A New Proxy Signature Scheme As Secure As ElGamal Signature

Song Han, Elizabeth Chang, Jie Wang, Wanquan Liu

Abstract— **Proxy signature helps the proxy signer to sign messages on behalf of the original signer. It is very useful when the original signer (e.g. the president of a company) is not available to sign a specific document. If the original signer can not forge valid proxy signatures through impersonating the proxy signer, it will be robust in a virtual environment; thus the original signer can not shift any illegal action initiated by herself to the proxy signer. In this paper, we propose a new proxy signature scheme. The new scheme can prevent the original signer from impersonating the proxy signer to sign messages. The proposed scheme is based on the regular ElGamal signature. In addition, the fair privacy of the proxy signer is maintained. That means, the privacy of the proxy signer is preserved; and the privacy can be revealed when it is necessary.**

Keywords: ElGamal signature, Proxy signature, Security, Hash function, Fair privacy.

I. Introduction

A digital signature is used to establish both of the signer authenticity and the data integrity assurance. One of the classic digital signatures is ElGamal signature [1]. Besides the property of the general digital signature [2], the proxy signature [3]-[6]-[9] could help the proxy signer to sign messages on behalf of the original signer. The concept of the proxy signatures was introduced by Mambo et al. [7]. The proxy signature is motivated by the scenario: A president of a company wants to ask his secretary to sign some important documents on behalf of him. However, the president does not hope the secretary know any secret information of the private key. Also, at a later time, any one who verifies the signature should be confirmed that the secretary signed the document on behalf of the president.

A number of proxy signature schemes have been proposed [3]-[6]-[9], [10], [11]. No one ever utilised the ElGamal signature to apply to the proxy signature design, though the (regular) ElGamal signature is efficient and powerful. A proxy signature scheme is a cryptographic primitive, that contains three entities: the original signer, the proxy signer and the verifier (at a later time). It allows the original signer to delegate her signing capability to a designated proxy signer. The proxy signer can sign some messages on behalf of the original signer. After receiving the proxy signature, the verifier, which knows the public keys of the original and proxy signers, verified the validity of the proxy signature. Generally, a proxy signature consists of four algorithms.

System setup: This establishes some public and private parameters for the original signer. In some situation, a pseudonym generation centre is involved in the setup algorithm.

Proxy generation: It outputs a pair of private and public keys for the proxy signer. The proxy generation usually involves a two-party protocol run between the original and proxy signers, and between the proxy signer and the pseudonym generation centre.

Signing: For an input that consists of a message to be signed and a proxy private key kept by the proxy signer, it outputs a valid proxy signature.

Verifying: For an input that includes a pair (a message and a signature) and the public keys of the original and proxy signers, it helps the verifier to verify the validity of the signature.

According to the original signer's control on the private key of the proxy signer, there exist two different kinds of proxy signature:

(1) ***Proxy unprotected proxy signature:*** Proxy unprotected proxy signature includes two cases: one is the full delegation, and the other is partial delegation. In the full delegation, a proxy signer is given the same private key as the original signer has, and computes the same signatures as the original signer does. Therefore, the original signer should take all the responsibility for messages signed by the proxy signer. In the partial delegation, the original signer uses her private key to create a proxy signature key and sends it to the proxy signer through a secure channel. The proxy signer uses the proxy signature key to compute proxy signatures on behalf of the original signer. For the security reason, it must be computationally infeasible to compute the original signer's private key from the proxy signer's proxy signature key.

(2) ***Proxy protected proxy signature:*** Proxy protected proxy signature includes one case: the delegation of the original signer by warrant. In the delegation by warrant, the original signer gives the proxy signer a warrant, which certifies that the proxy signer is legal. Delegation by warrant is performed through a series of executions of signing of the general digital signature scheme. The proxy signer generates the proxy signatures not only with the proxy signature key given by the original signer but also with the private key given by the

[0]Dr. Song Han is with the School of Information Systems, Curtin Business School, Curtin University of Technology. GPO Box U 1987, Perth WA 6845, Australia. Fax: +61 8 9266 3076. Email: song.han@cbs.curtin.edu.au.

[0]Prof. Elizabeth Chang is with the School of Information Systems, Curtin Business School, Curtin University of Technology. GPO Box U 1987, Perth WA 6845, Australia

[0]Prof. Jie Wang is with the National Science Foundation of China, Beijing 100085, China

[0]Dr. Wanquan Liu is with the Department of Computing, Curtin University of Technology, GPO Box U1987 Perth, WA 6845

pseudonym generation centre. Therefore, the original signer and the proxy signer share the responsibility for the underlying valid proxy signatures.

The proxy unprotected proxy signature can be applied to the scenario: If the president of a company is not available to sign an important document, he can entrust his secretary (using the proxy signature scheme) to sign the receipt of the document on behalf of him. This case is reasonable since the secretary is the reliable and important employee who belongs to the company.

The proxy protected proxy signature can be applied to the scenario: If the president of a company and his reliable secretary are both not available to sign an important document, he can entrust an employee (using the proxy signature scheme) to sign the receipt of the document on behalf of him. This case is reasonable since this employee may be not the very reliable employee who belongs to the company. When some illegal situation related to the underlying proxy signature happens, the privacy of the proxy signer can be revealed. Another scenario is that a bank approves its customers to sign bank's electronic checks, they need to use the proxy protected proxy signature. In this situation, the privacy of the customers should be preserved.

In this paper, we focus on the proxy protected proxy signatures. The proxy-protected signature scheme satisfies the following three basic security properties.

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

Strong unforgeability: Only a designated proxy signer can create a valid proxy signature for the original signer (even the original signer cannot do it).

Undeniability: Neither the origin signer nor the proxy signer must be able to sign in place of the other party. In other words, they cannot deny their signatures against any verifier.

An interesting issue is: how to preserve the privacy of the proxy signer? how can then this privacy be revealed when it is necessary. This paper will address this issue. In this paper, we propose a new proxy signature scheme. The new scheme is based on the regular ElGamal signature. Therefore, its signature generation and verifying are very efficient. Also, the security of the proposed scheme has the same security level (existential unforgeability) with the regular ElGamal signature (i.e. ElGamal signature with hash paddings). In addition, a very important property - fair privacy is preserved in the new scheme.

Therefore, among these three types of delegation, the delegation by warrant has the best security and privacy level. This can be confirmed by:

(1) The full delegation can not provide enough security and privacy, in that (a) the proxy signer knows the private key of the original signer; (b) it is probable that the original signer (or the proxy signer) signs a specific message and repudiate the signature at a later time to the verifier.

(2) The partial delegation can not provide enough security and fair privacy, in that the original signer has the full control on the proxy signing key of the proxy signer, although the proxy signer does not know the private key of the original signer compared with the full delegation.

(3) The delegation by warrant can provide enough security and fair privacy, in that (a) the proxy signer does not know the private key of the original signer; (b) the original signer has no full control on the private key of the proxy signer; (c) any verifier only gets the pseudonym of the proxy signer, but not the identity; (d) the privacy of the proxy signer is preserved; and also, the privacy can be revealed when it is necessary.

The organization of the rest of our paper is as follows: Section 2 introduces the discrete logarithm problem, discrete logarithm assumption, and the ElGamal signature. Section 3 presents a new proxy signature scheme. This proxy signature scheme is based on ElGamal signature. Section 4 provides security analysis and privacy analysis. The performance analysis is presented in section 5. The concluding remarks are provided in section 6.

II. Preliminary

In this section, we provide some preliminaries for the proposed proxy signature scheme. What we present include: discrete logarithm, discrete logarithm problem, and discrete logarithm assumption. The security of the new proxy signature scheme is based on the ElGamal signatures.

A. Discrete Logarithm Problem

(Discrete Logarithm) Discrete Logarithm: Let G be a finite cyclic group of order n. g is a generator of G ($G = \langle g \rangle$), and let $y \in G$. The discrete logarithm of y to the base g, denoted $\log_g y$, is the unique integer x and $x \in Z_n$ such that $y = g^x$

Discrete Logarithm Problem (1) DLP: Given a large prime p, a generator g of Z_p^*, and an element $y \in Z_p^*$. The discrete logarithm problem is to find the integer $x \in Z_p^*$ such that $g^x = y \bmod p$. (2) GDLP: Given a finite cyclic group G of order n, a generator $g \in G$, and an element $y \in G$, the GDLP is to find the integer x such that $g^x = y$. Algorithms to compute discrete logarithm include: (1) Arbitrary group: Exhaustive search, baby-step giant-step, Pollards rho; (2) Group with small prime factors: Pohlig-Hellman; (3) Index calculus efficient only in certain groups.

Discrete Logarithm Assumption Given a large prime p carefully chosen, such that the above algorithms do not work on the DLP. Let g be a generator of Z_p^*, and an element $y \in Z_p^*$. Then, it is computationally difficult to find an integer x such that $y = g^x \bmod p$.

B. ElGamal Signature

ElGamal Signature Scheme: The ElGamal signature scheme is as follows: Let p be a prime and let α be a primitive element in Z_p^*. The set of all messages is $P = Z_{p-1}$, the set of all signatures $C = Z_{p-1} \times Z_{p-1}$. And the set of all keys $K = \{(p, \alpha, a, \beta) : \beta = \alpha^a \bmod p\}$. p, α, and β make up the public key; and the private key is a. Alice is the signer, and Bob is the verifier.

Signing Given a message m,

(1) Alice chooses a number $k \in Z_{p-1}$ randomly, and

computes $\gamma = \alpha^k \text{mod } p$.

(2) Alice computes $\delta = (m - a\gamma)(k^{-1}) \text{mod } (p-1)$. Then, $\{\gamma, \delta\}$ is the signature on the message m.

Verifying Given a signature $\{\gamma, \delta\}$ on a message m, Bob will do as follows:

(1) Bob computes $z = \gamma^\delta \beta^\gamma$.

(2) Bob computes $e = \alpha^m \text{mod } p$ and checks whether $e = z \text{mod } p$. If it holds, Bob accepts it as a valid signature. Otherwise, it is an invalid signature.

Security If the attacker can compute the value $a = \log_\alpha \beta$, then ElGamal signatures can be forged. As long as p is chosen carefully and α is a primitive element modulo p, then solving the Discrete Logarithm problem in Z_p^* is infeasible. Additionally, k must be secret, only used once and random. As shown above, ElGamal signature can also be used for encryption as well, but the messages should be relatively small in size.

It should be noticed that the plain ElGamal signature scheme is not secure against existential forgery attack. Therefore, we need to shift it to a much more secure version when our new proxy signature security is based on the ElGamal signatures. We have exploited the hash padding technique in the following proxy signature scheme. We call this new version of ElGamal signatures the regular ElGamal signatures.

III. New Proxy Signature as Secure as ElGamal Signature

A. System Setup

There are three entities involving in the proxy signature scheme: an original signer S, a pseudonym generation centre (PGC), and a proxy signer P. The original signer will delegate his signing rights to the proxy signer. To do this, the original signer needs the help of the pseudonym generation centre. The proxy signer will first register herself at the pseudonym generation centre. And then, the PGC will generate a pseudonym, which is privately linked to the identity of the proxy signer. This mechanism will help an entity (who is a verifier) to ask the pseudonym generation centre to identify the proxy signer, only if the proxy signer abuses his proxy signing right. The system parameters are as follows:

$$\{p, q, g, L, H(\cdot,\cdot), id_S, id_p, id_{pgc}, x_s, x_{pgc}, y_1, y_2\}$$

Here, p and q are two large primes. L is an integer and $512 \leq L \leq 1025$. The bit length of p is L. g is an q-order element and $g \in Z_p^*$. id_s and id_{pgc} are the identity of the original signer and the proxy signer, respectively. $x_s \in Z_q^*$ and $x_{pgc} \in Z_q^*$ are the private key of the original signer and the pseudonym generation centre, respectively. $y_1 \in Z_p$ and $y_2 \in Z_p$ are the public key of the original signer and the pseudonym generation centre, respectively. $H(\cdot,\cdot)$ is a cryptographic hash function, $H : Z_p \times \{0,1\}^* \mapsto Z_q$.

B. Proxy Generation

Proxy generation includes three stages: the first stage is the issuing pseudonym stage; the second stage is the delegation stage; and the third stage is the proxy signing key generation stage. The details are as follows:

1) Issuing pseudonym stage: In this stage, the pseudonym generation centre (PGC) will interact with the proxy signer. PGC issues a pseudonym n_p, a public parameter r_1 and a partial secret key s_1 to the proxy signer P. The PGC will record the tuple $\{n_p, r_p, id_p\}$ in its own database, where r_p is a random element chosen by the PGC. The following is the detail:

(1) The proxy signer P registers her identity id_p to the pseudonym generation centre PGC.

(2) The pseudonym generation centre PGC chooses a random element $r_p \in Z_q^*$ and computes n_p

$$n_p = H(id_p, r_p). \tag{1}$$

(3) The pseudonym generation centre PGC chooses another random element $k_p \in Z_q^*$, computes r_1 and s_1,

$$r_1 = g^{k_p} \text{mod } p \tag{2}$$

and

$$s_1 = (k_p + x_{pgc} H(n_p, r_1)) \text{mod } q, \tag{3}$$

and records the tuple $\{n_p, r_p, id_p\}$ in its own database. This will help the pseudonym generation centre identify the identity of the proxy signer in future, when the proxy signer does something illegal. PGC then sends $\{n_p, r_1, s_1\}$ to P.

(4) After receiving $\{n_p, r_1, s_1\}$, the proxy signer P checks whether

$$y_2^{H(n_p, r_1)} = g^{s_1} (\text{mod } p). \tag{4}$$

If it holds, P will use s_1 to compute her proxy signing key in the third stage (Proxy signing key generation stage) and n_p as the pseudonym which is linked to her identity id_p.

2) Delegation stage: In this stage, the original signer S interacts with the proxy signer P. S will delegate P to sign some messages on behalf of S. To fulfil it, S sends a partial secret s_2, a warrant $m_w \in \{0,1\}^*$ (m_w is required not to include the identity of the proxy signer), and a public parameter t_S. After receiving the elements, P will verify the correctness of the tuple. Only if it is correct, P accepts the delegation. The details are the followings:

(1) The original signer S chooses a random number $k_s \in Z_q^*$, computes $t_S = g^{k_s}$mod p and

$$s_2 = (k_s + x_s H(t_S, m_w)) \text{mod } q. \tag{5}$$

And then, S sends the tuple $\{s_2, m_w, t_S\}$ to P.

(2) After receiving the tuple $\{s_2, m_w, t_S\}$, P will verify whether

$$y_1^{H(t_S, m_w)} = g^{s_2} (\text{mod } p). \quad (6)$$

If this equality holds, P will accept the delegation from S. Otherwise, P will not accept.

3) Proxy signing key generation stage: In this stage, the proxy signer P will construct the proxy signing private key and public key pair. P utilizes the partial secret keys s_1 and s_2 from S and PGC, respectively. The detail is as follows:

(1) P computes s as the proxy signing private key:

$$s = (s_2 H(s_1, id_S) + s_1)(\text{ mod } q) \quad (7)$$

(2) P computes $y = g^s \text{mod } p$ as the proxy signing public key, which will be used in the verifying signatures generated by the proxy signer on behalf of the original signer. Notice that the public key y should be certified by the pseudonym generation centre, in order to maintain the authentication, integration and non-repudiation. We do not provide the details of the certificate information, since it can be easily deployed using the certificate technique in [10].

C. Signature Generation

In this procedure, the proxy signer P will sign messages on behalf of the original signer. Given a message $m \in \{0, 1\}^*$,

(1) P chooses $k \in Z_p^*$ randomly and uniformly, and computes a,

$$a = g^k (\text{mod } p). \quad (8)$$

(2) P computes $z = k^{-1} (\text{mod } q)$ and b

$$b = z(H(m, m_w \| id_S \| n_p) - as)(\text{mod } q). \quad (9)$$

The proxy signer will publish $\{a, b\}$ as the proxy signature on the message m. Also, any verifier can get the public parameters $\{m_w, n_p\}$ and the identity id_S of the original signer. $\{m_w, n_p, id_S\}$ are associated with the proxy signature $\{a, b\}$.

D. Signature Verifying

The verifier will work on this procedure. Given an alleged proxy signature $\{a, b\}$ on a message m, the verifier utilizes the public key parameters $\{m_w, n_p\}$ and the identity id_S of the original signer to follow the steps:

(1) The verifier verifies whether $a \in Z_p$.

(2) The verifier computes $v_1 = y^a a^b (\text{mod } p)$.

(3) The verifier computes $v_2 = g^{H(m, m_w \| id_S \| n_p)} \text{mod } p$.

(4) The verifier accepts the signature as valid if and only if $v_1 = v_2$.

E. Identifying Proxy Signer

If the verifier holds a valid proxy signature $\{a, m_w, n_p, b, id_S\}$ on a message m, this procedure will help the verifier to ask the proxy generation centre to identify the proxy signer if the proxy signer does anything illegal through using the proxy signing algorithm. To do this, the verifier first sends the pseudonym n_p to the proxy generation centre PGC. After receiving n_p, PGC checks $n_p = H(r_p, id_p)$ by using the tuple $\{n_p, r_p, id_p\}$ in its own database, and then PGC can identify the proxy signer.

IV. Security and Privacy Analysis

This paper has presented a new proxy signature scheme. The security of the proposed proxy signature scheme is based on the regular ElGamal signatures. Besides the strong unforgeability, the new proxy signature scheme has the fair identifiability of the proxy signer, the verifiability of proxy signatures generated by the proxy signer on behalf of the original signer, and the undeniability of the proxy signer. Another important point on the proxy signature scheme is that the privacy of the proxy signer is preserved. The details of the related proofs are presented as follows:

A. Security Analysis

We will prove that the proposed proxy signature scheme satisfies the following security properties: (1) strong unforgeability; (2) *fair identifiability*; (3) verifiability; and (4) strong undeniability.

(1) Strong unforgeability. Strong unforgeability means that the original signer even can not forge valid proxy signatures. From Equation (2) and (6), we have

$$\begin{aligned} s &= (s_2 H(s_1, id_S) + s_1) \text{mod } q \\ &= (k_p + s_2 H(s_1, id_S) + x_{pgc} H(n_p, r_1)) \text{mod } q. \end{aligned} \quad (10)$$

It is difficult to derive the value of s, since k_p and x_{pgc} are two random and private elements of Z_q^*. If the original signer tries to tackle Equation (1), she will need to solve the discrete logarithm problem. However, the discrete logarithm problem is difficult. On the other hand, from Equation (8) and (9), the proposed proxy signature is based on the ElGamal signature. Therefore, the strong unforgeability is maintained, since the hash padding technique is utilized in the proxy signature generation.

(2) Fair Identifiability. Fair identifiability means that no one (excluding the pseudonym generation centre and the original signer) is able to identify the underlying proxy signer only from a valid proxy signature. An exceptional situation is the pseudonym generation centre can identify the underlying proxy signer through the pseudonym. This only takes place when the proxy signer denies a valid proxy signature (the verifier has verified that the underlying proxy signature is valid). In fact, from the proxy signature generated by the proxy signer

$$\{a, m_w, n_p, b, id_S\} \quad (11)$$

on the message m, any one (excluding the pseudonym generation centre) cannot identify the proxy signer. This is because: (a) We have limited the warrant m_w not to include the identity information of the proxy signer. It does not leak any information of the identity for the proxy signer. And (b) $n_p = H(id_p, r_p)$. Therefore, the identity of the proxy signer has been encapsulated using a cryptographic hash function.

(3) Verifiability. Verifiability means that any verifier who holds the related public parameters can check whether a proxy signature is valid and can be confirmed that the signature is generated according to the delegation of the original signer, if the signature is valid. From the Signature Verifying algorithm in section 3, we know that the verifier utilizes the public key parameters $\{m_w, n_p\}$ and the identity id_S of the original signer to verify an alleged proxy signature $\{a, b\}$ on a message m. Only if $\{m_w, n_p\}$ and id_S are involved in the verifying algorithm, and also the signature is really computed by the proxy signer according to the delegation, the verifier would be confirmed that the original signer has delegated the proxy signer to sign the document on behalf of the original signer.

(4) Undeniability. Undeniability means that the proxy signer cannot deny she ever generated a valid signature on a given message on behalf of the original signer. In fact, from Equation (1) we know that the pseudonym n_p is information-theoretically linked to the proxy signer (i.e. her identity). More importantly, the verifying equation $v_1 = y^a a^b (\text{mod } p)$ implies the proxy signer ever generated the signature on the message.

B. *Privacy Analysis*

We will prove that the privacy of the proxy signer is preserved. In fact, it is known that privacy is paramount particularly in respect to financial situations of the participant. Therefore, it is necessary to preserve the privacy. There are three aspects that will be discussed on the privacy: privacy of the proxy signing private key, privacy of the identity of the proxy signer, and fair privacy of the proxy signer.

1) Privacy of the proxy signing private key: This privacy is maintained, since the proxy signer constructs the proxy signing private key $s = (s_2 H(s_1, id_S) + s_1)(\text{ mod } q)$ using the partial secret elements s_1 (from the pseudonym generation centre) and s_2 (from the original signer), respectively. Therefore, the original signer can not figure out the proxy signing private key.

2) Privacy of the identity of the proxy signer: This privacy is maintained, since the identity id_p of the proxy signer is hashed through a secure cryptographic hash function. Therefore, an attack (excluding S and PGC) can not derive the proxy singer from the pseudonym n_p (a public parameter) and a valid signature $\{a, b\}$.

3) Fair Privacy of the proxy signer: Fair privacy means that the privacy of the proxy signer can be revealed when it is necessary. Given a valid signature $\{a, b, n_p, m_w, id_S\}$, the pseudonym generation centre can identify the proxy signer using the equation $H(id_p, r_p) = n_p$.

V. Performance Analysis

We discuss the computation performance by the following three procedure: the proxy generation algorithm, the signing algorithm, and the verifying algorithm.

Proxy generation involves four hash function evaluations, seven exponentiations modulo p, and three multiplication modulo p. However, all these computations can be done in a pre-computed way.

Signing algorithm involves one exponentiation modulo p, one inversion computation modulo q, two multiplications modulo q, and one hash function evaluation. Therefore, the signing algorithm is efficient.

Verifying algorithm involves three exponentiations modulo p, one multiplication modulo p, and one hash function evaluation.

VI. Conclusion

This paper has presented a new proxy signature scheme. The proposed scheme is derived from the ElGamal signature. Therefore, the security (unforgeability) of the proposed proxy signatures is based on the regular ElGamal signature. Also, the new scheme has the fair identifiability, the verifiability, and the undeniability. Since the proxy signer signs some messages on behalf of the original signer, the fair privacy of the proxy signer is preserved.

Acknowledgment

The authors would like to thank the anonymous reviewers.

This work is supported by the ARC Funding and Research Fellowship within the Centre CEEBI and School of Information Systems, Curtin Business School, Curtin University of Technology.

References

[1] T. Elgamal, ”A public key cryptosystem and a signature scheme based on discrete logarithms”, IEEE Trans. IT-31 (1985) 469-472.

[2] E. J. Goh, S. Jarecki, ”A signature scheme as secure as the Diffie-Hellman problem”, in: Proceedings of Eurocyrpt’2003, LNCS 2656, 2003, pp. 401-415.

[3] S. J. Hwang, C.-C. Chen, ”A new proxy multi-signature scheme”, in: International Workshop on Cryptology and Network Security, Taipei, Taiwan, ROC, December 2000, pp. 134-138.

[4] C. L. Hsu, T.-S. Wu, T.-C. Wu, ”New nonrepudiable threshold proxy signature scheme with known signers”, The Journal of System and Software 58 (2001) 119-124.

[5] S.J. Hwang, C.-H. Shi, ”A simple multi-proxy signature scheme”, in: Proceedings of the Tenth National Conference on Information Security, Hualien, Taiwan, ROC, 2000, pp. 134-138.

[6] B. Lee, H. Kim, K. Kim, ”Strong proxy signature and its application”, in: Proceedings of SCIS 2001, 2001, pp. 603-608.

[7] M. M. Mambo, K. Usuda, E. Okmamoto, ”Proxy signatures: delegation of the power to sign message”, IEICE Transaction Functional E79-A (9) (1996) 1338-1354.

[8] D. R. Stinson, ”Cryptography: theory and practice”’ CRC Press, 1995.

[9] M. S. Hwang, L.-C. Lin, J.-L.L.U. Eric, ”A secure nonrepudiable threshold proxy signature scheme with known signers”, Information 11 (2) (2000) 137-144.

[10] S. Kim, S. Park, and D. Won, ”Proxy Signatures, Revisited”, Proc. of ICICS’97, Y. Han et al(Eds.), LNCS 1334, Springer-Verlag, pp. 223-232, 1997.

[11] H. X. Wang, J. Pieprzyk, ”Efficient one-time proxy signatures”, in: Proceedings of Asiacrypt 2003, LNCS 2894, 2003, pp. 507-522.