A short ID-based proxy signature scheme

Maryam Rjabzadeh Asaar  
*Sharif University of Technology, asaar@ee.sharif.edu*

Mahmoud Salmasizadeh  
*Sharif University of Technology, salmasi@sharif.edu*

Willy Susilo  
*University of Wollongong, wsusilo@uow.edu.au*

Publication Details
A short ID-based proxy signature scheme

Keywords
proxy, short, signature, scheme, id

Disciplines
Engineering | Science and Technology Studies

Publication Details
A Short ID-based Proxy Signature Scheme

Maryam Rajabzadeh Asaar\(^1\), Mahmoud Salmasizadeh\(^2\), and Willy Susilo\(^{**2}\)

\(^1\) Department of Electrical Engineering, 
\(^2\) Electronics Research Institute (Center), Sharif University of Technology, Tehran, Iran. 
Centre for Computer and Information Security Research, University of Wollongong, Australia. asaar@ee.sharif.ir, salmasi@sharif.edu, wsusilo@uow.edu.au

Abstract. The notion of identity-based proxy signature with message recovery feature has been proposed to shorten identity-based proxy signatures and improve their communication overhead since signed messages are not transmitted with these kinds of signatures. There are a few schemes for this notion: Singh and Verma’s scheme and Yoon et al.’s scheme. Unfortunately, Tian et al. by presenting two forgery attacks show that Singh and Verma’s scheme is not secure, and also Yoon et al.’s scheme does not support provable security. The contributions of this paper are twofold. First, we review Yoon et al.’s scheme and discuss why it does not have message recovery property, and consequently it is not short. Second, we propose a short identity-based proxy signature scheme with the help of message recovery property, and show that it is secure under CDH assumption in the random oracle model. Furthermore, our scheme is more efficient than (as efficient as) previous identity-based proxy signatures.

Keywords: identity-based signature, identity-based signature with message recovery, identity-based proxy signature, CDH assumption, random oracle model.

1 Introduction

The possibility to implement digital signatures is one of the most important achievements of modern cryptography. Digital signatures are widely deployed around the world and have the backing of significant international legislation to support their use in electronic business. Businesses need to be flexible about the way that they employ signatures and so there is a need for digital signature algorithms to support typical business practices. One such flexible type of signature is the proxy signature which permits the common business practice of delegating signing authority in a flexible manner. We are interested in exploring proxy signatures in the identity-based setting.

Identity-based cryptography. Public-key cryptography has many different applications, but in its basic format requires extensive public-key infrastructure for practical use [1]. In order to provide more flexible management of public keys the notion of identity-based cryptography was introduced by Shamir [2]. The main feature of identity-based cryptosystems is to remove the requirement of certification of the public keys. The public key of each entity is obtained from its public identity, such as the IP address or email address, which uniquely identifies it. Since Shamir introduced this notion [2], different identity-based signature schemes [3–5] are proposed. Identity-based signatures with message recovery feature are proposed to minimize the size of signed messages and consequently size of these kinds of signatures. Therefore, these primitives are helpful when an organization intends to reduce the required bandwidth in transmission, or when it is necessary that small messages should be signed. In 2005, Zhang et al. presented the first identity-based signature with message recovery feature [6] for shortening signatures' length, subsequently, Tso et al. [7] presented a revision to Zhang et al.’s scheme, and proposed a construction more efficient than Zhang et al.’s scheme in 2007.

Proxy signatures. Proxy signatures for the first time were introduced by Mambo et al. [8] in 1996. In a proxy signature scheme, an original signer, Alice, can delegate her signing right for signing messages to another signer, Bob, called the proxy signer. Since the notion of proxy signatures has been introduced, several variants of proxy signatures have been proposed. These include proxy signature schemes [9–17], identity-based proxy signature schemes based on the bilinear pairings [18–25], designated verifier signature

\(^{**}\) This research was supported in part by the Office of Vice-President for Science and Technology, I.R. Iran.

\(^{**}\) W. Susilo is supported by the Australian Research Council Discovery Project (DPDP130101383).
schemes [26–28], (identity-based) multi-proxy signatures [23, 29–33], identity-based proxy multi-signatures [30, 34–37], (identity-based) multi-proxy multi-signature schemes [30, 38–43], identity-based proxy signatures with message recovery [44, 45], a certificate-less proxy signature with message recovery and its security analysis [46, 47] and analysis of proxy signatures [48–50]. In this study, we focus on identity-based proxy signature schemes with message recovery property to shorten identity-based proxy signatures.

In identity-based proxy signature schemes with message recovery, a signed message is not required to be transmitted with the signature since the signed message has been inserted to the signature and can be retrieved from the signature by anyone. The main feature of these schemes is to shorten identity-based proxy signatures. This primitive is useful where bandwidth is one of the crucial concerns [51, 52]. If original messages are transmitted with these signatures, it defeats the main purpose of an identity-based proxy signature with message recovery to save bandwidth. In order to save the bandwidth and provide more flexible management of public keys, two identity-based proxy signatures with message recovery have been proposed. The first one was proposed by Singh and Verma [44] in 2012. In 2013, Tian et al. [48] show that Singh and Verma’s scheme is insecure by presenting two forgery attacks. Next, Yoon et al. proposed a secure identity-based proxy signature scheme with message recovery [45] to solve the problem of Singh and Verma’s scheme. Unfortunately, Yoon et al.’s scheme [45] does not support provable security, and also we show that it is not actually a scheme with message recovery property. Then, we present a short identity-based proxy signature scheme with employing message recovery property, which is based on the identity-based signature scheme [4] and the technique employed in [7] to achieve message recovery property, and show that it is secure under CDH assumption in the random oracle model.

1.1 Organization of the paper

The rest of this paper is organized as follows. Section 2 presents bilinear pairings and the CDH complexity assumption employed as the signature foundation, the outline of identity-based proxy signature schemes with message recovery and its security security model. Review and security drawback of Yoon et al.’s identity-based proxy signature with message recovery [45] are given in Section 3. Our proposed scheme and its formal security proof are presented in Section 4. Section 5 and 6 present the comparison and conclusion, respectively.

2 Background

In this section, we review several fundamental backgrounds employed in this research, including bilinear pairings and Computational Diffie-Hellman complexity assumption.

2.1 Bilinear pairings

Let $(G, +)$ and $(G_T, .)$ be two cyclic groups of the same prime order $q$; furthermore, let $P$ be a generator of $G$. The map $e : G \times G \rightarrow G_T$ is said to be an admissible bilinear pairing if the following conditions hold true.

1. $e$ is bilinear which means that $e(cP, dP) = e(P, P)^{cd}$ for all $c$ and $d \in \mathbb{Z}_q^*$
2. $e$ is non-degenerate which means that $e(P, P) \neq 1_{G_T}$
3. $e$ is efficiently computable.

2.2 Complexity assumptions

Definition 1. Computational Diffie-Hellman (CDH) problem. Given $(P, cP, dP \in G)$ for some unknown $c$ and $d \in \mathbb{Z}_q^*$, output $cdP \in G$.

Definition 2. The advantage of $B$, which is a polynomial bounded algorithm with a security parameter $l$, in solving the CDH problem in group $G$ is

$$\text{Adv}_B^{CDH}(l) = \Pr \left[ cdP \leftarrow B(\cdot, cP, dP) \mid c, d \in \mathbb{Z}_q^* \right].$$

The probability is taken over the choice of $c$, $d$ and $B$’s coin tosses. An algorithm $B$ $(t, \epsilon)$-breaks CDH problem on $G$ if it runs in time at most $t$, and $\text{Adv}_B^{CDH}(l)$ is at least $\epsilon$. The CDH assumption states that the CDH problem on $G$ is $(t, \epsilon)$-hard if there is no algorithm $B$ $(t, \epsilon)$-breaks the CDH problem.
2.3 Notations

Throughout the paper, we will use the following notations.

- $y|x$: a concatenation of two strings $y$ and $x$ such that from $y|x$, $y$ and $x$ are effectively recoverable.
- $\oplus$: XOR operation.
- $[y]_{10}$: the decimal representation of $y \in \{0, 1\}^*$.
- $[y]_2$: the binary representation of $y$.
- $t_l|y$: the first left $t_l$ bits of $y$.
- $y|t_r$: the first right $t_r$ bits of $y$.
- $|y|$: the number of bits of $y$.
- $s \leftarrow S$: the operation of assigning a uniformly random element of $S$ to $s$.

2.4 Outline of identity-based proxy signature schemes with message recovery

An original signer with identity $ID_o$ and a proxy signer with identity $ID_p$ and a verifier are participants of an identity-based proxy signature with message recovery. An identity-based proxy signature scheme with message recovery consists of Setup, KeyExtract, DelGen, DelVer, PSign and PVer/MR algorithms as follows [44].

- Setup: Given the system security parameter $l$, it outputs system’s parameters $Para$ and the system’s master key $(msk, mpk)$, i.e. $(Para, (msk, mpk)) \leftarrow Setup(l)$.

- KeyExtract: Given the system’s parameter $Para$, master public key $mpk$, master secret key $msk$, and an identity $ID_o$, it outputs the corresponding secret key $x_o$, i.e. $x_o \leftarrow KeyExtract(Para, mpk, msk, ID_o)$.

- DelGen: Given the system’s parameter $Para$, the master public key $mpk$, a warrant $w$, proxy signer’s identity $ID_p$, the original signer’s secret key $x_o$, it outputs the delegation $\sigma_o$ after a number of interactions to delegate original signer’s signing right on the warrant $w$ to the identity $ID_p$, i.e. $\sigma_o \leftarrow DelGen(Para, mpk, w, ID_p, x_o)$.

- DelVer: Given the system’s parameter $Para$, the master public key $mpk$, the original signer’s identity $ID_o$, the warrant $w$ and proxy signer’s identity $ID_p$ and the delegation $\sigma_o$, it outputs 1 if $\sigma_o$ is a valid delegation of the warrant $w$ for identity $ID_p$ under the identity $ID_o$ and outputs 0 otherwise, i.e. $\{0,1\} \leftarrow DelVer(Para, mpk, ID_o, w, ID_p, \sigma_o)$.

- PSign: Given the system’s parameter $Para$, the master public key $mpk$, original signer’s identity $ID_o$, the proxy signer’s identity $ID_p$, the warrant $w$, the delegation $\sigma_o$, secret key $x_p$ of the proxy signer with identity $ID_p$ and the message $m$ to be signed, it outputs the identity-based proxy signature $\theta$, i.e. $\theta \leftarrow PSign(Para, mpk, ID_o, ID_p, w, \sigma_o, x_p, m)$.

- PVer/MR: Given the system’s parameter $Para$, the master public key $mpk$, the original signer’s identity $ID_o$, the proxy signer’s identity $ID_p$, the warrant $w$ and the proxy signature $\theta$, it first recovers the message $m$, and outputs 1 if $\theta$ is a valid identity-based proxy signature of the message $m$ and outputs 0 otherwise, i.e. $(m, \{0,1\}) \leftarrow PVer/MR(Para, mpk, ID_o, ID_p, w, \theta)$.

2.5 Security model of identity-based proxy signatures with message recovery

In a warrant-based identity-based proxy signature scheme with message recovery, the delegation is the original signer’s standard signature on a proxy signer’s identity $ID_p$ and a warrant $w$ which contains information regarding the period of validity and the restriction on the class of messages for which the warrant is valid. Therefore, the properties of strong identifiability, strong undeniability, verifiability, and prevention of misuse are satisfied naturally. Therefore, the signature scheme should be secure against existential forgery under an adaptive-chosen-message, an adaptive-chosen warrant and chosen identity attack. The adversary $A$ can choose
the identities on which it wants to forge a proxy signature and can request the secret keys corresponding to
them (corrupted users) except for the honest signer, and also $A$ can make delegation and proxy signature
queries on arbitrary warrants and messages under arbitrary identities including honest identity. To achieve
existential unforgeability, three types of potential adversaries are considered. Adversaries of type I which only
have identities of an original and a proxy signer, adversaries of type II which have the secret key of the proxy
signer in addition to capabilities of adversaries of type I and adversaries of type III which have the secret key
of the original signer in addition to identities of the original signer and the proxy signer.

Since an identity-based proxy signature scheme secure against type II (or type III) adversaries is also
secure against type I adversaries we will henceforth only consider type II and type III adversaries. To have
a formal definition for strong unforgeability, the adversary $A$ and the challenger $C$ should interact through
the following game[44].

1. Setup: Algorithm $C$ runs the Setup algorithm with a security parameter $l$ to obtain system’s parameter
$para$ and the master key $(mpk,msk)$, then it sends $(mpk,para)$ to $A$.

2. The adversary $A$ issues a polynomially bounded number of questions to the following oracles adaptively.
   - KeyExtract: The adversary $A$ can ask for the secret key corresponding to $ID_u$, then $C$ returns the
     private key $x_u$ with running the KeyExtract algorithm.
   - DelGen: Adversary $A$ can request the delegation algorithm under the designator’s identity $ID_u$ on
     the pair $(w,ID_p)$ of its choice. Then, $C$ returns $σ_u ← DelGen(Para,mpk,w,ID_p,x_u)$ to $A$.
   - PSign: Adversary $A$ can request the proxy signature of $m$ under proxy signer’s identity $ID_p$, where
     $m$ is in the warrant $w$. In addition, $A$ provides a valid delegation $σ_o$ which indicates that an orig-
     inal signer with identity $ID_o$ delegates its signing right to the proxy signer with identity $ID_p$ on
     the warrant $w$. In response, $C$ firstly runs the KeyExtract algorithm to obtain the secret key $x_p$
     corresponding to the identity of the proxy signer with identity $ID_p$. Next, $C$ runs PSign protocol to
generate $θ ← PSign(Para,mpk,ID_o,ID_p,w,σ_o,x_p,m)$. Then, $C$ returns $θ$ to the adversary $A$.

3. Eventually, $A$ returns a valid signature $(w^*,θ^*)$ w.r.t. an original signer’s identity $ID_o^*$ and a proxy signer’s
    identity $ID_p^*$, and wins the game if the following conditions hold.

1. For adversaries of type II, we have
   - $E_0$: Identity of the original signer $ID_o^*$ has not been requested to the KeyExtract oracle.
   - $E_1$: The pair $(w^*,ID_p^*)$ has not been requested to the DelGen oracle under the original signer’s
     identity $ID_o^*$.

2. For adversaries of type III, we have
   - $E_0$: Identity of the proxy signer $ID_p^*$ has not been requested to the KeyExtract oracle.
   - $E_1$: The message $m^*$ has not been requested to the PSign oracle under the proxy signer’s identity
     $ID_p^*$.

The formal definition of existential unforgeability for adversaries of type II (type III) is expressed in
Definition 3.

**Definition 3.** An identity-based proxy signature is $(t,q_H,q_e,q_d,q_s,ϵ)$-existentially unforgeable against adaptive
chosen message (chosen warrant) attack and chosen identity attack if there is no adversary which runs
in time at most $t$, (makes at most $q_H$ queries to hash functions), makes at most $q_e$ KeyExtract queries, $q_d$
DelGen queries and $q_s$ PSign queries, can win the aforementioned game with probability at least $ϵ$.

3 An Identity-based proxy signature scheme with message recovery

In this section, first we review Yoon et al.’s identity-based proxy signature scheme [45] with message recovery,
then, we show that it does not have message recovery property despite authors’ claim.
3.1 Overview of Yoon et al.’s identity-based proxy signature scheme with message recovery

In 2013, Yoon et al. [45] proposed an identity-based proxy signature with message recovery to eliminate security problems [48] of Singh and Verma’s identity-based proxy signature with message recovery [44] presented by Tian et al. in 2013. This scheme includes an original signer with identity \( ID_o \) and a proxy signer with identity \( ID_p \). Their scheme consists of the following algorithms:

1. Setup: The system parameters are as follows. Let \( t_1, t_2 \in \mathbb{N} \), and let \( G \) be an additive cyclic group of order \( q \) and \( G_T \) be a multiplicative cyclic group of the same order \( q \), where \( q \) is a prime and \( P \in G \) be a generator of \( G \). Let \( H_0 : \{0,1\}^* \rightarrow G, H_1 : G_T \rightarrow \mathbb{Z}_q^*, H : \{0,1\}^* \rightarrow G, F_1 : \{0,1\}^{t_2} \rightarrow \{0,1\}^{t_1}, F_2 : \{0,1\}^{t_1} \rightarrow \{0,1\}^{t_2} \) be random oracles, where \( t_1 + t_2 = q \). It is assumed that \( x \) is a master secret key and \( P_{pub} = xP \) is a master public key. Therefore, public parameters are \( Para = \{G, G_T, H, H_0, H_1, F_1, F_2, e, P, P_{pub}, q, t_1, t_2\} \).

2. KeyExtract: On input a master secret key \( msk = x \) and a user’s identity \( ID_u \), the key distribution center computes \( x_u = xH(ID_u) \), and sends the user’s secret key \( x_u \) over a secure and authenticated channel to the user with identity \( ID_u \).

3. DelGen: The original signer with identity \( ID_o \) chooses \( k_o \in \mathbb{Z}_q^* \), computes \( K_o = k_oP, h_w = H_0(ID_o, w, K_o) \) and \( s = k_o h_w + x_u \), and outputs the delegation \((w, K_o, s)\) in order to delegate its signing right on a warrant \( w \) to a proxy signer with the identity \( ID_p \).

4. PKGen: If a delegation \((w, K_o, s)\) is valid, the proxy signing key \( s + x_p \) is generated by the proxy signer with identity \( ID_p \).

5. PSign: The proxy signer with identity \( ID_p \) can sign a message \( m \) in the warrant \( w \) as follows. The proxy signer with identity \( ID_p \) chooses \( k_p \in \mathbb{Z}_q^* \), computes \( K_p = k_pP, \beta = F_1(m)[F_2(F_1(m)) \oplus m], \alpha = [\beta]_{10} \), \( h_m = H(ID_p, \alpha, K_p) \) and \( U = k_p h_m + d_p \), and returns the proxy signature \((w, K_o, K_p, U, \alpha)\).

6. PVer: Given identities \( ID_o \) and \( ID_p \) and a signature \((w, K_o, K_p, U, \alpha)\), a verifier does as explained below:

   - Checks if \( ID_p \) is authorized by the identity \( ID_o \) in the warrant \( w \), otherwise, it stops.
   - Accepts the proxy signature if and only if \( H_1(e(U, P)e(H(ID_o) + H(ID_p), P_{pub})^{-1}) = H_1(e(h_w, K_o)) \) holds, where \( h_w = H_0(ID_o, w, K_o) \) and \( h_m = H_0(ID_p, \alpha, K_p) \), and recovers the message by \( \beta = [\alpha]_{2} \) and \( m = F_2([1]_{\beta}) \oplus [\beta]_{l_2} \).
   - Checks if the message \( m \) conforms to the warrant \( w \), otherwise, it stops.

3.2 Why Yoon et al.’s scheme does not have message recovery property

Yoon et al.’s identity-based proxy signature scheme with message recovery [45] not only does not support provable security but also it does not have message recovery property despite its authors’ claim. Hence, it is not short. The feature of message recovery in signatures is used to minimized the size of the signatures and the message can be recovered from the signature by everyone. The signature in this scheme is \((w, K_o, K_p, U, \alpha)\), where the message \( m \) is recovered as follows: a verifier computes \( \beta = [\alpha]_{2} \), and then \( m = F_2([1]_{\beta}) \oplus [\beta]_{l_2} \).

To show that Yoon et al.’s scheme does not have message recovery, we use the fact that there is a direct relation between \( \alpha \) and \( m \) and everyone can extract the message \( m \) from \( \alpha \), and \( \alpha \) is transmitted with the signature. Consider a scheme as a modified version of Yoon et al.’s scheme in which we transform \( \alpha \) to \( m \) because of the aforementioned fact. In this case, the result signature is \((w, K_o, K_p, U, m)\), and the signature \( U \) can be generated on \( m \) instead of \( \alpha \). As a consequence, we obtain an ordinary identity-based proxy signature. These two schemes are equivalent in their signature size since \( \alpha \) is equivalent to \( m \), and \( \alpha \) or equivalently \( m \) is transmitted with the signature. Therefore, Yoon et al.’s scheme does not support message recovery property to reduce the signature size.
4 Our identity-based proxy signature scheme with message recovery

In this section, we present an identity-based proxy signature scheme with message recovery based on the identity-based signature scheme [4] and the technique employed in [7] to achieve message recovery property. Then, we prove that it is secure under CDH assumption in the random oracle model.

4.1 Details of our identity-based proxy signature scheme with message recovery

In this section, we present the details of our identity-based proxy signature scheme with message recovery. There are three participants in the system, an original signer with identity \( ID_o \), a proxy signer with identity \( ID_p \), and a verifier. Our scheme consists of seven algorithms as follows.

1. Setup: The system parameters are as follows. Let \( l_1 \) and \( l_2 \in \mathbb{N} \), and let \( \mathbb{G} \) be an additive cyclic group of order \( q \) and \( \mathbb{G}_T \) be a multiplicative cyclic group of the same order \( q \), where \( q \) is a prime and \( P \in \mathbb{G} \) be a generator of \( \mathbb{G} \). Let \( H_0 : \{0,1\}^* \times \mathbb{G}_T \rightarrow \mathbb{Z}_q^* \), \( H_1 : \mathbb{G}_T \rightarrow \{0,1\}^{|q|} \), \( H : \{0,1\}^* \rightarrow \mathbb{G} \), \( F_1 : \{0,1\}^{l_2} \rightarrow \{0,1\}^{l_1}, F_2 : \{0,1\}^{l_1} \rightarrow \{0,1\}^{l_2} \) be random oracles, where \( l_1 + l_2 = |q| \). It is assumed that \( x \in \mathbb{R} \mathbb{Z}_q^* \) is a master secret key and \( P_{pub} = xP \) is a master public key. Therefore, public parameters are \( Para = \{\mathbb{G}, \mathbb{G}_T, H, H_0, H_1, F_1, F_2, e, P, P_{pub}, q, l_1, l_2\} \).

2. KeyExtract: Given \( Para \), master secret key \( msk = x \) and the user identity \( ID_u \), the key distribution center computes \( x_u = xH(ID_u) \), and sends the user’s secret key \( x_u \) over a secure and authenticated channel to the user with identity \( ID_u \).

3. DelGen: Let \( w \) be a warrant for that an original signer with identity \( ID_o \) who wants to delegate its signing right a proxy signer with identity \( ID_p \), the delegation \( \sigma_o \) is generated as follows: the original signer with identity \( ID_o \) selects \( k_o \in \mathbb{Z}_q^* \), computes \( K_o = e(H(ID_o), P_{pub})^{k_o} \), \( h_0 = H_0(w||ID_p, K_o) \) and \( U_o = (h_0 + k_o)x_o \), and outputs the delegation \((w, K_o, U_o)\).

4. DelVer: Given original signer’s identity \( ID_o \) and the delegation \((w, K_o, U_o)\), a verifier checks if \( e(U_o, P) = K_o e(H(ID_o), P_{pub})^{h_0} \) holds, where \( h_0 = H_0(w||ID_p, K_o) \).

5. PSign: The proxy signer with identity \( ID_p \) can sign a message \( m \) such that \( |m| = l_2 \) under the warrant \( w \) with having a valid delegation \( \sigma_o = (w, K_o, U_o) \). To do so, the proxy signer chooses \( k_p \in \mathbb{R} \mathbb{Z}_q^* \), computes \( K_p = K_o e(H(ID_p), P_{pub})^{k_p} \), \( h_1 = H_1(K_p), \beta = F_1(|m|) || F_2(F_1(m)) \) \( \oplus m \), \( \alpha = [\beta \oplus h_1]_{10} \) and \( U = U_o + (\alpha + k_p) x_p \). The proxy signature \( \theta \) on the message \( m \) is \((w, U, \alpha, K_o)\).

6. PVer/MR: Given identities \( ID_o \) and \( ID_p \) and a signature \( \theta = (w, U, \alpha, K_o) \), a verifier does as explained below:
   - Checks if \( ID_p \) is authorized by the identity \( ID_o \) in the warrant \( w \), otherwise, it stops.
   - Computes \( h_1 = H_1(e(U, P) e(H(ID_o), P_{pub})^{-h_0} e(H(ID_p), P_{pub})^{-\alpha}) \), \( \beta \leftarrow [\alpha]_{2} \oplus h_1 \), and then recover the message \( m = [\beta]_{2} \oplus F_2(h_1, |\beta|) \), and accepts the signature \( \theta \) on the message \( m \) if and only if \( F_1(|m|) \) holds, where \( h_0 = H_0(w||ID_p, K_o) \), otherwise, it stops.
   - Checks if the message \( m \) conforms to the warrant \( w \), otherwise, it stops.

4.2 Analysis of the proposed scheme

In this subsection, we verify the correctness and prove existential unforgeability of the new identity-based proxy signature scheme with message recovery in the random oracle model (see [53] for the background). In order to prove unforgeability of the proposed scheme, we need to show that it is unforgeable against adversaries of types II and III (as defined in Section 2.5). Since our security proofs are quite similar for two types of adversaries, we have parametrized these proofs to prevent unnecessary repetitions of arguments. Hence, just for notational settings, we refer to the adversary as \( A_{(1-k)II+kIII} \) in which the parameter \( k \in \{0,1\} \) makes the difference between adversaries of types II and III (i.e. notationally we assume that we have an adversary
of type II, $A_{II}$, when $k = 0$ and an adversary of type III, $A_{III}$, when $k = 1$). Note that, the proofs for different values of $k$ are independent since different types of adversaries are independent.

To prove the security of our proposed scheme, and by contradiction, assuming an adversary $A_{(1-k)II+kIII}$, we show that there is a solver (algorithm $B$) that can solve a random instance of the CDH problem with a non-negligible probability. Our main result on the security of the proposed scheme is summarized in Theorem 1, where the parameter $k$ is used to code the result for both adversaries of types II and III. To start let us verify the correctness of the proposed scheme, and we use $e(U,o) = K_o e(H(ID_o),P_{pub})^{h_o}$ in what follows.

\[
e(U,P)e(H(ID_o),P_{pub})^{-h_o}e(H(ID_p),P_{pub})^{-\alpha} = e(U,o + (\alpha + k_p)x_p,P)e(H(ID_o),P_{pub})^{-h_o}e(H(ID_p),P_{pub})^{-\alpha} = e(U,o,P)e(\alpha x_p,P)e(k_px_p,P)e(H(ID_o),P_{pub})^{-h_o}e(H(ID_p),P_{pub})^{-\alpha} = K_o e(H(ID_o),P_{pub})^{-h_o}e(H(ID_p),P_{pub})^{-\alpha} = K_o e(H(ID_p),P_{pub})^{k_p}e(H(ID_o),P_{pub})^{-h_o} = K_p.
\]

If $\theta$ is a valid signature, $H_1(e(U,P)e(H(ID_o),P_{pub})^{-h_o}e(H(ID_p),P_{pub})^{-\alpha} = K_p) = h_1$, and we have $F_1(m)||F_2(F_1(m)) + m = [a]_2 + h_1$. Hence, the message is recovered as $m = [\beta]|_2 \oplus F_2(l_1|\beta)$ and integrity of the message is checked by $F_1(m) = l_1|m|$.

Also, in what follows we will be needing the following SPLITTING lemma.

**Lemma 1.** [54] Let $S \subset X \times Y$ such that $Pr[(x,y) \in S] \geq \delta$. For any $\gamma < \delta$, define $\Omega = \{(x,y) \in X \times Y | Pr_{y' \in Y}[(x,y') \in S] \geq \delta - \gamma \}$ and $\bar{\Omega} = (X \times Y) \setminus \Omega$, then the following statements hold:

- $Pr[\Omega] \geq \gamma$
- $\forall (x,y) \in \bar{\Omega}, Pr_{y' \in Y}[(x,y') \in S] \geq \delta - \gamma$
- $Pr[\Omega|S] \geq \frac{\gamma}{2}$.

**Theorem 1.** If the CDH problem is $(t',e')$-hard, then the proposed scheme is $(t,q_H,q_{H_0},q_{H_1},q_e,q_d,q_s,\epsilon)$-secure against the adversary $A_{(1-k)II+kIII}$ for a constant $k \in \{0,1\}$ such that

\[
t' \geq \frac{(e_1-2^{-|q|+1})^2(1-2^{-|q|})}{4(q_H + (1-k)q_e + q_d)},
\]

\[
t'' \geq \frac{2t + q_e(t_m + t_e + q_d(t_m + t_e + p + t_e) + q_s(1t_p + 2t_e + 2t_{m_T})),
\]

where $e_1 \geq \frac{q_H}{q} - ((1-k)q_d(2q_2 + q_{H_0})2^{-|G_T|} - k_q)(2q_2 + q_{H_1})2^{-|G_T|}$, and $t_e, t_m, t_{m_T}$ and $t_p$ are the time of an exponentiation in $G_T$, a scalar multiplication in $G$ and $G_T$ and the time of a pairing computation, respectively. In addition, $q_H, q_{H_0}, q_{H_1}, q_e, q_d$ and $q_s$ are the number of queries to oracles $H, H_0, H_1, KeyExtract, DelGen and PSign$, respectively.

**Proof.** It is supposed that there is an adversary $A_{(1-k)II+kIII}$ against unforgeability of the scheme with success probability $\epsilon$. We construct another algorithm $B$ to solve CDH problem with success probability $\epsilon'$. Given a random instance of the CDH problem $(G,q,P,aP,bP)$ output by the challenger $C$, $B$ plans to find $abP$.

The algorithm $B$ runs Setup on a security parameter $l$, and gets $(G,q,P,aP,bP)$ to generate the public parameter $Para = \{G,G_T,H,H_0,H_1,F_1,F_2,e,P,P_{pub} = aP,q,l_1,l_2\}$ and sets $i \leftarrow 1$, and invokes the adversary $A_{(1-k)II+kIII}$ on it. The adversary $A_{(1-k)II+kIII}$ runs in time at most $t$, makes $q_H$ queries to the random oracle $H$, $q_{H_0}$ queries to the random oracle $H_0$, $q_{H_1}$ queries to the random oracle $H_1$, $q_e$ queries to the KeyExtract, $q_d$ queries to the DelGen and $q_s$ queries to the PSign algorithm, and can win the unforgeability game with probability at least $\epsilon$. Algorithm $B$ answers $A_{(1-k)II+kIII}$‘s oracle queries as described below.

- $H_0(\ldots)$ queries: if $T_0[\ldots]$ is defined for query $(w||ID_p, K_o)$, then, $B$ returns its value, otherwise $B$ chooses $T_0[w||ID_p, K_o] \xleftarrow{} Z_q$, and returns $T_0[w||ID_p, K_o]$ to $A_{(1-k)II+kIII}$.

- $H_1(\ldots)$ queries: if $T_1[\ldots]$ is defined for query $K_p$, then, $B$ returns its value, otherwise $B$ chooses $T_1[K_p] \xleftarrow{} \{0,1\}^{|q|}$, and returns $T_1[K_p]$ to $A_{(1-k)II+kIII}$.
- $H(.)$ queries: If $T[ID_i]$ for query $ID_a$ is defined, then, $B$ returns $X_i$ as $x_a$. If this entry is not yet defined, it sets $ID_i \leftarrow ID_a$, $i \leftarrow i + 1$, and sets $T[ID_i] = (0, x_i = bP)$ if $i = t$. If $i \neq t$, $x_i \leftarrow Z_q$, and sets $T[ID_i] = (x_i, x_i = x_iP)$. Then, $B$ returns $X_i$ to $A_{(1-k)II+kIII}$.

- KeyExtract queries for $ID_a$: The algorithm $B$ looks up $T[ID_i]$, if $i = t$, it sets $bad_{KG} \leftarrow true$ and aborts the execution of $A_{(1-k)II+kIII}$. Otherwise, $B$ computes $x_a = x_iP$, and returns $x_a$ to $A_{(1-k)II+kIII}$. If $T[ID_i]$ for query $ID_a$ is not yet defined, it makes $H(.)$ query for $ID_a$.

- DelGen queries for $(w,ID_p)$ under an identity $ID_o$: If $ID_o \neq ID_t$, the algorithm $B$ uses $x_o$ to generate the delegation following the real DelGen algorithm. If $ID_o = ID_t$, $B$ chooses $h_0 \leftarrow Z_q$ and $U_o \leftarrow \mathbb{G}$, and computes $K_o = e(U_o, P)e(H(ID_o), P_{pub})^{-h_o}$. If $T_0[w||ID_p, K_o]$ has already been defined, then, $B$ sets $bad_{DG} \leftarrow true$ and halts; otherwise, it sets $T_0[w||ID_p, K_o] \leftarrow h_0$. Then, $B$ returns the delegation $(w,K_o,U_o,h_0)$ to the adversary $A_{(1-k)II+kIII}$.

- PSign queries for a message $m$ under proxy signer’s identity $ID_p$: The algorithm $B$ receives the delegation $(w,K_o,U_o,h_0)$ from $A_{(1-k)II+kIII}$. Next, $B$ checks if the delegation is valid for proxy signer’s identity $ID_p$ and the warrant $w$, if $m$ conforms to the warrant $w$ and if $|m| = t_2$. If all conditions hold and $ID_p \neq ID_t$, $B$ generates a valid signature following the real PSign algorithm. If $ID_p = ID_t$, $B$ chooses $\alpha \leftarrow Z_q$ and $U \leftarrow \mathbb{G}$, and computes $\beta = F_1(m)||F_2(F_1(m)) \oplus m$ and $K_p = e(U,P)e(H(ID_o), P_{pub})^{-h_o}e(H(ID_p), P_{pub})^{-\alpha}$. If $T_1[K_p]$ has already been defined, then, $B$ sets $bad_{PS} \leftarrow true$ and halts; otherwise, it computes $h_1 = \beta \oplus [\alpha]_2$, and sets $T_1[K_p] \leftarrow h_1$. Hence, the signature on the message $m$ w.r.t. $ID_o$ and $ID_p$ is $\theta = (w,U,\alpha,K_o,h_0)$.

- Finally, $A_{(1-k)II+kIII}$ outputs a forged identity-based proxy signature $\theta$ with original signer’s identity $ID_o$ and proxy signer’s identity $ID_p$. The forgery is non-trivial if $ID_o = ID_t$, and $A_{II}$ has not made KeyExtract query on input $ID_o$ and DelGen query on input $(w, ID_p)$ under identity $ID_o$, or similarly, if $ID_p = ID_t$, and $A_{II}$ has not made KeyExtract query on input $ID_p$ and PSign query on input $m$ under identity $ID_p$.

The probability of $B$ in returning a forged signature $(w,U,\alpha,K_o,h_0)$ is $\epsilon_1 = \Pr[E_1] \Pr[E_2]E_1$ which is computed as follows. First of all, we define events $E_1$ and $E_2$.

- $E_1$: Algorithm $B$ does not abort as a result of signature simulation (as a result of KeyExtract and DelGen queries for $A_{II}$ and as a result of KeyExtract and PSign queries for $A_{III}$).

- $E_2$: Adversary $A_{(1-k)II+kIII}$ returns a non-trivial forgery.

To lower-bound the probability of $B$, $\Pr[E_1]$, we need to compute the probability that $B$ does not abort in signature simulation; $\eta = \Pr[E_1] = \Pr[-bad_{KE}]\Pr[-bad_{DG}||-bad_{KE}] + k \Pr[-bad_{PS}||-bad_{KE}]$, where events $bad_{KE}$, $bad_{DG}$ and $bad_{PS}$ indicate that $B$ aborts in signature simulation as a result of any of $A_{(1-k)II+kIII}$'s KeyExtract, DelGen and PSign queries, respectively. These probabilities are computed as follows.

Claim 1. $\Pr[-bad_{KE}] \geq \frac{1}{q_H}$.

Proof. $\Pr[-bad_{KE}]$ is the probability that $B$ does not abort as a result of $A_{(1-k)II+kIII}$’s KeyExtract queries. The algorithm $B$ does not abort at answering to KeyExtract queries when $H(ID_o) = bP$ for $A_{II}$ which means that $ID_o = ID_t$ (when $H(ID_p) = bP$ for $A_{II}$ which means that $ID_p = ID_t$), and the probability of this event is at least $\frac{1}{q_H}$.

Claim 2. $\Pr[-bad_{DG}||-bad_{KE}] \geq 1 - q_d((q_d + q_{H_0})2^{-|G_r|}) - q_d^22^{-|G_r|}$.

Proof. Events $-bad_{KE}$ and $-bad_{DG}$ are independent, so $\Pr[-bad_{DG}||-bad_{KE}] = \Pr[-bad_{DG}]$. The value of $\Pr[-bad_{DG}]$ is the probability that $B$ does not abort as a result of DelGen queries. The algorithm $B$ aborts at answering to a DelGen query if $bad_{DG}$ is set to true which means that there is a conflict
Claim 3. \( \Pr[\neg \text{bad}_{PS} | \neg \text{bad}_{KE}] \geq 1 - q_s((q_s + q_{H_U})2^{2^{\lceil \log T \rceil}} - q_s^22^{2^{\lceil \log T \rceil}}). \)

Proof. Events \( \neg \text{bad}_{KE} \) and \( \neg \text{bad}_{PS} \) are independent, so \( \Pr[\neg \text{bad}_{PS} | \neg \text{bad}_{KE}] = \Pr[\neg \text{bad}_{PS}]. \) The value of \( \Pr[\neg \text{bad}_{PS}] \) is the probability that \( B \) does not abort as a result of PSign queries. The algorithm \( B \) aborts at answering to a PSign query if \( \text{bad}_{PS} \) is set to true which means that there is a conflict in the table \( T_1[.] \). The probability of finding a conflict in \( T_1[.] \) for one PSign query \( K_p \) equals the probability that \( K_p \) generated in a PSign simulation has been occurred by chance in a previous query to the oracle \( H_1(\cdot) \).

Since there are at most \( q_{HT} + q_s \) entries in the table \( T_1[.] \) and the number of \( K_p \), uniformly distributed in \( G_T \), is \( 2^{\lceil \log T \rceil} \), the probability of this event for one PSign query is at most \((q_{HT} + q_s)2^{2^{\lceil \log T \rceil}}. \) Hence, the probability of this event for \( q_d \) queries is at most \( q_d(q_{HT} + q_s)2^{2^{\lceil \log T \rceil}}. \) In addition, this probability includes the probability that \( B \) previously used the same randomness \( K_p \), uniformly distributed in \( G_T \), in one PSign simulation. Since there are at most \( q_s \) PSign simulations, this probability is at most \( q_s2^{2^{\lceil \log T \rceil}}. \) Therefore, for \( q_s \) PSign queries the probability of this event is at most \( q_s^22^{2^{\lceil \log T \rceil}}. \)

Claim 4. \( \Pr[E_2 | E_1] \geq \epsilon. \)

Proof. The value of \( \Pr[E_2 | E_1] \) is the probability that \( A_{(1-k)II+kIII} \) returns a valid forgery provided that \( B \) does not abort as a result of \( A_{(1-k)II+kIII} \)’s KeyExtract, DelGen and PSign queries. If \( B \) did not abort as a result of \( A_{(1-k)II+kIII} \)’s queries, all its responses to those queries are valid. Therefore, by hypothesis \( A_{(1-k)II+kIII} \) will produce a non-trivial forgery with probability at least \( \epsilon. \)

Therefore, the probability that \( B \) returns \( (w, U, \alpha, K_o, h_0) \) is at least \( \epsilon_2 \geq \frac{1}{q_m}(1 - k)(q_s + q_{H_U})2^{2^{\lceil \log T \rceil}} - kq_s2^{2^{\lceil \log T \rceil}}(1 - (1 - k)q_d((q_d + q_{H_U})2^{2^{\lceil \log T \rceil}} - (1 - k)q_s2^{2^{\lceil \log T \rceil}}) \geq \frac{1}{q_m} - ((1 - k)q_d(2q_d + q_{H_U})2^{2^{\lceil \log T \rceil}} - kq_s(q_s + q_{H_U})2^{2^{\lceil \log T \rceil}}. \) Since \( H_k \) is a random oracle for \( k \in \{0, 1\} \), the probability of the event that \( h_k = H_k(kK_P + (1 - k)(w||ID_p, K_o)) \) for \( k \in \{0, 1\} \) is less than \( 2^{-\lceil \log q \rceil + 1}, \) unless they are asked during the attack. Hence, in what follows it is likely that queries \( kK_P + (1 - k)(w||ID_p, K_o) \) for \( k \in \{0, 1\} \) are asked during a successful attack. The lower bound of probability of producing a non-trivial forgery after making queries to \( H_0 \) and \( H_1 \) oracles is \( \epsilon_2 \geq \epsilon_1 - 2^{-\lceil \log q \rceil + 1}. \) Then, \( B \) uses the oracle replay technique \[54\] to solve the CDH problem.

Algorithm \( B \) employs two copies of \( A_{(1-k)II+kIII} \), guesses a fixed index \( 1 \leq j \leq (q_{H_k} + (1 - k)q_d + q_s), \) and hopes that \( j \) be the index of query \( kK_P + (1 - k)(w||ID_p, K_o) \) to oracle \( H_k \) for which \( A_{(1-k)II+kIII} \) forges a proxy signature, and the probability of a good guess by chance is \( \frac{1}{(q_{H_k} + (1 - k)q_d + q_s)}. \) Algorithm \( B \) gives the same system parameters, the same identities and the same sequence of random bits to the two copies of \( A_{(1-k)II+kIII} \), and responds with the same random answers to their queries for the oracles until they ask the oracle \( H_k \) for \( j \thinspace \text{th} \) query. At that point (the \( j \)th query to the oracle \( H_k \)), \( B \) gives two random answers \( h_k \) and \( h'_k \) such that \( h_k \neq h'_k \) to the hash queries \( H_k(\text{forking}) \). Hence, \( B \) obtains two proxy signatures \( (w, U, \alpha, K_o, h_0) \) and \((w, U', \alpha', K_o, h'_0) \) after \( A_{(1-k)II+kIII} \) asks the same query \( kK_P + (1 - k)(w||ID_p, K_o) \) from \( H_k \). We employ Splitting Lemma to compute the probability of \( B \) in returning these two valid forgeries.

It is assumed that \( S \) denotes the set of successful executions of \( A_{(1-k)II+kIII} \) when \( B \) simulates the signature scheme, and the success probability of \( A_{(1-k)II+kIII} \) in returning a non-trivial forgery is taken over the space \((X, Y_k)\), where \( X \) is the set of random bits and random oracle responses that \( A_{(1-k)II+kIII} \) takes up except for randomness related to the oracle \( H_k \), and \( Y_k \) is the set of random oracle responses to the oracle
H_k. Hence, we have Pr[(X, Y_k) ∈ S] = ε2. With Splitting Lemma, we split the randomness Y_k related to H_k to (Y'_k, h_k), where Y'_k is the set of all random responses to different queries of H_k except for jth query whose answer is denoted as h_k. The Splitting Lemma ensures the existence of a subset of executions Ω such that Pr[Ω|S] ≥ γ δ = 1 2, and for each (X, Y_k) ∈ S, Pr[h_k((X, Y'_k), h'_k) ∈ S] ≥ δ − γ = e2 2(qm_k + (1 − k)q_d + kq_s). If B replays the attack with fixed (X, Y'_k) and a randomly chosen h'_k ∈ {0, 1}^|y|, it gets another successful pair ((X, Y'_k), h'_k) such that h_k ≠ h'_k with probability e2(1 − 2^−|y|) 2(qm_k + (1 − k)q_d + kq_s), where e2 ≥ 1 − 2^−|y|+1.

After two successful executions of A_{1 − k} II + k III, B obtains ((X, Y'_k), h_k) and ((X, Y'_k), h'_k), h_k ≠ h'_k which means that it obtains two valid forgeries (w, U, α, K_o, h_0) and (w, U', α', K_o, h'_0) with probability ε' ≥ e2(1 − 2^−|y|) 2(qm_k + (1 − k)q_d + kq_s), where e2 ≥ 1 − 2^−|y|+1.

From valid forgeries (w, U, α, K_o, h_0) and (w, U', α', K_o, h'_0), B computes abP as abP = U − U' (α' − α) if k = 1. In this case, we have the adversary A_{II} and since h_1 ≠ h'_1 and α = [β ⊕ h_1]_10, we have α ≠ α', while other random values especially (x_p, x_o, k_o, k_p) are the same and h_0 = h'_0. If k = 0, B computes abP as abP = (U − U') (h_0 − h'_0). In this case, we have the adversary A_{II} and h_0 ≠ h'_0, while other random values especially (x_p, x_o, k_o, k_p) are the same and h_1 = h'_1 or equivalently α = α'.

Algorithm B’s run-time t' is twice of A_{1 − k} II + k III’s run-time, t, plus the time required to respond to hash queries, q_e KeyExtract, q_d DelGen and q_a PSign queries. We assume that an exponentiation in G_T takes time t_e, a scalar multiplication in G and G_T take time t_m and t_m_T respectively, and a pairing computation takes time t_p, while other operations take zero time. Each random oracle or KeyExtract query takes time t_m, each DelGen simulation takes time t_m + t_p + t_p + t_e, and a proxy signature with message recovery simulation takes time 1t_p + 2t_e + 2t_m_T, therefore B’s run-time is t' ≤ 2(t + q_e(t_m) + q_d(t_m + t_p + t_e) + q_a(1t_p + 2t_e + 2t_m_T)). This completes the proof.

5 Comparison

The comparison for some short identity-based proxy signatures is summarized in Table 2. The comparison is in terms of DeleGen-Cost, DeleVer-Cost, PSign-Cost and PVer-Cost which are dominating computational cost in delegation generation, delegation verification, proxy signature generation and proxy signature verification, respectively. The computational cost for proxy secret key generation algorithm is ignored in the comparison. For the sake of comparison, we consider expensive operations, and we consider pre-computations in presenting the number of operations such that the same operations during generation and verification of different signatures are computed one time and before each algorithm. For example, terms e(H(ID_o), P_{pub}) and e(H(ID_p), P_{pub}) in our scheme are the same for different signatures, so, they can be computed before signature generation and verification. As a consequence, we ignore them when we present the number of operations in Table 2. Table 1 presents definitions of symbols used in comparison.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>pairing evaluation</td>
</tr>
<tr>
<td>E_T</td>
<td>exponentiation in group G_T</td>
</tr>
<tr>
<td>m_G</td>
<td>scalar multiplication in G</td>
</tr>
<tr>
<td>m_{G_T}</td>
<td>scalar multiplication in G_T</td>
</tr>
<tr>
<td>l_2</td>
<td>the number of bits of a message m</td>
</tr>
<tr>
<td>l_3</td>
<td>the number of bits of a warrant w</td>
</tr>
</tbody>
</table>

Table 1. Symbols’ definitions in comparison

In Table 2, we do not consider two identity-based proxy signature schemes with message recovery [44,45] since non of them is secure.
As shown in Table 2, if we consider just pairing computation, the most expensive operation [55], cost to simplify the computational cost comparison, our scheme is more efficient than those presented in [25, 20, 24, 22], and approximately as efficient as those presented in [19, 21].

<table>
<thead>
<tr>
<th>Scheme</th>
<th>DeleGen</th>
<th>DeleVer</th>
<th>PSign</th>
<th>PVer</th>
<th>Signature Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Scheme</td>
<td>$1E_T + 1m_G$</td>
<td>$1P + 1E_T + 1m_G$</td>
<td>$1E_T + 1m_G + 1m_T$</td>
<td>$1P + 2m_G + 2E_T$</td>
<td>$2</td>
</tr>
<tr>
<td>Shim [25]</td>
<td>$3m_G + 1P$</td>
<td>$3P + 1m_G$</td>
<td>$3m_G$</td>
<td>$3P + 2m_G$</td>
<td>$3</td>
</tr>
<tr>
<td>Wu et al. [20]</td>
<td>$2m_G$</td>
<td>$2P$</td>
<td>$2m_G$</td>
<td>$3P$</td>
<td>$3</td>
</tr>
<tr>
<td>Ji et al. [22]</td>
<td>$2m_G$</td>
<td>$2P + 1m_G$</td>
<td>$2m_G$</td>
<td>$1m_G + 2P$</td>
<td>$3</td>
</tr>
<tr>
<td>Xu, Zhang and Feng [24]</td>
<td>$2m_G + 1E_T$</td>
<td>$2m_G + 1E_T$</td>
<td>$2m_G + 1E_T$</td>
<td>$1P + 2E_T$</td>
<td>$2</td>
</tr>
<tr>
<td>Gu and Zhu [21]</td>
<td>$1m_G + 2E_T$</td>
<td>$1P + 2E_T$</td>
<td>$1m_G + 2E_T + 2m_G + 2E_T$</td>
<td>$1P$</td>
<td>$4</td>
</tr>
</tbody>
</table>

Table 2. Comparison between our scheme and some existing schemes

As shown in Table 2, our scheme is shorter than all other existing identity based proxy signature schemes since $l_2$, the number of bits of a signed message, is omitted in its size. To make it clearer, we write signature size of schemes in terms of bits in Table 3. In the following, let’s assume that $q$ be a 170-bit prime, and using any of the families of curves described in [56] to have security the same as the security of the standard 1024-bit RSA signature, each element of the group $G$ and $G_T$ will be 171 and 1024 bits, respectively.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Actual Signature Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Scheme</td>
<td>$512 + l_3$</td>
</tr>
<tr>
<td>Shim [25]</td>
<td>$513 + l_2 + l_3$</td>
</tr>
<tr>
<td>Wu et al. [20]</td>
<td>$513 + l_2 + l_3$</td>
</tr>
<tr>
<td>Ji et al. [22]</td>
<td>$513 + l_2 + l_3$</td>
</tr>
<tr>
<td>Xu, Zhang and Feng [24]</td>
<td>$513 + l_2 + l_3$</td>
</tr>
<tr>
<td>Zhang and Zou [19]</td>
<td>$2219 + l_2 + l_3$</td>
</tr>
<tr>
<td>Gu and Zhu [21]</td>
<td>$2219 + l_2 + l_3$</td>
</tr>
</tbody>
</table>

Table 3. Signature-size comparison (in bits)

As shown in Tables 3 and 2, our scheme is the shortest compared to all other existing identity based proxy signature schemes since the message $m$ is not required to be transmitted with the signature.

6 Conclusion

In this paper, we showed that Yoon et al.’s identity-based proxy signature scheme with message recovery does not have message recovery property despite their authors’ claim, therefore, it is not short. Hence, there was no provably secure short identity-based proxy signature. Then, we proposed a short identity-based proxy signature scheme with message recovery feature which is secure under CDH problem in the random oracle model. As shown in comparison, the size of our signature is reduced compared to identity-based proxy signatures since the original message is not transmitted with the signature. This primitive is useful where bandwidth is one of the crucial concern. Our scheme is designed for messages with fixed length and it can be modified to provide partial message recovery property with the technique presented by Tso et al. in [7]. Furthermore, our scheme is more efficient than (as efficient as) previous identity-based proxy signature schemes.
References


