AN EFFICIENT PROXY SIGNCRYPTION SCHEME BASED ON THE DISCRETE LOGARITHM PROBLEM

Hassan M. Elkamchouchi [1], Eman F. Abu Elkhair[2] and Yasmine Abouelseoud[3]

[1]Elec. Eng. Dept, Fac. of Eng., Alexandria University.
helkamchouchi@ieee.org
[2]Elec. Eng. Dept, Fac. of Eng., Kafr El-Sheikh University.
eman.abouelkhair@eng.kfs.edu.eg
[3]Math. Eng. Dept, Fac. of Eng., Alexandria University.
yasmine.abouelseoud@gmail.com

ABSTRACT

Signcryption is a cryptographic primitive which simultaneously provides both confidentiality and authenticity in a single logical step. In a proxy signature scheme, an original signer delegates his signing power to a proxy agent, who signs a message on behalf of him. This paper introduces a new proxy signcryption scheme based on the Discrete Logarithm Problem (DLP) with a reduced computational complexity compared to other schemes in literature. In this proposed scheme, the receiver is the only one who can verify the origin of the ciphertext. Moreover, in this scheme, an authorized proxy signcrypter can create valid proxy signatures after verifying the identity of the original signcrypter. The proposed scheme achieves the various desirable security requirements. An elliptic curve based version of the proposed proxy signcryption scheme has been implemented using Mathematica for realistic (256-bit) parameters to emphasize the ease of its practical use.

KEYWORDS

Signcryption, Proxy Signcryption, Discrete Logarithm Problem (DLP), Elliptic Curve Discrete Logarithm Problem (ECDLP), Security Requirements.

1. INTRODUCTION

The proxy signature is a cryptographic primitive that was first introduced by Mambo, Usuda and Okamoto [1]. The scheme allows an entity, called the original signer, to designate another entity, called a proxy signer, to sign messages on its behalf. The proxy signature primitive has found numerous practical applications, particularly in distributed computing where delegation of rights is quite common, such as in e-cash systems, global distribution networks, grid computing, mobile agent applications, and mobile communications. This is because, in the areas of computer communications and electronic transactions, one of the important topics is how to send data in a confidential and authenticated way.

Usually, the confidentiality of delivered data is provided by an encryption algorithm, and the authentication of messages is guaranteed by digital signatures. In 1997, Zhang [2] proposed a cryptographic primitive, called signcryption, to achieve the combined functionalities of digital signatures and encryption in an efficient manner. Many researchers have proposed a variety of signcryption schemes [3]. One of these variants is a proxy signcryption scheme which efficiently combines a proxy signature scheme with an encryption mechanism. A proxy signcryption scheme allows an entity to delegate its authority of signcryption to a trusted agent. The proxy signcryption scheme is useful for applications that are based on unreliable datagram style network communication model, where the messages are individually signed and not serially linked via a

DOI : 10.5121/ijitmc.2013.1202

session key to provide authenticity and integrity. The first proxy signcryption scheme was proposed by Gamage et.al [4] in the traditional PKI based setting.

In this paper, a new efficient proxy signcryption scheme is proposed, whose security relies on the hardness of the discrete logarithm problem. The rest of the paper is organized as follows. Section 2 discusses the computationally hard problems; both the discrete logarithm problem (DLP) and the related elliptic curve discrete logarithm problem (ECDLP). In Section 3, the security requirements for proxy signcryption schemes are provided. Section 4 introduces the proposed proxy signcryption scheme based on the DLP together with its proof of correctness, security analysis and performance analysis. In Section 5, a variant of the proposed proxy signcryption scheme based on the ECDLP is presented along with its related analysis. Section 6 involves an example with realistic parameters to demonstrate the ease of implementation of the proposed scheme using Mathematica 7.0 program. Finally, Section 7 concludes this paper.

2. Computationally Hard Problems

2.1 The Discrete Logarithm Problem (DLP) [5,6]

Let p and q be two large primes satisfying $q|p-1$, and g a generator of order q over $GF(p)$. The discrete logarithm problem is, given an instance (y, p, q, g) , where $y = g^x \bmod p$ for some $x \in Z_q$, to derive x.

2.2. Discrete Logarithm (DL) Assumption [5,6]

A probabilistic polynomial-time algorithm B is said to (t, ε) break the DLP if given a DLP instance (y, p, q, g) where $y = g^x \bmod p$ for some $x \in Z_q$, B can derive x with probability ε after running at most t steps. The probability is taken over the uniformly and independently chosen instance and over the random bits consumed by B.

Definition 1 The (t, ε) DL assumption holds if there is no probabilistic polynomial-time adversary that can (t, ε) break the DLP.

2.3. Elliptic Curve Discrete Logarithm Problem (ECDLP) [5,7]

An elliptic curve group is described using multiplicative notation, then the elliptic curve discrete logarithm problem is: given points P and Q in the group Z_p , find a number such that $kP = Q; k$ is called the discrete logarithm of Q to the base P .

3. Syntax and Security Requirements of a Proxy Signcryption Scheme

3.1 Syntax

A proxy signcryption scheme involves three parties: the original signcrypter /sender, the proxy signcrypter and the unsigncrypter/verifier. Each party has a secret and a corresponding public key.

The proxy signcryption scheme can be viewed as the combination of a general proxy signature and a signcryption scheme. Let U_i be the original signcrypter, whose private key is x_i. He delegates his signing rights to a proxy signcrypter U_p , whose private key is x_p. A warrant is used to delegate the signing rights. The unsigncrypter U_v with a secret key x_v can decrypt the ciphertext and check the signature validity. A proxy signcryption scheme consists of the following algorithms:

Setup: Taking as input 1^k where k is a security parameter, the algorithm generates the system-wide parameters.

Proxy-Credential-Generation (PCG): The PCG algorithm takes as input the system parameters and the private key of the original signer. It outputs a corresponding proxy credential.

Proxy-Signcryption-Generation (PSG): The PSG algorithm takes as input the system parameters, a proxy credential, a message *m*, the public key of the designated verifier and the private key of the proxy signer. It generates a cryptogram δ.

Proxy- Unsigncryption -Verification (PUV): The PUV algorithm takes as input the system parameters, a cryptogram δ, the private key of the designated verifier and the public keys of the original and the proxy signers. It outputs **True** if δ involves a valid signature for *m*. Otherwise, an error symbol $\perp$ is returned as a result.

3.2. Security Requirements of a Proxy Signcryption Scheme

A secure proxy signcryption scheme should satisfy the following requirements[8,9]:

1. Verifiability: From the proxy signcryption text, the recipient can be convinced of original sender's agreement on the signcrypted message.

2. Unforgeability: The original sender and other third parties cannot create a valid proxy signcryption text.

3. Identifiability: Anyone can determine the identity of the corresponding proxy sender from the proxy signcryption text.

4. Prevention of Misuse: The proxy sender cannot use the proxy key for other purposes than generating a valid proxy signcryption text.

5. Confidentiality: Except the recipient, no one can extract the plaintext from the proxy signcryption text.

6. Non-repudiation: The recipient can efficiently prove to any third party that the message indeed originated from a specific sender on behalf of an original sender.

7. Forward Security: An attacker cannot reveal the messages signcrypted before even with the knowledge of the sender's private key.

4. The Proposed Scheme Based on the DLP

The original user key pair is (x_i, y_i), the proxy key pair is (x_p, y_p), and the recipient key pair is (x_v, y_v). The proposed DLP-based proxy signcryption scheme is shown in Figure 1.

4.1. The Proposed Scheme Construction

The proposed scheme is demonstrated over a finite field as follow.

4.1.1. Setup

Taking as input 1^k, the system authority (SA) selects two large primes p and q, where $|q| = k$ and $q|p-1$. Let g be a generator of order q and $H:\{0,1\}^k \times Z_q \to Z_q$. The system-wide parameters = $\{p, q, g, H\}$ are then published.

The original user U_i chooses his private key $x_i \in Z_q$ and computes the public key as $y_i = g^{x_i}$.

The proxy U_p chooses his private key $x_p \in Z_q$ and computes the public key as $y_p = g^{x_p}$.

The recipient U_v chooses his private key $x_v \in Z_q$ and computes the public key as $y_v = g^{x_v}$.

4.1.2. Proxy-Credential-Generation (PCG)

Let U_i be an original user delegating his signing power to a proxy signer U_p. U_i first chooses $d \in_R Z_q$ to compute $t \equiv (g^d \bmod p) \bmod q$ and $\sigma \equiv (d - x_i.h(m_w, t) \bmod p) \bmod q$, where m_w is a warrant consisting of the identifiers of the original and the proxy signers, the delegation duration and so on. (σ, m_w, t) is then sent to U_p. Upon receiving (σ, m_w, t), U_p computes $(g^{\sigma}.y_i^{h(m_w,t)} \bmod p) \bmod q$ and performs check its validity as follow:

$t \equiv (g^{\sigma}.y_i^{h(m_w,t)} \bmod p) \bmod q$.

If t is not equal to the right hand side, the proxy requests a new (σ, m_w, t) to be sent again.

The verification of the above equation proceeds as follows:

$$g^{\sigma}.y_i^{h(m_w,t)} \equiv (g^{(d-x_ih(m_w,t))}.y_i^{h(m_w,t)}) \bmod p) \bmod q$$

$$\equiv (g^d.g^{-x_ih(m_w,t)}.y_i^{h(m_w,t)}) \bmod p) \bmod q$$

$$\equiv (g^d.y_i^{-h(m_w,t)}.y_i^{h(m_w,t)}) \bmod p) \bmod q$$

$$\equiv (g^d \bmod p) \bmod q = t$$

After the proxy authenticates the original signer, the proxy computes the secret proxy key as follows: $skp \equiv (x_p + \sigma) \bmod q$.

4.1.3. Proxy-Signcryption-Generation (PSG)

In this phase, the proxy will do the following steps to sign and encrypt the message *m*. The proxy chooses a random number $w \in_R Z_q$ and computes:

1- $k \equiv h((y_r^w \bmod p) \bmod q)$

2- Splits k into k_1, k_2

3- $s_1 = E_{k_1}(m)$

4- $c = h(m, k_2)$

5- $s_2 \equiv (w - c * skp) \bmod q$

The proxy sends $\delta = (\sigma, m_w, t, s_1, s_2, c)$ to the receiver.

4.1.4. Proxy- Unsigncryption -Verification (PUV)

In this phase, the receiver decrypts the message and checks the signature validity.

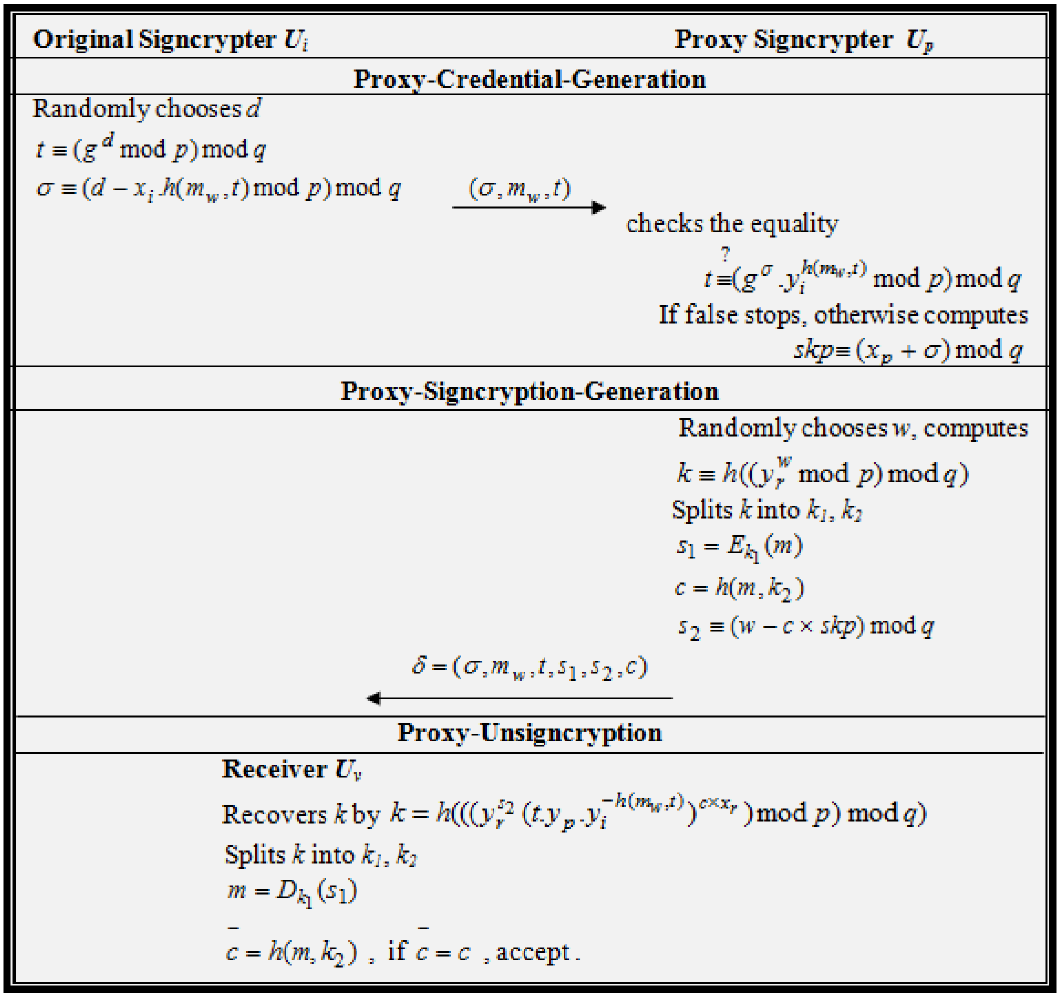
1- The receiver recovers the key k by computing:

$$k = h(((y_r^{s_2}(t.y_p.y_i^{-h(m_w,t)})^{c*x_r}) \bmod p) \bmod q)$$

2- Splits k into k_1,k_2

3- Computes $m = D_{k_1}(s_1)$

4- Computes $\bar{c} = h(m, k_2)$ and accepts if $\bar{c} = c$.



Original Signcrypter U_i **Proxy Signcrypter U_p**

Proxy-Credential-Generation

Randomly chooses d

$t \equiv (g^d \bmod p) \bmod q$

$\sigma \equiv (d - x_i.h(m_w, t) \bmod p) \bmod q$ $\xrightarrow{(\sigma, m_w, t)}$

checks the equality

$t \overset{?}{\equiv} (g^{\sigma}.y_i^{h(m_w,t)} \bmod p) \bmod q$

If false stops, otherwise computes

$skp \equiv (x_p + \sigma) \bmod q$

Proxy-Signcryption-Generation

Randomly chooses w, computes

$k \equiv h((y_r^w \bmod p) \bmod q)$

Splits k into k_1, k_2

$s_1 = E_{k_1}(m)$

$c = h(m, k_2)$

$s_2 \equiv (w - c \times skp) \bmod q$

$\xleftarrow{\delta = (\sigma, m_w, t, s_1, s_2, c)}$

Proxy-Unsigncryption

Receiver U_v

Recovers k by $k = h(((y_r^{s_2}(t.y_p.y_i^{-h(m_w,t)})^{c \times x_r}) \bmod p) \bmod q)$

Splits k into k_1, k_2

$m = D_{k_1}(s_1)$

$\bar{c} = h(m, k_2)$, if $\bar{c} = c$, accept .

Figure 1 The three phases of the proposed DLP signcryption scheme

4.2. Proof of Correctness

The following equations demonstrate the correctness of the proposed scheme

$$k = h(((y_r^{s2}(t.y_p.y_i^{-h(m_w,t)})^{c*x_r}) \bmod p) \bmod q)$$

$$= h(((y_r^{w-c*skp}(y_p.g^d.g^{-h(m_w,t)*x_i})^{c*x_r}) \bmod p) \bmod q)$$

$$= h(((y_r^{w-c*skp}(g^{x_p}.g^{d-h(m_w,t)*x_i})^{c*x_r}) \bmod p) \bmod q)$$

$$= h(((y_r^{w-c*skp}(g^{x_p}.g^{\sigma})^{c*x_r}) \bmod p) \bmod q)$$

$$= h(((y_r^{w-c*skp}.g^{x_p*c*x_r}.g^{\sigma*c*x_r}) \bmod p) \bmod q)$$

$$= h(((y_r^{w-c*skp}.y_r^{x_p*c}.y_r^{\sigma*c}) \bmod p) \bmod q)$$

$$= h(((y_r^{w-c*skp}.y_r^{(\sigma+x_p)*c}) \bmod p) \bmod q)$$

$$= h(((y_r^{w-c*skp} . {y_r}^{skp*c}) \bmod p) \bmod q)$$

$$= h(((y_r^{w}) \bmod p) \bmod q)$$

4.3. Security Analysis

In what follows, the security properties of the proposed scheme are investigated.

1. Verifiability: From the proxy unsigncryption phase, the receiver can be convinced that the proxy sender has the original sender's signature on the warrant. The warrant also contains the identity information of the original sender, the proxy sender and the limit of the delegated signcrypting capacity etc. Therefore, the receiver can be convinced of the original sender's agreement on the signcrypted message. Thus, the scheme satisfies the verifiability requirement.

2. Unforgeability: Because the proxy sender uses his private key x_p to generate the proxy signcryption key $skp \equiv (x_p + \sigma) \bmod q$, no one can get the proxy signcryption key skp except the proxy sender himself. To create a valid proxy signcryption $s_2 \equiv (w - c \times skp) \bmod q$, one needs to compute the value of w and skp . But due to the intractability of the DLP, it is difficult to compute w and skp. Thus, except the proxy signcrypter, no one can create a valid proxy signcryption text. Thus, the proposed scheme supports unforgeability.

3. Identifiability: The proxy signcrypted text $\delta = (\sigma, m_w, t, s_1, s_2, c)$ contains the warrant m_w. Moreover, the verification equation

$$k = h(((y_r^{s_2}(t.y_p.y_i^{-h(m_w,t)})^{c*x_r}) \bmod p) \bmod q)$$

includes the original signcrypter public key y_i and the proxy signcrypter public key y_p. Hence, anyone can determine the identity of the corresponding proxy signer from a proxy signature. So, the scheme satisfies the Identifiability requirement.

4. Prevention of Misuse: In the proposed proxy signcryption scheme, using the warrant m_w , the limit of the delegated signcrypting capacity is clearly specified in the warrant and then the proxy sender cannot signcrypt the messages that have not been authorized by the original sender.

5. Confidentiality: The message is encrypted so that it can only be decrypted by the intended recipient in possession of the secret session key. Only the verifier can recover the key by which the encryption process is constructed because the verifier uses his secret key to recover the encryption /decryption key as follows:

$$k = h(((y_r^{s_2}(t.y_p.y_i^{-h(m_w,t)})^{c*x_r}) \bmod p) \bmod q) .$$

Therefore, we conclude that the proposed scheme meets this security requirement.

6. Non-Repudiation: In this scheme, the original signer does not know the proxy signer's secret key x_p and the proxy signer does not know original signer's secret key x_i. Thus, neither the original signer nor the proxy signer can sign in place of the other party. Thus, the scheme provides non-repudiation.

7. Forward Security: Unsigncryption requires the knowledge of x_r. But, due to the intractability of the DLP, it is difficult to compute x_r from y_r. Thus the proposed scheme is forward secure.

4.4. Performance Analysis

In this section, the performance of the proposed signcryption scheme based on the DLP is discussed. The scheme is compared with another DLP-based scheme and this proved that the proposed signcryption scheme is more computationally efficient than the scheme in [10]. This comparison is provided in Table 2. Table 1 shows the abbreviations that will be used in the comparison.

Table 1 Time abbreviations

Symbol	**Operation**
$\mathbf{T_e}$	the time for performing a modular exponentiation computation
$\mathbf{T_{mult}}$	time required for executing a modular multiplication in a finite field
$\mathbf{T_h}$	time required for executing one-way hash function
$\mathbf{T_{encr}}$	time required by the system for executing an encryption operation
$\mathbf{T_{decr}}$	time required by the system for executing a decryption operation

Table 2. The comparison of the proposed signcryption scheme based on the DLP with the scheme in [10]

Phase	**Scheme in [10]**	**The proposed**
System Construction	$3T_e+2T_h+3\ T_{mult}$	$3T_e+2T_h+3\ T_{mult}$
Signcryption	$1T_e+1T_h+1T_{mult}+\ 1T_{encr}$	$1T_e+1T_h+1T_{mult}+\ 1T_{encr}$
Unsigncryption	$4T_e+2T_h+3T_{mult}+\ 1\ T_{decr}$	$3T_e+2T_h+3T_{mult}+\ 1\ T_{decr}$
Total	$8T_e+5T_h+7T_{mult}+1T_{encr}+\ 1\ T_{decr}$	$7T_e+5T_h+7T_{mult}+1T_{encr}+\ 1\ T_{decr}$

5. The Scheme over Elliptic Curves

Elliptic curve cryptography provides better security than that by other schemes such as the RSA [11], ElGamal [12], etc., with shorter keys, and this results in less storage requirements. Moreover, the ECDLP that was discussed in Section 2 is more difficult than the DLP.

In this version of the proposed scheme, the secret keys are chosen as random elements, where $x \in Z_q^*$. The system-wide parameters include an elliptic curve *E*, a point *G* on the elliptic curve with a prime order *q*. The corresponding public keys are computed as $Y = x.G$, where: (x_i, Y_i) is the original signer key pair, (x_p, Y_p) is the proxy signer key pair and (x_v, Y_v) is the recipient key pair. The ECDLP signcryption scheme is shown in Figure 2.

5.1. Proposed Scheme Construction

The proposed scheme over an elliptic curve is described in what follows.

5.1.1. Setup

The system authority (SA) selects two large primes *p* and *q* where $q|p-1$. An elliptic curve *E* is chosen with *G* is a generator point on the elliptic curve. The original user U_i chooses his private key $x_i \in Z_q$ and computes the public key as $Y_i = x_i.G$. The proxy U_p chooses his private key

$x_p \in Z_q$ and computes the public key as $Y_p = x_p.G$. The recipient U_v chooses his private key $x_v \in Z_q$ and computes the public key as $Y_v = x_v.G$.

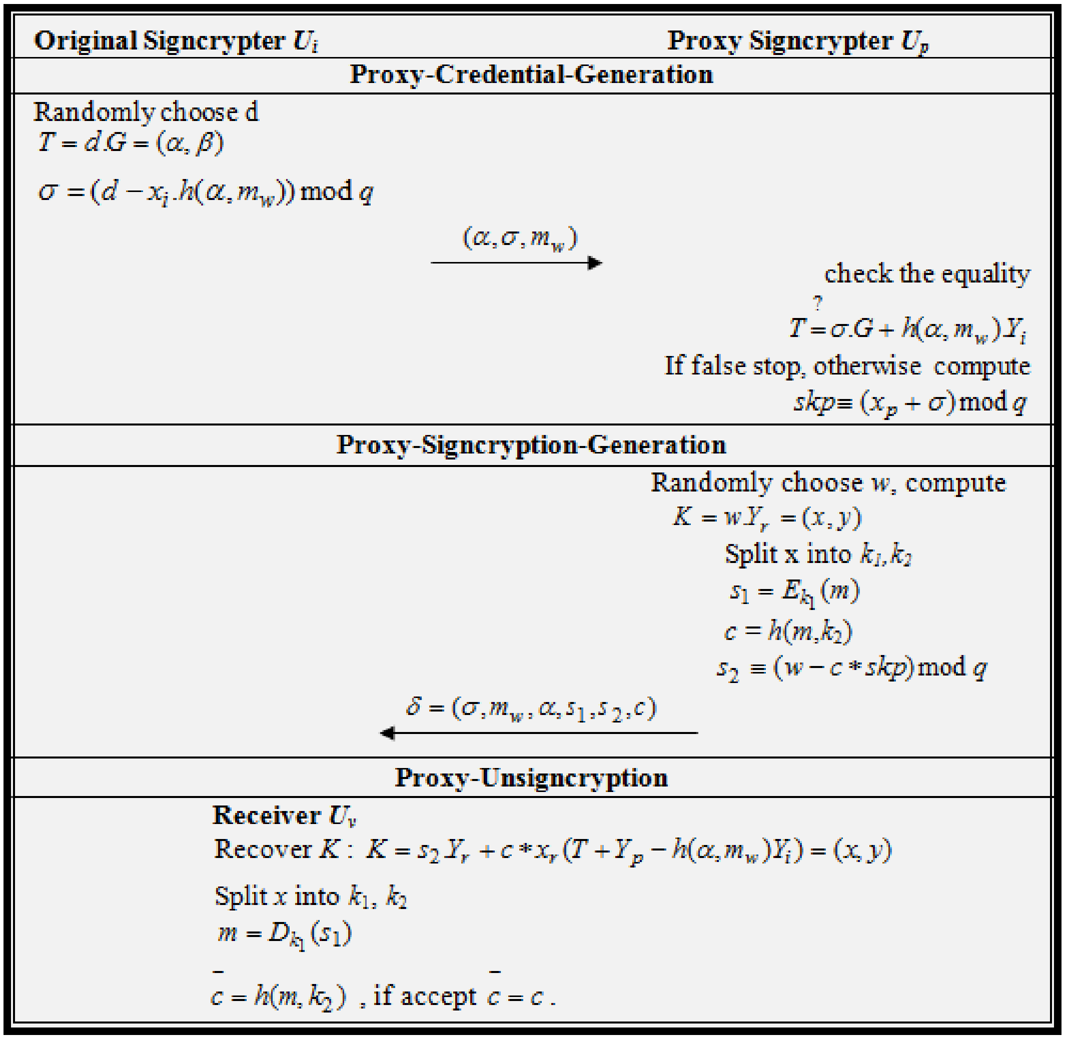

Original Signcrypter U_i
Proxy Signcrypter U_p
Proxy-Credential-Generation
Randomly choose d
$T = d.G = (\alpha, \beta)$
$\sigma = (d - x_i.h(\alpha, m_w)) \bmod q$
(α, σ, m_w)
check the equality
$T \overset{?}{=} \sigma.G + h(\alpha, m_w).Y_i$
If false stop, otherwise compute
$skp \equiv (x_p + \sigma) \bmod q$
Proxy-Signcryption-Generation
Randomly choose w, compute
$K = wY_r = (x, y)$
Split x into k_1, k_2
$s_1 = E_{k_1}(m)$
$c = h(m, k_2)$
$s_2 \equiv (w - c * skp) \bmod q$
$\delta = (\sigma, m_w, \alpha, s_1, s_2, c)$
Proxy-Unsigncryption
Receiver U_v
Recover K: $K = s_2 Y_r + c * x_r (T + Y_p - h(\alpha, m_w) Y_i) = (x, y)$
Split x into k_1, k_2
$m = D_{k_1}(s_1)$
$\bar{c} = h(m, k_2)$, if accept $\bar{c} = c$.

Figure 2 The three phases of the proposed ECDLP signcryption scheme

5.1.2. Proxy-Credential-Generation (PCG)

The original signer chooses a random number $d \in [1, q-1]$ and computes:

1- $T = d.G = (\alpha, \beta)$

2- $\sigma = (d - x_i.h(\alpha, m_w)) \bmod q$

The original signer sends (α, σ, m_w) to the proxy.

The proxy checks the validity of signature as follows: If $\sigma.G + h(\alpha, m_w).Y_i = T$,

The proxy computes the secret proxy key. Otherwise, the proxy requests a new (α, σ, m_w)-tuple.

The correctness of the verification of equation is demonstrated below:

$RHS = \sigma.G + h(\alpha, m_w).Y_i$

$$= (d - x_i.h(\alpha,m_w)).G + h(\alpha,m_w).Y_i$$
$$= d.G - x_i.h(\alpha,m_w).G + x_i.h(\alpha,m_w).G$$
$$= d.G = T = LHS$$

After the proxy authenticates the original signer, the proxy computes the secret proxy key as: $skp \equiv (x_p + \sigma) \bmod q$

5.1.3. Proxy-Signcryption-Generation (PSG)

In this phase, the proxy carries out the following steps to sign and encrypt a message *m*. The proxy chooses a random number $w \in_R Z_q$ and computes:

1- $K = w.Y_r = (x,y)$
2- Split x into k_1,k_2
3- $s_1 = E_{k_1}(m)$
4- $c = h(m,k_2)$
5- $s_2 \equiv (w - c * skp) \bmod q$

The proxy sends $\delta = (\sigma, m_w, \alpha, s_1, s_2, c)$ to the recipient.

5.1.4. Proxy- Unsigncryption -Verification (PUV)

In this phase, the receiver decrypts the message and verifies the alleged signature.

1- The receiver recovers the key K by computing:
$$K = s_2.Y_r + c * x_r(T + Y_p - h(\alpha,m_w)Y_i) = (x,y)$$

2- Split x into k_1,k_2

3- Compute $m = D_{k_1}(s_1)$

4- Compute $\bar{c} = h(m,k_2)$ and accept if $\bar{c} = c$.

5.2. Proof of Correctness

The following equations show the correctness of the proposed scheme

$$K = s_2.Y_r + c * x_r(T + Y_p - h(\alpha,m_w)Y_i)$$
$$= (w - c * skp)..Y_r + c * x_r.T + c * x_r.Y_p - c * x_r.h(\alpha,m_w)Y_i)$$
$$= w.Y_r - c * (x_p + \sigma)..Y_r + c * x_r.T + c * x_r.Y_p - c * x_r.h(\alpha,m_w)Y_i)$$
$$= w.Y_r - c * (x_p + d - x_i.h(\alpha,m_w))..Y_r + c * x_r.T + c * x_r.Y_p - c * x_r.h(\alpha,m_w)Y_i)$$
$$= w.Y_r - c * x_p.Y_r - c * d.Y_r + c * x_i.h(\alpha,m_w)..Y_r + c * x_r.T + c * x_r.Y_p - c * x_r.h(\alpha,m_w)Y_i)$$
$$= w.Y_r - c * x_p.x_r.G - c * d.x_r.G + c * x_i.h(\alpha,m_w).x_r.G + c * x_r.d.G + c * x_r.x_p.G$$
$$- c * x_r.h(\alpha,m_w)x_i.G)$$
$$= (w.Y_r) = K = (x,y)$$

5.3. Performance Analysis

Again, the performance of the proposed proxy signcryption scheme based on the ECDLP is analyzed and compared to the scheme by Yu Fang Chung [13]. It is found that the proposed scheme involves fewer computations than the scheme in [13]. Table 3 defines the notation that will be used in the comparison. Table 4 shows the comparison of the proposed signcryption scheme based on the ECDLP to that of Yu Fang Chung [13] .

Table 3: Comparison notation

Symbol	Operation
$\mathbf{T_{EC\text{-}mult}}$	time complexity required for executing multiplication operation on elliptic curve *E*
$\mathbf{T_{EC\text{-}add}}$	time complexity required for executing addition operation on elliptic curve *E*
$\mathbf{T_{mult}}$	time complexity required for executing modulus multiplication in a finite field
$\mathbf{T_{h}}$	time complexity required for executing one way dispersed row function operation
$\mathbf{T_{encr}}$	time complexity required by the system for executing encryption operation
$\mathbf{T_{decr}}$	time complexity required by the system for executing decryption operation

Table 4. The proposed signcryption compared with Yu Fang Chung [13]

Phase	Scheme in [13]	The proposed
System Construction	3 $T_{EC\text{-}mult}$	$3T_{EC\text{-}mult}+1T_{EC\text{-}add}+ 2T_{h}+ 2T_{mult}$
Signcryption	$7T_{EC\text{-}mult}+1T_{EC\text{-}add}+ 2T_{h}+ 1\ T_{mult}+ 1\ T_{encr}$	$1T_{EC\text{-}mult}+T_{h}+ 1T_{mult}+ 1\ T_{encr}$
Signcryption	$5T_{EC\text{-}mult}+2T_{EC\text{-}add}+ 2T_{h}+ 1\ T_{decr}$	$3T_{EC\text{-}mult}+3T_{EC\text{-}add}+ 1T_{h}+ 1\ T_{decr}$
Total	$15T_{EC\text{-}mult}+3T_{EC\text{-}add}+ 4T_{h}+1T_{mult}+1T_{encr}+ 1\ T_{decr}$	$7T_{EC\text{-}mult}+4T_{EC\text{-}add} +3T_{h}+ 3T_{mult}+ 1\ T_{encr}+ 1\ T_{decr}$

6. NUMERICAL EXAMPLE

Here is a numerical example as a proof-of-concept, which has been implemented using Mathemtica 7.0 program. In this example, the parameters used are among the 256-bit recommended domain parameters for elliptic curves suggested in [14].

- p is the prime specifying the base field.
- *a* and *b* are the coefficients of the equation $y^2 \equiv (x^2 + a.x + b) \bmod p$ defining the elliptic curve.
- *G* = (*x* , *y*) is the base point, i.e., a point in *E* of prime order, with *x* and *y* being its x- and y-coordinates, respectively.
- *q* is the prime order of the group generated by *G*.

6.1. Setup

p=
76884956397045344220809746629001649093037950200943055203735601445031516197751
a=
56698187605326110043627228396178346077120614539475214109386828188763884139993
b=
17577232497321838841075697789794520262950426058923084567046852300633325438902
x=
63243729749562333355292243550312970334778175571054726587095381623627144114786
y=
38218615093753523893122277964030810387585405539772602581557831887485717997975
q=

76884956397045344220809746629001649092737531784414529538755519063063536359079

6.2. Key Generation

x_i =(original signer secret key)
36970917057604995392163116089274538973784638899374105544422616072686917434464

x_p =(proxy signer secret key)
67388075705982992220549471663190937701872496524845891485128629547185746636907

x_v = (receiver secret key)
21130384841973895085117369749801061902813331654571109795701819708535032495858

m_w =(m_w identifiers of original user and proxy)
7372414995659354621756266831038493007687103260450141912601460306622985447522

Message= Maryam
Y_i =(original user public key)
{7722331765656151675963766905894768609675094184151969406165403177166454051199,
44957596601729384214424379527494636620054014645374456997318769737528198317509}

Y_p =(proxy public key)
{7877034842700288155439664896882850632711435866587094426483917070356571715558,5
2217726305833992167110353980070706594254804748676939014424841041113837483648}

Y_v =(verifier public key)
{74562134152238745110426266905280819603065543551183169953186297359971607384695,
50766633321475331806665713688372879615163255466237277745203762039855173719500}

6.3. Proxy Key Generation

d =
2066699979661083083227685742247885030508940010080093808165500809032761709671

T =
{11603919161609063432713514605716968075070369515679940565694322503179044255255,

65885113768109763301671143780299605591757178714021240525402700944123985384042}

σ =
18338614545713983859119699872924688512940666175940624769723361139711964578515

$\sigma.G + h(\alpha, m_w).Y_i$ =(verify the original user)
{11603919161609063432713514605716968075070369515679940565694322503179044255255,

65885113768109763301671143780299605591757178714021240525402700944123985384042}
=T

skp=
8841733854651631858859424907113977122075630916371986716096471623834174856343

6.4. Signcryption Generation

w=
22195343144847693995307624545183555130155159598294472744996959627283792967305

$K = w.Y_r = (x, y)$

{4614295587673439576491327078316083345217819169990767179865254042018489129602 5,
5129443178203224541650151276127876325609109585783286648323999557111638489654}

Splitting x into k_1 and k_2

$k_1 = 46142955876734395764913270783160833452$

$k_2 = 17819169990767179865254042018489129602 5$

$s_1 = E_{k_1}(m)$

46142955876734395764913205505257780417

$c = h(m, k_2) = 35948498$

$s_2 \equiv (w - c * skp) \bmod p$

70132592339047574638711924739288545169695431580974369346913867546313046834310

6.5. Signcryption Verification

$K = s_2.Y_r + c * x_r(T + Y_p - h(\alpha, m_w)Y_i) = (x, y)$

{4614295587673439576491327078316083345217819169990767179865254042018489129602 5,
5129443178203224541650151276127876325609109585783286648323999557111638489654}

Splitting x into k_1 and k_2

$k_1 = 46142955876734395764913270783160833452$

$k_2 = 17819169990767179865254042018489129602 5$

$m = D_{k_1}(s_1) =$ Maryam

$\bar{c} = h(m, k_2) =$ 35948498 = c

The receiver accepts the received cryptogram.

7. CONCLUSIONS

In this paper, a proxy signcryption scheme has been developed, in which the original signer delegates his signing rights to a proxy agent. However, the proxy signature is distinguishable from the original signer to protect a malicious proxy agent. Moreover, the secret proxy key is not known to the original signer as a means of protection for the proxy agent and to prevent a proxy signer from denying a signature it issued. The proposed scheme achieves all the security requirements that were discussed in Section 3. A comparative study with related schemes in literature revealed the superiority of the original proposed scheme as well as its elliptic-curve based variant from the computational viewpoint.

REFERENCES

[1] M. Mambo, K. Usuda, E. Okamoto, "*Proxy Signatures: Delegation of the Power to Sign Messages*," IEICE Trans. on Fundamentals, E79-A (9), pp.1338–1354, 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] Fagen Li, M.K. Khan, "*A Survey of Identity-Based Signcryption*," *IETE Technical Review*, Vol. 28, No. 3, pp. 265-272, 2011.

[4] C. Gamage, J. Leiwo, Y. Zheng, "*An Efficient Scheme for Secure Message Transmission Using Proxy-Signcryption*," 22nd Australasian Computer Science Conference, Springer- Verlag, pp. 420–431, 1999.

[5] H. Lin , T. Wu and S. Huang " *An Efficient Strong Designated Verifier Proxy Signature Scheme for Electronic Commerce*" Journal Of Information Science And Engineering 28, 771-785 (2012)

[6] H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications* ,Springer, Berlin, 2002.

[7] http://www.certicom.com/index.php/index.php/52-the-elliptic-curve-discrete-logarithm-problem

[8] S. Pradhan and R. K. Mohapatra," *Proxy Blind Signature Based On ECDLP*", International Journal Of Engineering Science And Technology (IJEST) ISSN : 0975-5462 Vol. 3 No. 3 March 2011.

[9] G. Swapna , P.V.S.S.N. Gopal , T. Gowri and P. Vasudeva Reddy" *An Efficient ID-Based Proxy Signcryption Scheme*", International Journal of Information & Network Security (IJINS) Vol.1, No.3, August 2012, pp. 200~206 ISSN: 2089-3299

[10] H. M. Elkamchouchi, Y. Abouelseoud and W. S. Shouaib, "*A new proxy Signcryption scheme using warrants*", volume 1, number 3, April 2011.

[11] R. Soram and M. Khomdram , "*Juxtaposition of RSA and Elliptic Curve Cryptosystem*" IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.9, September 2009

[12] M. Leslie ," *Elliptic Curve Cryptography*", Advanced Combinatorics June 5, 2006

[13] Y. Chung, Z. Y. Wu, F. Lai and T. Chen, " *Anonymous Signcryption in Ring Signature Scheme over Elliptic Curve Cryptosystem*", In proceeding of: Proceedings of the 2006 Joint Conference on Information Sciences, JCIS 2006, Kaohsiung, Taiwan, ROC, October 8-11, 2006

[14] http://www.ietf.org/rfc/rfc5639.txt