An Efficient Proxy Signature Scheme Without Bilinear Pairings

Hassan Elkamchouchi[1], Eman Abou El-kheir[2], and Yasmine Abouelseoud[3]
Alexandria University[1,3], Kafr El-Sheikh University[2], Egypt

Abstract— A signature scheme is a method for signing a message stored in electronic form. As such, a signed message can be transmitted over a computer network in an authenticated manner. This paper introduces a digital signature scheme and a proxy digital signature scheme without bilinear pairings. Both schemes are based on the elliptic curve discrete logarithm problem (ECDLP). Both schemes achieve the standard security requirements. Moreover, the two schemes are compared with other schemes and it is shown that the proposed schemes are more efficient and require less computational effort.

Index Terms— Digital Signature, Proxy Signature, ECDLP, Without Bilinear Pairings, Authentication

1 Introduction

Digital signatures offer source authentication in cryptography. To handle the situations arising in the digital world related to authentication, different types of digital signatures have been developed [1]. The concept of a proxy signature was first introduced by Mambo et al. [2] in 1996. In a proxy signature scheme, generally, there are two entities: an original signer and a proxy signer. The original signer can delegate his signing power to a proxy signer. The proxy signer can generate a valid signature on behalf of the original signer. Since then, many proxy signature schemes have been proposed [3, 4, 5, 6].

This paper proposes two digital signature schemes; the first one is a digital signature that satisfies unforgeability and verifiability properties, and the second is a proxy signature scheme in which the original signer delegates his signing rights to a proxy. The receiver verifies the identities of both the original signer and the proxy signer as discussed in details in the rest of paper.

The rest of paper organized as follows. In Section 2, the security requirements for a digital signature are presented. Then the proposed digital signature scheme is presented in Section3. Section 4 discusses the security analysis of the proposed digital signature scheme followed by the performance analysis and a comparative study in Section 5. Section 6 introduces the proxy signature algorithms followed by the security requirements for any proxy Signcryption scheme in Section 7. The proposed proxy signature scheme is introduced in Section 8 followed by its security analysis and a comparative study in Section 9. Finally, the conclusion is in Section 10.

- *Hassan Elkamchouchi : Elec. Eng. Dept, Fac. of Eng., Alexandria University.E-mail: helkamchouchi@ieee.org*
- *Eman Abou El-kheir: Elec. Eng. Dept, Fac. of Eng., Kafr El-Sheikh University. E-mail: eman.abouelkhair@eng.kfs.edu.eg*
- *Yasmine Abouelseoud: Eng. Math. Dept, Fac. of Eng., Alexandria University. E-mail: yasmine.abouelseoud@gmail.com*

2 The Security Requirements for a Digital Signature Scheme

The security requirements for any digital signature scheme are summarized below [4,2]:

2.1 Unforgeability

Only the original signer can produce a valid signature.

2.2 Verifiability

A verifier can be convinced that the original signer agrees on signing the message by testing the verification condition.

3 The Proposed Signature Scheme

3.1 Setup

Given the security parameter k (usually 160), the CA (certifying authority) chooses q a large prime number with $q > 2^k$, (a, b) is a pair of integers which are smaller than q and satisfy $(4a^3 + 27b^2) \bmod q \neq 0$. E is the selected elliptic curve over the finite field $F_q : y^2 = (x^3 + ax + b) \bmod q$. P is the base point or generator of a group of points on E, denoted as G. Also, O is the point at infinity and n is the order of the point P, with n being a prime number, $n.P = O$ and $n > 2^k$. The CA selects a cryptographic one way hash function $H : \{0,1\}^* \rightarrow Z_q$. The CA publishes the system parameters: $\{k, a, b, E, P, H\}$

3.2 Key generation

A user chooses his secret key $d_a \in [q-1]$ and computes $d_a.G = Q_a$, Q_a is the user's public key.

3.3 Signature generation

A signer chooses a random number $w \in [q-1]$ and computes :

- $r = [(w + d_a) \bmod q].G = (u, v)$

- $s = (u + h(m).d_a) \bmod q$
- The signer sends (u, s, m) to the verifier.

3.4 Signature verification

The receiver computes:

- $v_1 = h(m).Q_a$
- $v_2 = [(s - u) \bmod q)].G$
- If $v_1 = v_2$ accept the signature

4 Security analysis of the proposed signature

4.1 Correctness

The correctness of the verification equation as follow:
$v_2 = [(s - u) \bmod q)].G = (u + h(m).d_a - u).G = h(m).Q_a = v_1$

4.2 Security Properties

4.2.1 Unforgeability

Only the original signer with his/her secrete key d_a can produce both (r, s) because of the way they are computed:

$r = [(w + d_a) \bmod q].G = (u, v)$,

$s = (u + h(m).d_a) \bmod q$

They depend on the sender secret key d_a. Therefore, only the original signer can generate a valid signature.

4.2.2 Verifiability

A verifier can be convinced of the agreement of the signer to the message contents by computing

$v_1 = h(m).Q_a$, $v_2 = [(s - u) \bmod q)]$. Then, testing if $v_1 = v_2$, a verifier then accepts the signature .

4.3 Performance Analysis and Comparative Study

Table 1 shows the symbol definitions that are used in the comparative study.

TABLE 1
Time Abbreviations

Symbol	Operation
$T_{EC\text{-}mult}$	time complexity required for executing multiplication operation on elliptic curve E
$T_{EC\text{-}add}$	time complexity required for executing addition operation on elliptic curve E
T_{mult}	time complexity required for executing modulus multiplication over a finite field
$T_{inverse}$	time complexity required for executing inverse modulus over a finite field
$T_{pairings}$	time of executing a bilinear pairing operation

The proposed Signature scheme is compared with the elliptic curve digital signature algorithm (ECDSA) and EC ElGamal Signature Scheme in Zhaohui Cheng[1] . The comparison is shown in table 2.

TABLE 2
The proposed signature scheme compared with the schemes in [1]

Phase	Zhaohui Cheng[1] (ECDSA)	Zhaohui Cheng[1] ElGamal	The proposed
System Construction	$1T_{EC\text{-}mult}$	$1T_{EC\text{-}mult}$	$1T_{EC\text{-}mult}$
Signature Generation	$1T_{EC\text{-}mult} + 1T_h + 2T_{mult} + 1\ T_{inverse}$	$1T_{EC\text{-}mult} + 2T_{mult} + 1T_h + 1\ T_{inverse}$	$1T_{EC\text{-}mult} + 1T_{mult} + 1T_h$
Signature Verification	$2T_{EC\text{-}mult} + 1T_h + 2T_{mult} + 1\ T_{inverse} + 1T_{EC\text{-}add}$	$3T_{EC\text{-}mult} + 1T_h + + 1T_{EC\text{-}add}$	$2T_{EC\text{-}mult} + 1T_h$
Total	$4T_{EC\text{-}mult} + 2T_h + 4T_{mult} + 2T_{inverse} + 1T_{EC\text{-}add}$	$5T_{EC\text{-}mult} + 2T_h + 2T_{mult} + 1T_{inverse} + 1T_{EC\text{-}add}$	$4T_{EC\text{-}mult} + 1T_{mult} + 2T_h$

From the comparative study, it is clear that the proposed scheme requires less computational effort compared to the schemes in [1].

5 Proxy Signature Scheme algorithms

Any proxy signature scheme is specified by the following four algorithms[7]:

5.1 Setup

This algorithm takes as input a security parameter n and outputs the system public parameters PP. And the original signer S_a selects its key pair (pk_a, sk_a) and the proxy signer S_p selects its key pair (pk_p, sk_p) , respectively.

5.2 Proxy key generation

This algorithm takes as input the private keys sk_a and sk_p. It outputs a secret key proxy skp for the proxy signer S_p.

5.3 Proxy signature generation

This algorithm takes as input the proxy secret key skp and a message m. It outputs a proxy signature μ .

5.4 Proxy signature verification

This algorithm takes as input a proxy signature μ on the massage m and outputs 1 if the signature is valid. Otherwise, it outputs 0.

6. The security requirements for a proxy

SIGNATURE SCHEME

A secure proxy signature scheme should fulfill the following properties [4, 2, 8]:

6.1 Distinguishability

The proxy signature must be distinguishable from the original signer's signature.

6.2 Verifiability

From proxy signatures, a verifier can be convinced of the original signer's agreement on the signed messages.

6.3 Unforgeability

Only the proxy signer can produce a valid proxy signature on behalf of the original signer.

6.4 Identifiability

Anyone can determine the identity of the corresponding proxy signer from a proxy signature.

6.5 Nonrepudiation

Once the proxy signer creates a valid proxy signature on behalf of the original signer, he cannot repudiate his signature creation against anyone else.

6.6 Prevention of misuse

The proxy signer cannot use the proxy key for other purposes than generating a valid proxy signature.

7 THE PROPOSED PROXY SIGNATURE SCHEME

Setup

The set up phase is similar as the signature scheme.

7.1 Key generation

- Signer chooses his secret key $d_a \in [q-1]$ and computes $d_a.G = Q_a$, where (d_a, Q_a) are the private and public key pair of the original signer.
- Proxy chooses his secret key $d_p \in [q-1]$ and computes $d_p.G = Q_p$, where (d_p, Q_p) are the private and public key pair of the proxy signer.

7.2 Proxy delegation

The original signer chooses a random number d and computes

- $T = d.G = (\alpha, \beta)$
- $\sigma = (d - d_a.h(\alpha, m_w)) mod\ q$
- The original signer sends (α, σ, m_w) to the proxy signer, where m_w is a warrant specifying the identities of both the original signer and the proxy signer as well as the signing rights of the proxy agent and possibly a time frame for the validity of the warrant..

7.3 Proxy key generation

The proxy checks if $T \stackrel{?}{=} \sigma.G + h(\alpha, m_w).Q_a$. If the equation holds, the proxy signer computes the secrete proxy key $skp = (d_p + \sigma) mod\ q$. Then, the proxy signer generates the signature.

7.4 Proxy Signature generation

The proxy signer chooses a random number $w \in [q-1]$ and computes :

- $r = [(w + skp) mod\ q].G = (u, v)$
- $s = (u + h(m).skp) mod\ q$
- signer sends $(\alpha, \sigma, m_w, u, s, h(m))$ to the verifier

7.5 Proxy Signature verification

The receiver computes :

- $v_1 = h(m).[T - h(\alpha, m_w).Q_a + Q_p]$
- $v_2 = [(s - u) mod\ q)].G$

If $v_1 = v_2$, accept the signature .The receiver verifies the identities of both the original signer as well as the proxy signature using the warrant.

8 SECURITY ANALYSIS AND COMPARATIVE STUDY

8.1 Correctness

The proxy agent checks the equation : $T = \sigma.G + h(\alpha, m_w).Q_a$

$= (d - d_a.h(\alpha, m_w)).G + h(\alpha, m_w).Q_a$

$= d.G - d_a.h(\alpha, m_w).G + h(\alpha, m_w).Q_a = d.G = T$

The receiver computes: $v_1 = h(m).[T - h(\alpha, m_w).Q_a + Q_p]$

$v_1 = h(m).[d.G - h(\alpha, m_w).d_a.G + Q_p]$

$v_1 = h(m).[d.G - h(\alpha, m_w).d_a.G + Q_p]$

$v_1 = h(m).(\sigma.G + Q_p)$

The receiver computes: $v_2 = [(s - u) mod\ q)].G$

$v_2 = (u + h(m).skp - u).G$

$v_2 = h(m).skp.G = h(m).(d_p + \sigma).G$

$v_2 = h(m).(d_p.G + \sigma.G) = h(m).(\sigma.G + Q_p) = v_1$, then the receiver accepts the signature if the equality holds.

8.2 Security properties

8.2.1 Distinguishability

The proposed proxy signature $(\alpha, \sigma, m_w, u, s, h(m))$ contains the warrant m_w while the normal signature does not, so both are different in the form. Also in the verification equation, public keys Q_a and Q_p, also and warrant m_w are used. So anyone can distinguish the proxy signature from a normal signature easily.

8.2.2 Verifiability

The verifier of a proxy signature can check easily that the verification equation $v_1 = h(m).[T - h(\alpha, m_w).Q_a + Q_p] = v_2$,where $v_2 = [(s - u) mod\ q)].G$, if $v_1 = v_2$ accept the signature holds. In addition, this equation involves original signer's public key Q_a and warrant m_w, so any one can be convinced of the original signer's agreement on the proxy signer.

8.2.3 Unforgeability

In our scheme only the designated proxy signer can create a valid proxy signature, since the proxy private key

$skp = (d_p + \sigma) mod\ q$ includes the private key d_p of the proxy signer and to compute d_p from Q_p is equivalent to solving the ECDLP.

8.2.4 Nonrepudiation

This is because of the presence of the warrant m_w and public keys Q_a and Q_p in the verification equation. Also, the generation of a proxy signature involves both the original and proxy signers' private keys d_a and d_p respectively. It is already proved that neither the original signer nor the proxy signer can sign in place of any other party. So the original signer cannot deny his delegation and the proxy signer cannot deny having signed the message *m* on behalf of original signer to another party.

8.2.5 Identifiability

In the proposed scheme, it can be checked who is original signer and who is proxy signer from the warrant m_w. Also, it clear from the verification equation

$$v_1 = h(m).[T - h(\alpha, m_w).Q_a + Q_p] = v_2$$

where $v_2 = [(s - u) mod\ q)].G$

that the public keys Q_a and Q_p are asymmetrical in position. So anyone can distinguish the identity of the proxy signer from the proxy signature.

8.2.6 Prevention of Misuse

The original signer generates the delegation (α, σ, m_w) where $T = d.G = (\alpha, \beta)$ and $\sigma = (d - d_a.h(\alpha, m_w)) mod\ q$ using its private key and sends it to the proxy. So the delegation cannot be modified or forged. Also the warrant m_w contains the limit of delegated signing capability. So it is not possible to sign the messages that have not been authorized by original signer

8.3 Comparative study

The proposed proxy signature scheme is compared with the schemes in [9,10]. Table 3 shows the comparison in details.

From the comparison, it can be seen that the proposed proxy signature scheme requires less computational effort than the scheme with pairings [9] and also the scheme without pairings[10].

TABLE 3
THE PROPOSED PROXY SIGNATURE SCHEME COMPARED WITH THE SCHEMES IN[9,10]

Phase	The scheme in [9] with pairings	The scheme in [10] without pairings	The proposed scheme without pairings
Proxy delegation	$1T_{EC\text{-}mult}$ + $1T_{inverse}$+ $1T_h$	$1T_{EC\text{-}mult}$+ $1T_h$+ $1T_{mul}$	$1T_{EC\text{-}mult}$+ $1T_h$+ $1T_{mul}$
Proxy key generation	$2T_{EC\text{-}mult}$ + $1T_{EC\text{-}mult}$+ $2T_{pairings}$+ $1T_{inverse}$	$2T_{EC\text{-}mult}$ + $1T_{EC\text{-}add}$ +$2T_h$ +$1T_{mul}$	$2T_{EC\text{-}mult}$ + $1T_{EC\text{-}add}$ +$1T_h$
Proxy Signature generation	$2T_{EC\text{-}mult}$ +$1T_h$+ $1T_{inverse}$	$1T_{EC\text{-}mult}$ +$1T_h$+ $1T_{mult}$	$1T_{EC\text{-}mult}$ +$1T_h$+ $1T_{mult}$
Proxy Signature verification	$1T_{EC\text{-}mult}$ + $2T_{EC\text{-}mult}$+ $2T_h$ + $1T_{mult}$ + $2T_{pairings}$	$4T_{EC\text{-}mult}$ + $3T_{EC\text{-}add}$ +$3T_h$+ $3T_{mult}$	$3T_{EC\text{-}mult}$ + $2T_{EC\text{-}add}$ +$2T_h$
Total	$6T_{EC\text{-}mult}$ +$3T_{EC\text{-}}$	$8T_{EC\text{-}mult}$ +$4T_{EC\text{-}}$	$7T_{EC\text{-}mult}$ +$3T_{EC\text{-}}$

From the comparison, it can be seen that the proposed proxy signature scheme requires less computational effort than the scheme with pairings [9] and also the scheme without pairings[10].

9 Conclusion

This paper proposes two schemes; the first is a digital signature with its security analysis discussion, and the second is a proxy signature with its security analysis discussion. Both schemes are more efficient than other schemes when compared with them. Both schemes are without bilinear pairing.

References

[1] Z. Cheng, "Simple Tutorial on Elliptic Curve Cryptography", Chapter 2. ECC In Practice , December 1, 2004

[2] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: delegation of the power to sign messages," Transactions on Fundamentals of Electronic Communications and Computer Science, vol. E79-A, pp. 1338-1354, 1996.

[3] S. Kim, S. Park, and D. Won, "Proxy signatures," Proceedings of international conference on information and communications security (ICICS)'97, LNCS 1334, pp. 223-232, Springer-Verlag, 1997.

[4] B. Lee, H. Kim, and K. Kim, "Strong proxy signature and its applications," SCIS2001, vol. 2, no. 2, pp. 603-608, 2001.

[5] J. Lee, J. Cheon, and S. Kim, "An analysis of proxy signatures: Is a secure channel necessary," Cryptology-CT-RSA'03, LNCS 2612, pp. 68-79, Springer-Verlag, 2003.

[6] S. F. Tzeng, M. S. Hwang, and C. Y. Yang, "An improvement of nonrepudiable threshold proxy sig- nature scheme with known signers," Computers & Security, vol. 23, pp. 174-178, 2004.

[7] M. Tian and L. Huang,"Breaking A Proxy Signature Scheme From Lattices. International Journal of Network Security, Vol.14, No.6, PP.320-323, Nov. 2012

[8] Y. Kim and J. H. Chang, " Self Proxy Signature Scheme ", IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.2, February 2007

[9] F. Zhang, R. Safavi-Naini and W. Susilo," An Efficient Signature Scheme from Bilinear Pairings and Its Applications", Springer-Verlag ,pp 277-290 PKC 2004, LNCS 2947

[10] S. Padhye, N. Tiwari," Improved Proxy Signature Scheme without Bilinear Pairings", In the Proceeding of 9th International Conference, QShine 2013, Greader Noida, India, January 11-12, 2013.