(IJC)

ISSN 2307-4523 (Print & Online)

© *Global Society of Scientific Research and Researchers*

http://ijcjournal.org/

Proxy Signcrypion Scheme Based on Hyper Elliptic Curves

Insaf Ullah [a]*, Inam Ul Haq[b], Noor Ul Amin [c], Arif Iqbal Umar [d], Hizbullah khattak[e]

[a,c,d,e]*Department of Information technology, Hazara University, Mansehra, Khyber Pakhtunkhwa, Pakistan.*

[b]*Department of Computer Science and Bio Informatics, Khushal, Khan Khattak University, Karak, Postcode 27200,Khyber Pakhtunkhwa, Pakistan.*

[a] *Email: insafktk@gmail.com*

[b]*Email:inamix@gmail.com*

[c] *Email: namin@hu.edu.pk*

[d]*Email: arifiqbalumar@yahoo.com*

[e]*Email: hizbullahkhattak@yahoo.com*

Abstract

Delegation of rights is promising in Internet applications like distributed computing, e-cash systems, global distribution networks, grid computing, mobile agent applications, and mobile communications. This paper presents a novel Proxy Signcrypion Scheme based on hyper elliptic curves, attractive for resource constrained environment due to its shorter key size. It has properties of warrant integrity, message integrity, message confidentiality, warrant unforgeability, message unforgeability, proxy non repudiation and public verifiability. The proposed scheme has reduced computational cost as compared to the other existing schemes.

Keywords: Proxy Communication; Proxy Signature; Proxy Signcrypion; Hyper Elliptic Curve.

----

* Corresponding author.

1. Introduction

Proxy communications play vital role in emerging internet environment and business processes, rights delegations for business organizations and enterprises. It helps in applications such as digital contract signing and online proxy auction. It has importance for resource constrained devices to transform heavy computational work from a low power device to a more powerful server. The term proxy signature was coined by Membo and his colleagues [1]. It enables the original user (Alice) to transfer his signing capability to another entity called proxy thus proxy signs messages on the behalf of the original user. Authentication is provided by digital signature and confidentiality is guaranteed by encryption algorithms. Signcrypion combines the functionality of digital signature and encryption in a single logical step [2].

Gamage extended the concept of proxy signature and proposed a new cryptography primitive called proxy signcrypion [3]. Proxy signcrypion scheme allows the original user (Alice) to transfer the signing capability to the trusted entity known as proxy signcrypter. The proxy signer signcrypt the message on behalf of the original user and sends to the recipient (Bob) . Proxy Signcrypion schemes based on discrete logarithm problem (DLP) and elliptic curve discrete logarithm problem (ECDLP) were proposed. The common problems of these schemes are high computational cost and more communication overhead. In this paper, proxy signcrypion based on hyper elliptic curve is proposed, whose security relies on the hardness of the hyper elliptic curve discrete logarithm problem (HECDLP). In order to achieve the same security level, NIST has recommended key length of 80 bits for HECC, 1024 bits for RSA and 160 bits for ECC. Hyper elliptic curve is of public key cryptosystem with small key length. The proposed scheme saves the computational and communication cost due to its shorter key size. The scheme achieves the security properties.

1.1. Preliminaries

Let q be a prime number, where $q \geq 2^{80}$ and $F_q$is a finite field of order q. Hyperelliptic curve $C(Fq)$ over finite field F_q be define the equation (1)

$$C: y^2 + h(x)y = f(x) \bmod q \qquad (1)$$

Where $h(x) \in F[x]$ is a polynomial and degree$h(x) \leq g$ and $f(x) \in F[x]$ is a monic polynomial and degree $f(x) \leq 2g + 1$. Unlike points on elliptic curve, the points on hyper elliptic curve do not form a group. Divisor D is a finite formal sum of points on hyper elliptic curve and represented in *mumford* form as:

$$D = \big(u(x), v(x)\big) = \left(\sum_{i=0}^{g} u_i\, x^i\, , \sum_{i=0}^{g-1} v_i\, x^i\right)$$

Divisor form an Abelian group called Jacobian group $J_c(F_q)$ and the order of Jacobian group $o(J_c(F_q))$ is defined as

$$\left|\left(\sqrt{q} - 1\right)^{2g}\right| \leq o(J_c(F_q)) \leq \left|\left(\sqrt{q} + 1\right)^{2g}\right|$$

1.2. Definition

HECDLP

Let D be divisor of order n in the Jacobian group $J_c(F_q)$, find an integer $x \in F_q$, such that :

$$D_1 = x.D$$

1.3. Participants

Signer (Alice): Signer signs the warrant and send to proxy.

Proxy: Proxy signcrypts the message on behalf of the original signer and sends to the verifier.

Verifier (Bob): The verifier checks the validity of signcrypted text and accepts or rejects it accordingly.

The rest of the paper is organized as follows. In section 2 we discuss the related work. Section 3 discusses the proposed scheme , section 4 discusses the security analysis, section 5 discusses cost analysis and the last section is conclusion.

2. Literature Review

C. Gamage and his colleagues [3] Introduced first Proxy Signcrypion based on IF. The limitation of scheme is that it requires secure channel between signer and proxy. Y. Zhou and his colleagues [4] Introduced warrant-based proxy signcrypion scheme based on integer factorization assumption. The scheme required secured channel between signer and proxy. H. Elkamchouchi and his colleagues [5] Proposed a New Proxy Signcrypion scheme with DSA verifier based on the work of Shin DSA verifiable signcrypion.

The proposed scheme provides the property like public verifiability; authentication, signature and confidentiality are achieved through an unsecured channel. H. Elkamchouchi and his colleagues [6] introduced a new proxy signcrypion scheme based on the combination of hard problems: i.e. IFP, DLP, DHP, and irreversibility of a One-Way Hash Function (OWHF). The authors claimed that the combination of these hard problems provides strong security as compared to existing schemes. M. Elkamchouchi and his colleagues [7] introduced two proxy signcrypion schemes based on two different hard problems. The first one is based on Discrete Logarithm Problem (DLP) and the second one is based on elliptic curve discrete logarithm problem (ECDLP). The computational complexity is smaller than the other schemes in the literature.

3. Proposed Scheme

Our scheme consists of six phases: setup phase, key generation phase, proxy key generation phase, proxy key verification phase, signcrypion phase, and Unsigncryption phase.

3.1. Setup

There exists a system authority (SA) whose tasks are to initialize the system and to manage the public directory as:

Select and publish the following parameters:

- C: Secure Hyperelliptic curve C
- D:Divisor of prime order $\mathrm{n} \geq 2^{80}$
- $\psi : \mathrm{J_c}\left(\mathrm{F_q}\right) \longrightarrow \mathrm{Z_q}$ be a mapping function used to map Jacobian group element to an integer.
- h: a one way hash function
- $\mathrm{m/c}$: message/ciphertext
- $\mathrm{E_k}/\ \mathrm{D_k}$: Symmetric encryption/decryption using session key k
- $\perp$: message is not encrypted or signed correctly

3.2. Key Generation

Alice, Proxy, Bob generate their private keys and compute their respective public keys as:

Alice: Selects a random number $x_a \in \{0, 1, 2, \ldots\ldots, q-1\}$as his private key and computes public key Y_a: $Y_a = x_a D$.

Proxy Signer: Selects a random number $x_p \in \{0, 1, 2, \ldots\ldots, q-1\}$as his private key and computes public key Y_p: $Y_p = x_p D$.

Bob: Selects a random number $x_b \in \{0, 1, 2, \ldots\ldots, q-1\}$ as his private key and computes public key Y_b: $Y_b = x_b D$.

3.3. Proxy key generation

In this phase Alice signs a warrant message and sends to proxy.

Alice

- Randomly chose d
- Compute $(\alpha, \beta) = T = d.D$
- Compute $\gamma = \left(d - x_a . h(\alpha, m_w)\right) mod n$

Send (α, γ, m_w) to proxy

3.4. Proxy key verification phase

Proxy checks the validity of warrant message, whether the message is sent by the original signer or not. If the message is from original signer then accepts it otherwise rejects it.

Proxy

- Compute $T' = \gamma . D + h(\alpha, m_w). Y_a$

3.5. Proxy Signcrypion phase

In this phase the Proxy Signcrypt a message on behalf of the original user (Alice).

Proxy

1. Compute $skp = (x_p + \gamma) mod q$
2. Randomly chosen $w \in \{0,1,2, \ldots \ldots n\}$
3. Compute $(k_1 || k_2) = K = w. Y_b mod n$
4. Compute $c = E_{k_1}(m)$
5. Compute $r = h(m, k_2)$
6. Compute $s = (w - r * skp)$
7. Send $(c, \gamma, m_w, \alpha, s, r)$ to verifier

3.6. Unsigncryption phase

In proxy Unsigncryption phase only the verifier (Bob) can recover the plain text from cipher text.

Bob

1. Compute $(k_1 , k_2) = K = s. Y_b + r * x_b(T + Y_p - h(\alpha, m_w). Y_a)$
2. Compute $m = D_{k_1}(c)$
3. Compute $r' = h(m, k_2)$
4. Accept if $r' = r$

4. Security Analysis

The proposed scheme ensures the confidentiality, warrant integrity, message integrity, warrant unforgeability, message unforgeability, sender non- repudiation, and sender public verifiability.

4.1. Confidentiality

In our scheme, if the attacker wants to derive the original message, he must get the secret key K. There are many cases that the attacker can try to derive the secret key K.

Case 1: if the attacker computes K from Eq (1), then he needs secret parameter w. However, he has to solve HECDLP first, as he just knows the pointY_b , ~~It~~ which is computationally infeasible for the attacker to get the point K from Eq.(1)

$$K = w.Y_b \qquad (1)$$

Case 2: The attacker can get w from Eq (2). In this case the attacker needs the private key of proxy x_p . To calculate proxy's private key, one has to solve the HECDHP as follows:

$$s = (w - r * skp) \quad (2)$$

$$r = h(m, k_2) \qquad (3)$$

$$skp = \left(x_p + \gamma\right) \quad (4)$$

4.2. Warrant Integrity

Our proposed scheme provides warrant integrity. The sender calculates hash function $h(\alpha, m_w)$ of the warrant and sends to the proxy. If the attacker changes m_w into $m_w'.leth'(\alpha, m_w')$ then he constructs γ using Eq. (5). Thus the attacker finds the private key x_i of the sender from Eq. (6) and random number d from Eq. (7). To generate x_i and d is equivalent to solve two HECDLP which is computationally infeasible for attacker. For integrity the proxy checks the equality of Eq. (8), if the equation satisfies then warrant can’t change.

$$\gamma = \left(d - x_i.h(\alpha, m_w)\right) \quad (5)$$

$$Y_i = x_i.D \qquad (6)$$

$$T = d.D \qquad (7)$$

$$T = \gamma.D + h(\alpha, m_w) \quad (8)$$

4.3. Message integrity

The proxy calculates the digest/ hash function $r = h(m, k_2)$ of the message$m$, and sends to the recipient. If the attacker changes the original cipher text C to C', the related message is changed to m'. Let$r' = h(m', k_2)$. By the property of one-way hash function, it is computationally infeasible for the attacker to modify C to C'. The recipient can check the integrity of the message by comparing $r' = r$ then there is no change in the message and message from original sender.

4.4. Warrant unforgeability

In our scheme the attacker can’t generate the valid signature for warrant without the private key of sender. To generate valid signature from Eq. (5) the attacker needs the random number d and private key of sender x_i. To find sender's private key x_i from Eq. (6) and d from Eq. (7) as complex as to solve two HECDLP which is computationally infeasible.

4.5. Message unforgeability

In our scheme only proxy generates a valid signature for message. When attacker/ verifier generates a valid signature for message then he gets private key of proxy x_p from Eq (8) and session key $K$from Eq (11).

Attacker

$$Y_p = x_p . D \qquad (8)$$

$$s = (w - r * skp) \qquad (9)$$

$$skp = (x_p + \gamma) mod n \qquad (10)$$

Verifier (Recipient)

$$s = (w - r * skp) \qquad (9)$$

$$K = w . Y_v mod n \qquad (11)$$

4.6. Sender non repudiation

When dispute occurs between sender and proxy then trusted third party/judge verifies and decides that warrant m_w is from the sender or not. The proxy provides (T, γ, m_w) to the third party/judge. The judge performs the following steps:

- Verifies sender's public key Y_i by using certificate.
- Uses the one-way hash function to generate $h(\alpha, m_w)$.
- If$(Y_i = T - \gamma . D / h(\alpha, m_w))$ from Eq. (12) then the sender has sent (T, γ, m_w) to the proxy; otherwise not.

$$T = \gamma . D + h(\alpha, m_w) . Y_i \qquad (12)$$

4.7. Sender public verifiability

The proposed scheme provides sender public verification. Using Eq. (13) anybody can verify that warrant m_w is sent by the valid sender or not.

$$\left(Y_i = T - \gamma . \frac{D}{h(\alpha, m_w)} \right) \qquad (13)$$

5. Scheme Correctness Analysis

In this section we analyze the correction proofs of proposed scheme.

5.1. Proof 01

The following equations show the correctness between sender and proxy.

$\gamma . D + h(\alpha, m_w). Y_a$

$= \left(d - x_a . h(\alpha, m_w)\right). D + h(\alpha, m_w). Y_a$

$= \left(d - x_a . h(\alpha, m_w)\right). D + h(\alpha, m_w). x_a . D$

$= D(\left(d - x_a . h(\alpha, m_w)\right). + h(\alpha, m_w). x_a)$

$= D(d - x_a . h(\alpha, m_w) + x_a . h(\alpha, m_w))$

$= D(d - x_a . h(\alpha, m_w) + x_a . h(\alpha, m_w))$

$= d . D = T$

5.2. Proof 02

The following equations show the correctness between proxy and verifier.

$s . Y_b + \ c * x_b(T + Y_p - h(\alpha, m_w). Y_a)$

$= (w - c * skp). x_b . D + c * x_b\left(d . D + x_p . D - h(\alpha, m_w). x_a . D\right)$

$= (w - c * skp). x_b . D + c * x_b . D\left(d + x_p - h(\alpha, m_w). x_a\right)$

$= x_b . D(w - c * skp) + c\left(d + x_p - h(\alpha, m_w). x_a\right)$

$= x_b . D(w - c * skp + cd + cx_p - ch(\alpha, m_w). x_a)$

$= x_b . D(w - c\left(skp - d - x_p + h(\alpha, m_w). x_a\right))$

$= x_b . D(w - c\left(skp - x_p - (d - h(\alpha, m_w). x_a\right))$

$= x_b . D(w - c\left(skp - x_p - \gamma\right))$

$= x_b . D(w - c * \left(x_p + \gamma - x_p - \gamma\right))$

$= x_b.D * (w)$

$= w * x_b.D$

$= w.Y_b \ = K$

6. Cost analysis

In this section we analyze and compare two types of costs; computational cost and communication overhead of the proposed proxy signcryption scheme and existing proxy signcryption scheme.

6.1. Computational cost

The computational cost means the amount of computational efforts to be invested both by the sender and recipient of a message. Generally, the computational cost is estimated by counting the number of dominant operations involved. Typically these operations include private key encryption and decryption, hashing, modular addition, multiplication, division and exponentiation The Elliptic curve point Multiplication (ECPM) and hyper-elliptic curve divisors scalar multiplication (HECDM) are is the most costly operations.The proposed scheme is compared and analyzed in terms of HECPM (most expensive operation) with the existing scheme.It is observed that the single scalar multiplication consumes 4.24 ms for elliptic curve point multiplication (ECPM) and 2.2 ms for hyper elliptic curve divisors scalar multiplication (HECDM) on a PC running jdk 1.6 having two cores of Intel CPU with processing speed of 2.00 GHz and primary memory capacity of 4 GB operating with Microsoft Windows vista [8]. The Figure 1 shows the comparative computation cost of proxy signcrypion vs proposed proxy Signcryption.

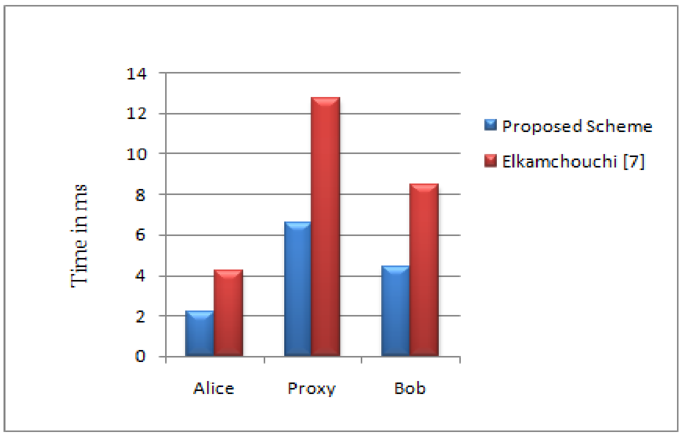

Figure 1: Computational Cost Comparison

6.2 Communication cost

Communication cost depends on the choice of parameters and amount of information and is calculated as the size of plain text vs signcrypted text in bits of existing proxy signcrypion and proposed proxy signcrypion scheme. The Figure 2 shows the Comparative Communication overhead analysis of proxy signcrypion vs proposed proxy signcrypion.

7. Conclusion

This paper presents an efficient proxy signcrypion based on hyper elliptic curves. As compared to previous schemes the proposed scheme reduces 48% computation cost and 27% communication overhead. The scheme is suitable for devices in resource constrained environments due hyper elliptic curve cryptosystem's shorter key size. The proposed scheme also achieves the security properties like Confidentiality, Warrant Integrity, Warrant unforgeability, Message integrity, Message unforgeability, Sender non repudiation, Sender public verifiability.

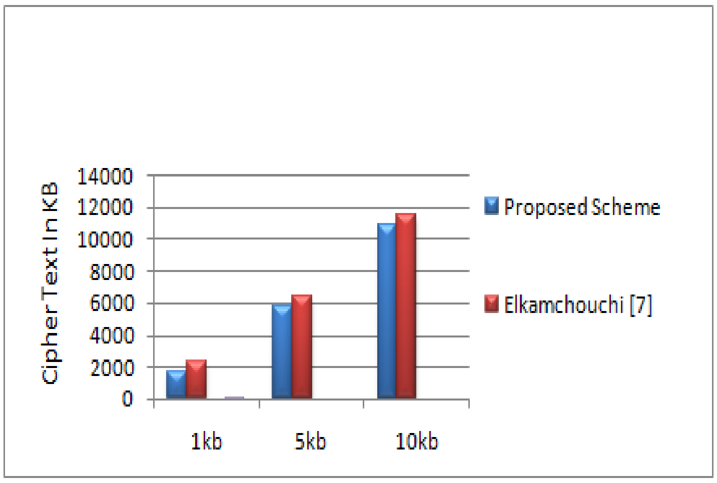

Figure 2: Communication Overhead Analysis

References

[1] M. Mambo, K. Usuda, and E. Okamoto, “Proxy signatures for delegating signing operation,” Proc. 3rd ACM Conf. Comput. Commun. Secur. - CCS ’96, pp. 48–57, 1996.

[2] Y. Zheng, “Digital Signcryption or How to Achieve Cost (Signature and Encryption) Cost (Signature) + Cost (Encryption),” Advances in Cryptology, LNCS, Vol. 1294. Springer-Verlag, pp.165–179, 1997.

[3] C. Gamage, J. Leiwo, and Y. Zheng, “An Efficient Scheme for Secure Message Transmission using Proxy-Signcryption,” in Proceedings of the 22nd Australasian Computer Science Conference ,pp.420-431, Springer,1999.

[4] Y. Zhou, Z. Cao, and R. Lu, “Constructing Secure Warrant-Based Proxy Signcryption Schemes,” pp. 172–185, 2005.

[5] D. H. Elkamshoushy, a. K. AbouAlsoud, and M. Madkour, “New proxy signcryption scheme with DSA verifier,” Natl. Radio Sci. Conf. NRSC, Proc., no. Nrsc, 2006.

[6] H. Elkamchouchi, “Based on a Combination of Hard Problems,” no. October, pp. 5123–5127, 2009.

[7] E. F. A. Elkhair, “An Efficient Proxy Signcryption Scheme,” vol. 1, no. 2, pp. 7–19, 2013.

[8] R. Ganesan and M. Gobi, “E-Commerce Channel 4 Hyper-Elliptic Curve Cryptosystems.” Int J Netw Secur 11(3):121–127,2010.