A Secure Proxy Signature Scheme Based on the Hardness of the Decisional Diffie-Hellman Problem

Constantin Popescu

Department of Mathematics and Computer Science,
University of Oradea,
Oradea 410087, Romania,
cpopescu@uoradea.ro

Abstract: In this paper we present a secure proxy signature scheme, which allows an original signer to delegate his/her signing capability to a proxy signer. Then the proxy signer can sign a message on behalf of the original signer. The proposed proxy signature scheme is based on the hardness of the decisional Diffie-Hellman problem. We give the formal definition and security model of a proxy signature scheme and prove its security in our security model. Our proxy signature scheme does not use bilinear pairings, which results in greater efficiency and ease of implementation.

Keywords: Proxy signature, proxy signer, security model, delegation, warrant.

1. Introduction

The notion of proxy signature was introduced by Mambo, Usuda and Okamoto in 1996 [1]. The proxy signature scheme allows the original signer to delegate his/her signing right to the proxy signer to sign a message on behalf of the original signer. Afterwards, a verifier, which knows the public keys of the original signer and the proxy signer, can verify the validity of the proxy signature issued by the proxy signer. Based on the delegation type, the proxy signature schemes are classified in full delegation, partial delegation and delegation by warrant. In a full delegation proxy signature scheme, a proxy signer uses the same private key as the original signer and generates a proxy signature as the original signer does. The disadvantage of the full delegation comes from the difficulty of distinguishing between the original signer and the proxy signer. In the partial delegation proxy signature scheme, an original signer derives a proxy key from his private key and sends it to a proxy signer in a secure channel. In a proxy signature scheme with delegation by warrant, the original signer gives a proxy signer a special message, namely, warrant. A warrant certifies that the proxy signer is legal and consists of signers' identity, delegation period and the types of the message on which the proxy signer can sign. Also, proxy signature schemes can be classified as proxy-unprotected and proxy-protected schemes. In an proxy-protected scheme, the original signer cannot forge a proxy signature in the name of the proxy signer. The proxy-protected schemes provide more security level than the proxy-unprotected signature schemes. A lot of proxy signature schemes [6], [7] and some ID-based proxy signature schemes with special features were proposed, such as identity-based multi-proxy signature [8], [9], identity-based strong designated verifier proxy signature [10], [11]. Cao and Cao [9] claimed that their scheme is provably secure in the random oracle model. However, Xiong et al. [12] proved that their scheme is not secure under their security model. The first proxy signature scheme based on the factoring integer problem is proposed by Shao [13], in 2003. Recently, Zhou et al. [14] proposed two efficient proxy protected signature schemes. Their first scheme is based on RSA [15] assumption and the second scheme is based on the integer factorization problem. Zhou et al. [14] claim that their schemes are more efficient than other schemes. However, Park et al. [16] point out their schemes are insecure. Moreover, Liu et al. [17] point out that Zhou et al.'s [14] schemes are vulnerable to the undelegated proxy signature attack: any attacker without the delegation of the original signer can generate a valid proxy signature. Xue et al. [18] proposed two proxy signature schemes based on the difficulty of factorings of large integers without formal security proofs. Recently, Shao [19] proposed proxy protected signature scheme based on RSA. Also, most proxy signature schemes are based on the difficulty of discrete logarithm problem [20] or elliptic curve discrete logarithm problem [21]. Chen et al. proposed in [20] a proxy signature scheme based on the Digital Signature Algorithm (DSA). Mambo et al. [1], [22] proposed three proxy signature schemes based on ElGamal's signature scheme [23], Schnorr's signature scheme [24], and Okamoto's signature scheme [25]. Proxy signature schemes are useful in many applications [29], [30] such as electronic payment systems [2], [3] and wireless networks [4], [5].

In this paper we propose a secure proxy signature scheme based on the hardness of the decisional Diffie-Hellman problem. The proposed proxy signature scheme is derived from the Goh et al.'s signature scheme [26]. Our proxy signature scheme inherits the strength security properties of the signature scheme proposed in [26].

The rest of this paper is organized as follows. In the next section we review the model of a proxy signature scheme. Then we present our proxy signature scheme in the section 3. Furthermore, we discuss some aspects of security in the section 4. The section 5 concludes the work of our paper.

2. Preliminaries

In this section, we briefly review the model and the security properties of a proxy signature scheme and the decisional Diffie-Hellman problem.

2.1 The model of our proxy signature scheme

In this section, we describe a formal definition of the proposed proxy signature scheme. Our proxy signature scheme has four entities: a Key Generation Center, an original signer, a proxy signer and a verifier.

Definition 1. The proposed proxy signature scheme is comprised of six algorithms: Master Key Generation, Partial Key Generation, User Key Generation, Proxy Key Generation, Proxy Signature Generation, Verify:

Master Key Generation: Given a security parameter l as input, a random algorithm outputs a master public key (g, h, y_1, y_2) and a master secret key t. The algorithm is assumed to be run by a Key Generation Center.

Partial Key Generation: This is a random algorithm which takes as input the master secret key t and the user's identity $ID \in \{0,1\}^*$ and generates a user partial key. This algorithm is run by the Key Generation Center once for each user (the original signer and the proxy signer) and the partial private key is assumed to be distributed securely to the corresponding user.

User Key Generation: The algorithm takes as input the master public key (g, h, y_1, y_2), the original signer's identity ID_{os} and the proxy signer's identity ID_{ps} and outputs a secret key $x_{ID_{os}}$ of the original signer and the corresponding public key (y_1', y_2'). The algorithm also outputs a secret key $x_{ID_{ps}}$ of the proxy signer and the corresponding public key (y_1'', y_2''). This algorithm is assumed to be run by each user (the original signer and the proxy signer) in the system.

Proxy Key Generation: The input of this algorithm includes the original signer's identity ID_{os}, the proxy signer's identity ID_{ps} with a warrant w. The warrant w records the delegation policy, limits of authority, valid periods of delegation and proxy signatures, and the identities of the proxy signer and the original signer. The algorithm also takes as input the user secret key, the user partial key of the original signer, the user secret key and the user partial key of the proxy signer. This algorithm is run by the original signer and the proxy signer interactively and outputs the proxy private key x_{ps}.

Proxy Sign ature Genera tion: For a message $m \in \{0,1\}^*$, the proxy signer computes the proxy signature σ by using his/her proxy private key x_{ps}.

Verify: This is a deterministic algorithm. Given a proxy signature σ, a verifier uses the public key of the original signer and the proxy signer, to check its validity and outputs 1 if σ is valid, otherwise outputs 0.

2.2 The ecisional Diffie-Hellman problem

The decisional Diffie-Hellman problem was described by Goh, Jareck, Katz and Wang in [26]. Let $\mathbb{G}$ be a finite and cyclic group of prime order q with generator g (the group operation is represented multiplicatively). The decisional Diffie-Hellman problem may be defined informally as follows [26]:

Definition 2. Given the generator g and the group elements g^x, g^y, g^{xy}, g^z, the decisional

Diffie-Hellman problem is to distinguish between tuples of the form (g, g^x, g^y, g^{xy}) for random $x, y \in \mathbb{Z}_q$ and tuples of the form g, g^x, g^y, g^z for random $x, y, z \in \mathbb{Z}_q$.

The decisional Diffie-Hellman problem is hard in $\mathbb{G}$ if no efficient algorithm can distinguish with high probability between randomly generated tuples of these two types with high probability [26]. The decisional Diffie-Hellman problem may be defined formally as follows [26]:

Definition 3. A distinguishing algorithm $\mathcal{A}$ is said (T, ε)-solve the decisional Diffie-Hellman problem in the group $\mathbb{G}$ if $\mathcal{A}$ runs in time at most T and furthermore

$$\begin{aligned} &| Pr[x, y, z \in \mathbb{Z}_q : \mathrm{A}(g, g^x, g^y, g^z) = 1] - \\ &-Pr[x, y \in \mathbb{Z}_q : \mathrm{A}(g, g^x, g^y, g^{xy}) = 1] | \geq \varepsilon. \end{aligned} \quad (1)$$

The group $\mathbb{G}$ is a (T, ε)-decisional Diffie-Hellman group if no algorithm (T, ε)-solves the decisional Diffie-Hellman problem in $\mathbb{G}$.

2.3 The se curity mod el of our proxy signature scheme

The strong unforgeability is defined using two algorithms: algorithm $\mathcal{A}$ and algorithm $\mathcal{B}$. Algorithm $\mathcal{B}$ simulates the hash and signing query for algorithm $\mathcal{A}$ as follows:

– Hash queries: A query H serves as input for algorithm $\mathcal{B}$. Algorithm $\mathcal{B}$ outputs a value of a previous hash query if this is defined before or a random value from $\{0,1\}^l$, otherwise.

– Proxy signing queries: Algorithm $\mathcal{A}$ request a proxy signature on the message m under the delegation warrant w. In response, algorithm $\mathcal{B}$ outputs the previously proxy signature, if the message m was signed before. Otherwise, algorithm $\mathcal{B}$ simulates a proof that (g, h, y_1'', y_2'') is a decisional Diffie-Hellman tuple.

Algorithm $\mathcal{B}$ runs in time T' and the success probability of algorithm $\mathcal{B}$ to solve an instance of the decisional Diffie-Hellman problem in $\mathbb{G}$ is ε'.

2.4 Security properties of our proxy signature scheme

Mambo, Usuda and Okamoto defined the basic security properties of a proxy signature scheme [1], [22]. Our proxy signature scheme should satisfy the following requirements:

– Verifiability: From a proxy signature, a verifier can be convinced of the original signer's agreement on the signed message.

– Strong unforgeability: A proxy signer can create a valid proxy signature on behalf of the original signer. However, the original signer and any third party cannot generate a valid proxy signature with the name of proxy signers.

– Strong identifiability: From a proxy signature, anyone can determine the identity of the corresponding proxy signer.

– Strong undeniability: Once a proxy signer generates a valid proxy signature on behalf of the original signer, the proxy signer cannot deny his signature generation against anyone.

– Prevention of misuse: It should be confident that the proxy key pair cannot be used for other purposes. In the case of misuse, the responsibility of proxy signers should be determined explicitly.

3. Our Proxy Signature Scheme

In the proposed proxy signature scheme we need a third party called Key Generation Center to help a user (the original signer and the proxy signer) in order to generate his private key. However, the Key Generation Center does not have access to a user’s full private key. The Key Generation Center just generates a user’s partial private key using the user’s identity. A user computes his full private key by combining his partial private key and a secret value chosen by himself. The public key of a user is computed from the Key Generation Center’s public parameters and the secret value of the user and it is published by the user himself. The proposed proxy signature scheme is derived from the Goh et al.'s signature scheme [26].

3.1 Master key generation

This algorithm is assumed to be run by a Key Generation Center. Given a security parameter $l \in \mathbb{Z}$, the algorithm is as follows:

1. Generate a prime q.
2. Assume that $\mathbb{G}$ is a finite and cyclic group of prime order q with generator g.
3. Choose a cryptographic hash function $H:\{0,1\}^* \to \{0,1\}^l$. The parameter l should be set such that the probability of forgery is $q_h \cdot 2^{-l}$, where q_h represents the number of times the adversary calculates the hash function. See [26] for more details.
4. Choose a random $h \in \mathbb{G}$.
5. Select a master secret key $t \in \mathbb{Z}_q^*$ and computes $y_1 = g^t, y_2 = h^t$. The master secret key is t and the corresponding master public key is the tuple (g,h,y_1,y_2).

3.2 Partial key generation

1. Key Generation Center first generates the partial key of the original signer:

– Chooses a random number $r \in \mathbb{Z}_q^*$.

– Computes $A = g^r, B = h^r$ and $c = H(A,B,ID_{os})$, where ID_{os} is the identity of the original signer.

– Computes $s = ct + r \bmod q$.

Key Generation Center transmits the pair (c,s) to the original signer secretely. The original signer verifies (c,s) as follows:

– Computes $\overline{A} = g^s y_1^{-c}$ and $\overline{B} = h^s y_2^{-c}$.

– Checks whether $c = H(\overline{A},\overline{B},ID_{os})$.

2. Then Key Generation Center generates the partial key of the proxy signer:

– Chooses a random number $r' \in \mathbb{Z}_q^*$.

– Computes $A' = g^{r'}, B' = h^{r'}$ and $c' = H(A',B',ID_{ps})$, where ID_{ps} is the identity of the proxy signer.

– Computes $s' = c't + r' \bmod q$.

Key Generation Center transmits the pair (c',s') to the proxy signer secretly. The proxy signer verifies (c',s') as follows:

– Computes $\overline{A'} = g^{s'} y_1^{-c'}$ and $\overline{B'} = h^{s'} y_2^{-c'}$.

– Checks whether $c' = H(\overline{A'},\overline{B'},ID_{ps})$.

3.3 User key generation

1. The original signer generates his/her private key and the corresponding public key. The original signer does the following:

– Picks a random number $x_{ID_{os}} \in \mathbb{Z}_q^*$.

– Computes $y_1' = g^{s+x_{ID_{os}}}$ and $y_2' = h^{s+x_{ID_{os}}}$.

– The private key of the original signer is $x_{ID_{os}}$.

– The corresponding public key is the tuple (y_1', y_2').

2. The proxy signer generates his/her private key and the corresponding public key. The proxy signer does the following:

– Picks a random number $x_{ID_{ps}} \in \mathbb{Z}_q^*$.

– Computes $y_1'' = g^{s'+x_{ID_{ps}}}$ and $y_2'' = h^{s'+x_{ID_{ps}}}$.

– The private key of the proxy signer is $x_{ID_{ps}}$.

– The corresponding public key is the tuple (y_1'', y_2'').

3.4 Proxy key generation

The proxy signer generates his/her proxy private key. Firstly, the original signer signs the delegation warrant w and then sends the signature of w to the proxy signer. The delegation warrant w contains the delegation policy, including limits of authority, the message type to be signed, valid periods of delegation and proxy signatures, and the identities of the proxy signer and the original signer. The protocol between the original signer and the proxy signer is as follows:

1. The original signer generates the signature (d,s'') of the delegation warrant w:

– Chooses a random number $z \in \mathbb{Z}_q^*$.

– Computes $a = g^z, b = h^z$ and $d = H(a,b,w)$.

– Computes $s'' = d(s + x_{ID_{os}}) + z \bmod q$.

Then the original signer sends the tuple (d, s'', w) to the proxy signer.

2. The proxy signer verifies the signature (d, s'') as follows:

– Computes $\bar{a} = g^{s''}(y_1')^{-d}$ and $\bar{b} = h^{s''}(y_2')^{-d}$.

– Checks whether $d = H(\bar{a}, \bar{b}, w)$.

If $d = H(\bar{a}, \bar{b}, w)$ holds, the proxy signer computes the proxy private key $x_{ps} = s'' + x_{ID_{ps}} \bmod q$ and keeps it as his/her proxy signing key.

3.5 Proxy signature generation

The proxy signer signs a message $m \in \{0,1\}^*$ on behalf on the original signer using the proxy private key x_{ps}. The proxy signer does the following steps:

1. Randomly chooses an integer $k \in \mathbb{Z}_q$ and computes $A'' = g^k, B'' = h^k$.
2. Computes $c'' = H(A'', B'', m, w)$ and $s_{ps} = c''(s' + x_{ps}) + k \bmod q$.

The proxy signature of the message m is the tuple (m, w, s'', s_{ps}, c'').

3.6 Proxy signature verification

The verification of the proxy signature (m, w, s'', s_{ps}, c'') is carried out as follows:

1. Computes $\overline{A''} = g^{s_{ps}} g^{-c''s''}(y_1'')^{-c''}$ and $\overline{B''} = h^{s_{ps}} h^{-c''s''}(y_2'')^{-c''}$.
2. Checks whether $c'' = H(\overline{A''}, \overline{B''}, m, w)$.

The verifier accepts the proxy signature (m, w, s'', s_{ps}, c'') if and only if the above equation holds.

4. Security Analysis and Comparisons

In this section we present aspects of security and efficiency of the proposed proxy signature scheme. The security of our proxy signature scheme is based on the hardness of the decisional Diffie-Hellman problem.

4.1 Correctness of the proposed proxy signature

Firstly, we prove the correctness of the signature (d, s'') of the delegation warrant w.

Theorem 1. The pair (d, s'') is a valid signature of the delegation warrant w.

Proof. We have to check that $d = H(\bar{a}, \bar{a}, w)$, which is equivalent with $\bar{a} = a$ and $\bar{b} = b$. We have:

$$\begin{aligned}\bar{a} &= g^{s''}(y_1')^{-d}\\ &= g^{ds+dx_{ID_{os}}+z} g^{-sd-dx_{ID_{os}}}\\ &= g^{ds+dx_{ID_{os}}+z-sd-dx_{ID_{os}}}\\ &= g^{z}\\ &= a.\end{aligned}$$

Also,

$$\begin{aligned}\bar{b} &= h^{s''}(y_2')^{-d}\\ &= h^{ds+dx_{ID_{os}}+z} h^{-sd-dx_{ID_{os}}}\\ &= h^{ds+dx_{ID_{os}}+z-ds-dx_{ID_{os}}}\\ &= h^{z}\\ &= b.\end{aligned}$$

Secondly, we prove the correctness of the proxy signature (m, w, s'', s_{ps}, c'') of the message m.

Theorem 2. If the proxy signature (m, w, s'', s_{ps}, c'') is generated by the proxy signer correctly, then it will pass the proxy signature verification.

Proof. We have to prove that $c'' = H(\overline{A''}, \overline{B''}, m, w)$, which is equivalent with $\overline{A''} = A''$ and $\overline{B''} = B''$. Obviously, the relations follows from:

$$\begin{aligned}\overline{A''} &= g^{s_{ps}} g^{-c''s''}(y_1'')^{-c''}\\ &= g^{c''s'+c''x_{ps}+k} g^{-c''s''} g^{-c''s'-c''x_{ID_{ps}}}\\ &= g^{c''s'+c''(s''+x_{ID_{ps}})+k-c''s''-c''s'-c''x_{ID_{ps}}}\end{aligned}$$

$$= g^{c''s'+c''s''+c''x_{ID_{ps}}+k-c''s''-c''s'-c''x_{ID_{ps}}}$$

$$= g^{k}$$

$$= A''.$$

Similarly,

$$\overline{B''} = h^{s_{ps}} h^{-c''s''} (y_2'')^{-c''}$$

$$= h^{c''s'+c''x_{ps}+k} h^{-c''s''} h^{-c''s'-c''x_{ID_{ps}}}$$

$$= h^{c''s'+c''(s''+x_{ID_{ps}})+k-c''s''-c''s'-c''x_{ID_{ps}}}$$

$$= h^{k}$$

$$= B''.$$

4.2 Verifiability

The verifier obtains the proxy signature (m, w, s'', s_{ps}, c'') and then the identities of the original signer and the proxy signer, valid periods of delegation from w. Also, the verifier can verify the proxy signature (m, w, s'', s_{ps}, c'') by using the following equations:

1. Computes $\overline{A''} = g^{s_{ps}} g^{-c''s''} (y_1'')^{-c''}$ and $\overline{B''} = h^{s_{ps}} h^{-c''s''} (y_2'')^{-c''}$.
2. Checks whether $c'' = H(\overline{A''}, \overline{B''}, m, w)$.

Therefore, the verifiability is derived from the correctness of the proxy signature (see Theorem 2). Also, because the delegation warrant w contains the identity information and the limitation of the delegated signing capability, the verifiability is satisfied.

4.3 Strong unforgeability

Inspired by the work [26], we provide a proof of unforgeability of our proxy signature scheme.

Theorem 3. Let $\mathbb{G}$ be a cyclic and finite group of prime order q with generator g and assume $\mathbb{G}$ is a (T', ε')-decisional Diffie-Hellman group such that exponentiation in $\mathbb{G}$ takes t_e. Assume we have an algorithm $\mathcal{A}$ that runs in time at most T, makes at most q_h hash queries, at most q_s signing queries and outputs a valid proxy signature (m, w, s'', s_{ps}, c'') with probability at least ε. If there exists another algorithm $\mathcal{B}$ who can solve an instance of the decisional Diffie-Hellman problem in $\mathbb{G}$, then the proxy signature (m, w, s'', s_{ps}, c'') is $(T, q_h, q_s, \varepsilon)$-secure for $T \approx T' - \mathcal{O}(q_s \cdot t_e)$ and $\varepsilon = \varepsilon' + q_h/q + q_h/2^l$.

Proof. We use algorithm $\mathcal{A}$, like in [26], to construct an algorithm $\mathcal{B}$ which will run in time T' and solves the decisional Diffie-Hellman problem with probability ε'. The tuple (g, h, y_1'', y_2'') serves as input for algorithm $\mathcal{B}$ and its goal is to determine whether this is a random tuple or a Diffie-Hellman tuple (see Definition 2). Algorithm $\mathcal{B}$ will response the queries of algorithm $\mathcal{A}$ in the following way:

1. Hash queries: In response to a query $H(\overline{A''}, \overline{B''}, m, w)$, algorithm $\mathcal{B}$ checks if the output of H on this input is a previous hash query or is a part of proxy signature query, denoted h_{qry}. If $H(\overline{A''}, \overline{B''}, m, w) = h_{qry}$, then algorithm $\mathcal{B}$ returns the value h_{qry}. Otherwise, algorithm $\mathcal{B}$ responds with a random value from $\{0,1\}^l$.
2. Proxy signing queries: Suppose that algorithm $\mathcal{A}$ issues a proxy signature query for a message m under the delegation warrant w. Algorithm $\mathcal{B}$ checks if the message m was signed before. If so, algorithm $\mathcal{B}$ outputs the previously generated signature. Otherwise, algorithm $\mathcal{B}$ simulates a proof that (g, h, y_1'', y_2'') is a decisional Diffie-Hellman tuple as follows: algorithm $\mathcal{B}$ chooses three random values $c'' \in \{0,1\}^l$, $s'', s_{ps} \in \mathbb{Z}_q^*$ and computes $\overline{A''} = g^{s_{ps}} g^{-c''s''} (y_1'')^{-c''}$ and $\overline{B''} = h^{s_{ps}} h^{-c''s''} (y_2'')^{-c''}$. If a query H with $H(\overline{A''}, \overline{B''}, m, w) \neq c''$, then algorithm $\mathcal{B}$ aborts, otherwise, algorithm $\mathcal{B}$ sets $H(\overline{A''}, \overline{B''}, m, w) = c''$ and outputs the proxy signature (m, w, s'', s_{ps}, c'').

Suppose that algorithm $\mathcal{A}$ outputs its forgery $(\overline{m}, \overline{w}, \overline{s''}, \overline{s_{ps}}, \overline{c''})$. If $\overline{c''} = H(\overline{A''}, \overline{B''}, \overline{m}, \overline{w})$, then algorithm $\mathcal{B}$ outputs 1, otherwise $\mathcal{B}$ outputs 0.

We have two possibilities:

– If (g,h,y_1'',y_2'') is a decisional Diffie-Hellman tuple, then the probability that algorithm $\mathcal{B}$ aborts is at most q_h/q, where q_h is the number of hash queries made by algorithm $\mathcal{A}$ of the form $H(\bullet,\bullet,m,w)$. Algorithm $\mathcal{A}$ outputs a valid forgery with probability at least $\varepsilon - q_h/q$. In this case, algorithm $\mathcal{B}$ outputs 1.

– If (g,h,y_1'',y_2'') is a random tuple, then for any $\overline{A''},\overline{B''}$ and any query $H(\overline{A''},\overline{B''},m,w)$ made by algorithm $\mathcal{A}$ there is at most one value of c'' for which there exists an s_{ps} such that (see [26] for more details):

$$\overline{A''} = g^{s_{ps}} g^{-c''s''} (y_1'')^{-c''}$$

$$\overline{B''} = h^{s_{ps}} h^{-c''s''} (y_2'')^{-c''}.$$

Algorithm $\mathcal{A}$ outputs a valid forgery with probability at most:

$1/q + q_h/2^l$.

We have:

$$|\,Pr[x,y \in \mathbb{Z}_q : \mathcal{B}(g,g^x,g^y,g^{xy}) = 1] -$$

$$Pr[x,y,z \in \mathbb{Z}_q : \mathcal{B}(g,g^x,g^y,g^z) = 1]\,| \geqslant$$

$$\geqslant \varepsilon - q_h/q - q_h/2^l \geqslant \varepsilon'.$$

4.4 Strong undeniability

The valid proxy signature (m,w,s'',s_{ps},c'') contains the delegation warrant w, which must be verified in the verification phase. The delegation warrant w cannot be modified by the proxy signer. Thus once a proxy signer creates a valid proxy signature (m,w,s'',s_{ps},c'') of an original signer, he cannot repudiate this signature. Also, the equation $c'' = H(\overline{A''},\overline{B''},m,w)$ prevents the proxy signer from denying that he have signed the given message m, since the delegation warrant w is specified by the original signer in the proxy key generation phase.

4.5 Strong identifiability

A valid proxy signature (m,w,s'',s_{ps},c'') contains the delegation warrant w, which contains the identities and the public keys of the original signer and the proxy signer. Therefore, anyone can determine the identity of the proxy signer from the delegation warrant w.

4.6 Prevention of misuse

The delegation warrant w contains the signers' identity, the delegation period and the information about the type of the message can be signed by the proxy signer. Therefore, the proxy signer cannot sign other messages that have not been authorized by the original signer.

4.7 Comparisons

We summarize the computation time of the proxy signature schemes in Table 1. For security reasons, assume that q is 160 bits and the output size of the secure hash function H is 160 bits [27]. Also, we assume that h is the computation time of one hashing operation, m is the computation time of one modular multiplication in a 1024-bit modulo, p is the computation time for a paring operation, pm is the computation time of the multiplication of an element over an elliptic curve and e is the computation time of one modular exponentiation operation in a 1024-bit modulo. The pairing operation is the most time consuming operation in an elliptic curve cryptosystem. Catalano et al. [28] showed that one pairing operation is about 20 times more expensive than one modular exponentiation operation. Also, one modular exponentiation operation takes on 240 modular multiplication operations [20].

From the Table 1, we conclude that, no pairing operation is required in our proxy signature scheme, only four exponentiation operations, two modular multiplication operations and two hashing operations are involved in the key generation phase. The signature generation phase of our proxy signature scheme needs two exponentiation operations, one modular multiplication operation and one hashing operation. Also, only two exponentiation operations and one hashing operation are needed in the verification phase of the proposed proxy signature scheme.

Table 1. Computation time of the proxy signature schemes

Proxy Schemes	Key Generation	Signature Generation	Signature Verification
Shim [6]	3pm+2p+2h	1pm+1p+1h	2pm+2p+3h
Lee [4]	4e+6m+2h	3e+3m+2h	3e+3m+2h
Liu [7]	8pm+11p+4h	3pm+3p+4h	1pm+8p+2h
Shao [19]	4e+2m+2h	3e+1m+2h	2e+1m+3h
Xiong [12]	2p+2pm+1h	2p+2pm+1h	3p+1pm+1h
Our Scheme	4e+2m+2h	2e+1m+1h	2e+1h

5. Conclusions

In this paper we proposed a secure proxy signature scheme based on the hardness of the decisional Diffie-Hellman problem. We proved that our proxy signature scheme satisfies the following security requirements: verifiability, strong unforgeability, strong identifiability, strong undeniability and prevention of misuse. The proposed proxy signature scheme does not use bilinear pairings, which results in greater efficiency and ease of implementation.

REFERENCES

1. MAMBO, M., K. USUDA, E. OKAMOTO, **Proxy Signatures: Delegation of the Power to Sign Messages**, IEICE Transactions on Fundamentals, vol. E79-A, 1996, pp. 1338-1354.

2. POPESCU, C., **An Anony mous Mobile Payment System Base d on Bilinear Pairings**, Informatica, vol. 20, 2009, pp. 579-590.

3. POPESCU, C., **A Secur e and Efficient Off-line Electr onic Tr ansaction Protocol**, Studies in Informatics and Control, vol. 19, no. 1, 2010, pp. 27-34.

4. LEE, B., H. KIM, K. KIM, **Secure Mobile Agent u sing Str ong Nondesignated Proxy Sig nature**, Proceedings of the Australasian Conference on Information Security and Privacy, vol. 2119, 2001, pp. 474-486.

5. WANG, G., **Designated-Verifier Pr oxy Signatures for e-Commerce**, Proceedings of the IEEE 2004 International Conference on Multimedia and Expo (ICME 2004), 2004, pp. 1731-1734.

6. SHIM, K. A., **Short Designated Verifier Proxy S ignatures**, Computers and Electrical Engineering, vol. 37, 2011, pp. 180-186.

7. LIU, J., S. HUANG, **Identity-Based Threshold Proxy Signature from Bilinear Pai rings**, Informatica, vol. 21, 2010, pp. 41-56.

8. CAO, F., Z. CAO, **A Secure Identity-based Proxy Multi-signature Scheme**, Information Sciences, vol. 179, 2009, pp. 192-202.

9. CAO, F., Z. CAO, **A Secure Identity-based Multi-proxy Signature Scheme**, Computers and Electrical Engineering, vol. 35, 2009, pp. 86-95.

10. YU, Y., C. XU, X. ZHANG, Y. LIAO, **Designated Verifier Proxy Signature Scheme without Random Oracles**, Computers and Mathematics with Applications, vol. 57, 2009, pp. 1352-1364.

11. LEE, J.S., J. H. CHANG, D. H. LEE, **Forgery Attacks on Kang et al.'s Identity-based Strong Designated Verifier Signature Scheme and Its Improvement with Security Proof**, Computers and Electrical Engineering, vol. 36, 2010, pp. 948–954.

12. XIONG, H., J. HUA, Z. CHEN, F. LI, **On the Security of an Identity based Multi-proxy Signature Scheme**, Computers and Electrical Engineering, vol. 37, 2011, pp. 129–135.

13. SHAO, Z., **Proxy Signature Schemes based on Factoring**, Information Processing Letters, vol. 85, 2003, pp. 137-143.

14. ZHOU, Y., Z. CAO, R. LU, **Provably Secure Proxy-protected Signature Schemes based on Factoring**, Applied Mathematics and Computation, vol. 164, 2005, pp. 83-98.

15. RIVEST, R. L., A. SHAMIR, L. ADELMAN, **A Method for Obtain Digital Signatures and Public-key Cryptosystem**, Communication on ACM, vol. 21, 1978, pp. 120-126.

16. PARK, J. H., B. G. KANG, J. W. HAN, **Cryptanalysis of Zhou et al.'s Proxy-protected Signature Schemes**, Applied Mathematics and Computation, vol. 169, 2005, pp. 192-197.

17. LIU, Y., H. WEN, C. LIN, **Proxy-protected Signature Secure Against the Undelegated Proxy Signature Attack**, Computers and Electrical Engineering, vol. 33, 2007, pp. 177-185.

18. XUE, Q., Z. CAO, **Factoring based Proxy Signature Schemes**, Journal of Computational and Applied Mathematics, vol. 195, 2006, pp. 229-241.

19. SHAO, Z., **Provably Secure Proxy-protected Signature Schemes based on RSA**, Computers and Electrical Engineering, vol. 35, 2009, pp. 497-505.

20. CHEN, I., M. CHANG, Y. S. YEH, **Design of Proxy Signature in the Digital Signature Algorithm (DSA)**, Journal of Information Science and Engineering, vol. 22, 2006, pp. 965-973.

21. POPESCU, C., **A Secure Proxy Signature Scheme with Delegation by Warrant**, Studies in Informatics and Control, vol. 20, issue 4, 2011, pp. 373-380.

22. MAMBO, M., K. USUDA, E. OKAMOTO, **Proxy Signatures for Delegating Signing Operation**, Proceedings of the Third ACM Conference on Computer and Communications Security, ACM press., 1996, pp. 48-57.

23. ELGAMAL, T., **A Public Key Cryptosystem and Signature Scheme based on Discrete Logarithms**, IEEE Transactions on Information Theory, vol. 31, 1985, pp. 469-472.

24. SCHNORR, C. P., **Efficient Signature Generation by Smart Cards**, Journal of Cryptology, vol. 3, 1991, pp. 161-174.

25. OKAMOTO, T., **Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes**, Advances in Cryptology - CRYPTO'92, Springer - Verlag, 1983, pp. 31-53.

26. GOH, E. J., S. JARECKI, J. KATZ, N. WANG, **Efficient Signature Sc hemes with T ight Reductions to t he Diffie-Hellman Proble ms**, Journal of Cryptology, vol. 20, 2007, pp. 493-514.

27. NIST, **Secure Hash Signature St andard** (SHS), National Institute of Standards and Technology, FIPSP 180-2. 2002.

28. CATALANO, D., G. RUFALO, R. SCHIFANELLA, **A P2P Mar ket Place based on Aggregate Sign atures**, Proceedings of ISPA Workshops, 2005, pp. 54-63.

29. THORSTEINSSON, G., T. PAGE, A. NICULESCU, **Using Virtual Reality for Developing Design Communication**, Studies in Informatics and Control, vol. 17, issue 1, vol. 19, no. 1, 2010, pp. 93-106.

30. CIOCA, M., L. I. CIOCA, L. DUȚA, **Web Technologies and Multi-cr iterion Analysis used in Enterpr ise Integration**, Studies in Informatics and Control, vol. 20, issue 2, 2011, pp. 129-134.