
Department of Library and Information Science
Fu Jen Catholic University
510 Jhongjheng Road, Taipei 24205, Taiwan, R.O.C.
and
Department of Photonics and Communication Engineering
Asia University
No. 500, Lioufeng Road, Taichung City 41354, Taiwan, R.O.C.
E-mail address: cclee@mail.fju.edu.tw