A Novel Proxy Signcryption Scheme and its Elliptic Curve Variant

Roayat Ismail Abdelfatah
Faculty of Engineering
Tanta University
Egypt

ABSTRACT

Proxy signcryption scheme allows an original signer to delegate his signing power to a proxy such that the latter can signcrypt a message on behalf of the former. In this paper, a new proxy signcryption scheme is proposed based on Discrete Logarithm Problem (DLP) and Diffie-Hellman Problem (DHP) with a reduced computational cost compared to other schemes in literature. The proposed scheme achieves public ciphertext authentication as the signcrypted message before being accepted, the receiver first verifies the signature. This property is very useful as the receiver can filter some incorrect ciphertext before decrypting it which achieves more efficient unsigncryption. Also, a variant of the main scheme that works over elliptic curves will be considered, since it has proven to provide better security with shorter keys and hence less storage requirements which makes it more suitable for resource constrained devices such as pagers and mobile phones. Numerical examples have been given with Mathematica to emphasize the ease of its practical use.

Keywords

Proxy signcryption, Discrete Logarithm Problem (DLP), Elliptic Curve Cryptography.

1. INTRODUCTION

The public key cryptography was firstly proposed by Diffie and Hellman [1]. Since then, public key encryption and digital signature are important tools to communicate with each other in a secure and authenticated way over open and insecure channels. The proxy signature is a cryptographic primitive that was first introduced by Mambo et al. [2] in 1996. 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. Based on the delegation type, there are three types of proxy signature [3-5]: full delegation, partial delegation, and delegation by warrant. In full delegation schemes, a proxy signer is given the same secret key *SK* that the original signer has, so that both create the same signatures Obviously, when the proxy signer deliberately signs a document unfavorable to the original signer, this misbehavior is not detected because the signature created by the proxy signer is undistinguished from the signatures created by the principal signer. In partial delegation schemes, a new secret key SK^* is given securely to the proxy signer. The proxy signature is checked by the modified equation and not by the original equation. This implies that a signature created by the proxy signer is distinguishable from a signature created by the original signer. In such delegation schemes, only the public key of the original signer is required for the signature verification. Partial delegation is further classified as proxy-unprotected and proxy-protected according to protection of proxy signer. In a proxy-protected scheme, only the proxy signer can generate valid proxy signatures, while in a proxy-unprotected scheme, either the original signer or the proxy signer can produce proxy signatures because both have knowledge on the proxy private key. In many applications, proxy-protected signature schemes are required to avoid the potential dispute between the original signer and the proxy signer. Kim et. al. [6] gave a new type of delegation called partial delegation with warrant, which certifies that the proxy is exactly the signer to be entrusted. This type of delegation is more secure than the full delegation as the created proxy signatures are distinguishable from ordinary signatures. Confidentiality, integrity, non-repudiation, and authentication are the important requirement for many cryptographic applications. Traditional approach to achieve these requirements is to sign a message and then encrypt the signature. In 1997, Zheng [7] proposed a cryptographic primitive, called signcryption, to achieve the combined functionalities of digital signatures and encryption in an efficient manner. Since then, some research works on signcryption have been carried out [8-17]. In 1999, Gamage et al. [18] extended the proxy signature and introduced a proxy signcryption scheme by combining proxy signature and encryption technology. It allows an entity to delegate his authority of signcryption to a trusted agent. Gamage's scheme is based on Discrete Logarithm Problem (DLP) in the traditional PKI based setting. The proxy signcryption scheme is a useful cryptographic tool. Let us consider a scenario that a boss in a company can delegate his or her capability of signing a message to an entity, when absence of time. If the message involves some sensitive information, the traditional proxy signature cannot satisfy this requirement. However, a proxy signcryption scheme just solves the problem. Many research works on proxy signcryption have been introduced [19-21].

In this paper, a new efficient proxy signcryption scheme based on delegation of signing rights using warrants is proposed. The security of the proposed scheme is based on the hardness of the DLP. The security aspects of the proposed scheme are discussed showing that proposed scheme satisfies several desirable requirements with a better efficiency compared to other schemes in literature. Moreover, a more efficient variant of the proposed proxy signcryption scheme based on elliptic curve DLP (ECDLP) and ECDHP is presented along with its related analysis.
The organization of the reset of the paper is as follows. Section 2 discusses the computationally hard problems; both the DLP, the DHP and the related ECDLP and ECDHP. Section 3 introduces the structure and security properties of the proposed scheme; section 4 discusses the proposed scheme together with its proof of correctness, security analysis and performance analysis; section 5 introduces a

variant of the proposed proxy signcryption scheme based on the ECDLP and ECDHP. A long with its related analysis. Finally, a numerical example with Mathematica10 program is given in section 6 to demonstrate the ease of implementation of the proposed scheme.

2. COMPUTATIONALLY HARD PROBLEMS

2.1 Discrete Logarithm Problem (DLP)

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 [22,23].

2.2 Discrete Logarithm (DL) Assumption

A probabilistic polynomial-time algorithm B is said to $(t, \mathcal{E})$ break the DLP if given an instance (y, p, q, g) where $y = g^{x} \bmod p$ for some $x \in Z_q$, B can derive with probability $\mathcal{E}$ 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. The $(t, \mathcal{E})$ DL assumption holds if there is no probabilistic polynomial-time adversary that can $(t, \mathcal{E})$ break the DLP [22, 23].

2.3 Elliptic Curve Discrete Logarithm Problem (ECDLP)

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 [22, 24].

2.4 Diffie-Hellman Problem (DHP)

Let p and q be two large primes satisfying q/p - 1, and g a generator of order q over $GF(p)$. The Diffie-Hellman Problem (DHP) is, given an element $\alpha^{a} \bmod p$ and $\alpha^{b} \bmod p$ and, find $\alpha^{ab} \bmod p$.

2.5: The Elliptic Curve Diffie - Hellman Problem (ECDHP)

The elliptic curve Diffie-Hellman problem is: given three points P, R, Q: $R = xP$ and $Q = yP$ in the group Z_p, find xyP.

3. STRUCTURE AND SECURITY PROPERTIES

The proxy signcryption scheme can be viewed as the combination of a general proxy signature and a signcryption scheme. A proxy signcryption scheme is a cryptographic primitive involving three entities: an original signer/sender (O), a proxy signer (P), and a verifier/receiver (R). Each entity has a secret and a corresponding public key.

3.1 Structure

A proxy signcryption scheme consists of the following algorithms:

1. Setup: This algorithm generates the system parameters.

2. Proxy Designation Protocol: This is an interactive protocol between the two parties (the original signer and his proxy agent. It takes the system parameters and private key of the original signer and outputs a corresponding proxy signing which the proxy agent can use to produce proxy signatures on behalf of the original signer.

3. Proxy Signcryption: It is usually a randomized algorithm. It takes as input the system parameters, the private key of the proxy signer, a message m, the public key of the intended receiver, and outputs a proxy signcrypted text.

4. Proxy Unsigncryption and Verification: It is a deterministic algorithm that takes as inputs the system parameters, the signcrypted text, the private key of the receiver and the public keys of the original and proxy signers. It validates the alleged signature of the proxy signer as well as decrypts the contents of the ciphertext part of the signcrypted text.

3.2 Security Properties

The scheme should achieve all the following proxy signcryption scheme properties:

1. Correctness: A properly formed signcrypted ciphertext by the signcryption algorithm must be accepted by the unsigncryption algorithm.

2. Confidentiality: Without the knowledge of the sender or the designated receiver's private key, it should be infeasible for an adaptive attacker to gain any partial information on the contents of the signcrypted ciphertext.

3. Distinguishability: The proxy signature must be distinguishable from the normal signature.

4. Unforgeability: A designated signer, called proxy signer, can create a valid proxy signature for the original signer. But the original signer and third parties who are not designated as a proxy signer cannot create a valid proxy signature

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

6. Verifiability: Validity of a proxy signature as well as the original signer's delegation on signature signing on a message can be verified using public parameters.

7. Non-repudiation of proxy signing: It is difficult for a proxy signer to falsely deny having signed its proxy signatures.

8. Non-repudiation of signature delegation: It is difficult for an original signer to falsely deny that it has delegated the signing power to a proxy signer.

9. Prevention of proxy key misuse: It should be confident that proxy key pair should be used only for creating proxy signature, which conforms to delegation information. In case of any misuse of proxy key pair, the responsibility of proxy signer should be determined explicitly.

4. THE PROPOSED SCHEME

First, we assume that all the parties: the original signcrypter, the proxy signcrypter, and the intended recipient are members of some common public key infrastructure.

The original signer key pair is (x_i, y_i), the proxy signer key pair is (x_p, y_p) , and the recipient key pair is (x_r, y_r).

4.1 The Proposed Scheme Construction

4.1.1. Setup: On input a security parameter k, the setup algorithm generates the two primes p, q such that $2^{k-1} \leq p \leq 2^{k}$ and q divides p -1. Moreover, it outputs an element $g \in$

$Z_p{}^*$ of order q and a hash function H:{0,1}$^*\rightarrow Z_q$. Furthermore, a secure symmetric key encryption algorithm, (*E*, *D*), must be agreed upon. The system parameters = (*p*, *q*, g, *H*) are then published.

4.1.2 Key Generation: The original user U_i chooses his private key x_i Є Z_q and computes the public key:

$y_i \equiv$ g ^{xi} mod *p*. The proxy U_p chooses his private key x_pЄ Z_q and compute the public key as $y_p \equiv$ g ^{xp} mod *p*. The recipient U_r chooses his private key x_r Є Z_q and compute the public key as $y_r \equiv$ g ^{xr} mod *p*. Both the proxy U_p and recipient U_r use Diffie-Hellman protocol [only one time (offline) before the transmission of any message] to exchange a shared secret key y_{sh} as follows:

The proxy *Up* computes: $y_{sh} \equiv (y_r)^{xp}$ mod $p \equiv$ g $^{xr\ xp}$ mod *p*

The recipient U_r computes: $y_{sh} \equiv (y_p)^{xr}$mod $p \equiv$ g $^{xr\ xp}$ mod *p*

4.1.3 Proxy Designation Protocol

The original user U_i delegates his signing power to a proxy signcrypter U_p as follows: U_i first choose a random integer *d* Є Z_q and compute $t \equiv$ (g ^{d} mod *p*) mod *q* and

$\sigma \equiv \left(\frac{d}{x_i} - H(m_W, t)\right)$ *mod q*, where m_W is a warrant consisting of the identifiers of the original and 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:

$(y_i{}^{\sigma + H(m_W, t)}$ mod *p*) mod *q* and performs check its validity as follows: $t \equiv (y_i{}^{\sigma + H(m_W, t)}$ mod *p*) mod *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:

$$y_i{}^{\sigma+H(m_w,t)} mod\ p \equiv g^{x_i\left(\frac{d}{x_i}-H(m_w,t)+H(m_w,t)\right)} mod\ p$$
$$\equiv g^d mod\ p \equiv t$$

After the proxy authenticates the original signer, the proxy computes the secret key as follows: $x_{pr} \equiv (\sigma + x_p)\ mod\ q$ as his proxy signature, secret key. Then he computes and publishes the corresponding proxy public key:

$$y_{pr} = g^{x_{pr}}\ mod\ p$$

$$Note\ that: y_{pr} = g^{x_{pr}}\ mod\ p = g^{(\sigma+x_p) mod\ q} mod\ p$$
$$= g^{\left(\frac{d}{x_i}-H(m_w,t)+x_p\right) mod\ q} mod\ p$$

4.1.4. Proxy Signcryption

The proxy will do the following steps to sign and encrypt the message *m*. The proxy chooses a random number *w* Є Z_q and computes:

1. $k \equiv H$ ((g^W mod *p*) mod *q*)
2. $z \equiv H$ ((y_{sh} . *k*) mod *p* mod *q*)
3. $C = E_Z(m)$
4. $V = H(C, k)$
5. $S \equiv \left(\frac{w}{x_{pr}} - V\right)\ mod\ q$

The proxy sends δ = (*C*, *V*, *S*) to the receiver.

4.1.5. Proxy Unsigncryption and Verification

Upon receiving (*C*, *V*, *S*), the receiver decrypts the message and checks the signature validity as follows:

1. Recover the key *k* by computing:
   $k` = H$ (($y_{pr}{}^{(S+V)}$ mod *p*) mod *q*).
2. Verify that the received signature is valid by computing:
   $V` = H(C, k`)$ and accepts if: $V`= V$
3. Compute $z` \equiv H$ (($y_{sh}$ . *k`*) mod *p* mod *q*), where y_{sh} is the shared secret key between the proxy and the receiver.
4. Decrypt the ciphertext $m = D_{z`}(C)$.

The signature verification equation involves the ciphertext instead of the plaintext message. Moreover, the equation for recovering *k* does not involve the private key of the recipient or any private information. Consequently, the signature part of the signcrypted text can be verified by any third party.

The proposed scheme achieves public ciphertext authentication: the signcrypted message δ = (*C*, *V*, *S*) before being accepted, the receiver first verifies the signature. This property is very useful as the receiver can filter some incorrect ciphertext before decrypt it which achieves more efficient unsigncryption.

4.2 Proof of Correctness

The following equations demonstrate the correctness of the proposed scheme

$k` = H$ (($y_{pr}{}^{(S+V)}$ mod *p*) mod *q*) =*H*

$$((y_{pr}{}^{\left(\frac{w}{x_{pr}}-V\right)+V})mod\ p)mod\ q) =$$
$$H(g^{x_{pr}\left(\frac{w}{x_{pr}}\right)}mod\ p)mod\ q) = H\big((g^w mod\ p)mod\ q\big) = k$$

4.3 Security Analysis

1. Confidentiality: The only way to decrypt *C* and obtain the message *m* is to have the shared-secret key y_{sh}. But it is difficult to obtain this key from the public keys y_p and y_r due to the intractability of DHP over the finite field $Z_p{}^*$. For a passive adversary, the information available is only (*C*, *V*, *S*). From this data, he can only obtain $k = H$ (($y_{pr}{}^{(S+V)}$ mod *p*) mod *q*) =$H\big((g^w mod\ p)mod\ q\big)$

but he cannot guess the corresponding *m*. Also, it is difficult to obtain the secret value *w* from $g^w mod\ p$ due to the intractability of the DLP. If an intruder intends to reveal the secret parameters x_{pr} and *w* from $S \equiv \left(\frac{w}{x_{pr}} - V\right)\ mod\ q$. This will be difficult because there are two unknown variables

(x_{pr} , *w*) in one equation. Then even if x_{pr} is revealed, a person cannot compute y_{sh} as we mentioned previously and so cannot decrypt *C*. Therefore, the proposed scheme provides forward secrecy property with respect to the proxy signer.

2. Verifiability: During the unsigncryption, the receiver can be convinced that the proxy sender has the original signer's signature on the warrant by using y_{pr} which includes the private key of both the original signer and the proxy signer besides the random value d and the warrant mw which contains the identity information of the original sender and the limit of delegated signcrypting capacity. Therefore, the receiver can be convinced of the original sender's agreement on the signcrypted message. Thus, the scheme satisfies the verifiability requirement.

3. Unforgeability: According to the Eqn.

$x_{pr} \equiv (\sigma + x_p)\ mod\ q$ the proxy sender includes his private key x_p in the computation of his proxy signcryption key, so no one can forge this key. For anyone who want to forge the proxy signature $S \equiv \left(\frac{w}{x_{pr}} - V\right) mod\ q$, he must know w and x_{pr} which both protected by DLP.

Thus, no one except the proxy sender can create a valid proxy signcryption. So, the proposed scheme achieves unforgeability.

4. Distinguishability: This is obvious, because there is a warrant m_w in a valid proxy signature, at the same time the proxy signature public key y_{pr} which includes the private key of both the original signer and the proxy signer**.** So, the proxy signature is easy to be distinguishable from the normal signature.

5. Prevention of Misuse: Using the warrant m_w in our scheme had determined the limit of the delegated signing capacity. So, the proxy signer cannot sign some messages that have not been authorized by the original signer and this prevent abuse of the proxy key.

6. Nonrepudiation: Since the original signer is the only one who can compute σ as his private key x_i is included in it, he cannot repudiate the signing capability delegation to the proxy signer and this can be verified publicly as we indicated previously. So, the proposed scheme provides non-repudiation of signature delegation. The original signer doesn't obtain the proxy signer's private x_p and the proxy signer doesn't obtain the original signer's private key x_i, neither the original signer nor the proxy signer can sign in place of the other party.

7. Public Verification: If the proxy signer denies the signature of the message m, the receiver can prove the dishonesty of the signer by passing (C, V, S) to the any third party who can check the signature validity by computing $k` = H\ ((y_{pr}{}^{(S+V)}\ \text{mod}\ p)\ \text{mod}\ q)$ then verify that the received signature is valid by computing $V` = H\ (C, k`)$ and accepts if:

$V` = V$. So, in the proposed scheme anyone can verify the signature without the need to any secret values. Thus, it achieves public verifiability.

4.4 Performance Analysis

The security of the proposed scheme is based on the DLP and DHP. Its performance is better than another DLP-based scheme introduced in [20]. This comparison is provided in Table 2 and Figure 1. Table 1 shows the abbreviations that will be used in the comparison.

Table 1. Time abbreviations

Symbol	Operation
T_e	Time for performing a modular exponentiation computation.
T_{mult}	Time required for executing a modular multiplication in a finite field.
T_h	Time required for executing one-way hash function.
T_{encr}	Time required for executing an encryption operation.
T_{decr}	Time required for executing a decryption operation.

Table 2. The comparison of the proposed proxy signcryption with the scheme in [20]

Phase	Scheme in [20]	Ours
Proxy Designation	$3T_e + 2T_h + 3T_{mult}$	$3T_e + 2T_h + 1T_{mult}$
Signcryption	$1T_e + 1T_h + 1T_{mult} + 1T_{encr}$	$1T_e + 3T_h + 2T_{mult} + 1T_{encr}$
Unsigncryption	$3T_e + 2T_h + 3T_{mult} + 1T_{decr}$	$1T_e + 2T_h + 1T_{decr} + 1T_{mult}$
Total No. of Operations	$7T_e + 5T_h + 7T_{mult} + 1T_{encr} + 1T_{decr} = 21$	$5T_e + 7T_h + 4T_{mult} + 1T_{encr} + 1T_{decr} = 18$

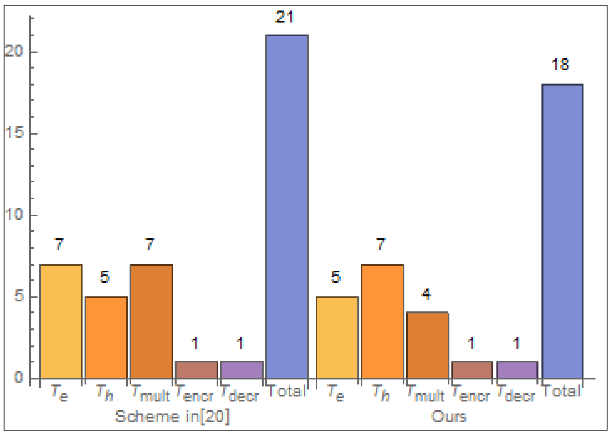

Fig 1: The comparison of the proposed proxy signcryption with the scheme in [20]

To simplify the estimation of computational costs, we consider only the major operation. For example, the computational cost of modular multiplication, hash function, symmetric key encryption and decryption and hash function is ignored as compared with the expensive costs of modular exponentiation. So, the proposed scheme needs 5 modular exponentiations compared to 7 modular exponentiations of the scheme in [20].

Thus, the computational cost saving $\frac{7-5}{7} = 28.5\%$.

Another advantage of the proposed scheme over the scheme in [20] is that the proposed scheme achieves public verifiability while the scheme in [20] does not. The equation for recovering k does not involve the private key of the recipient or any private information. Consequently, the signature part of the signcrypted text can be verified by any third party.

While in the scheme in [20], the signature verification requires the knowledge of the receiver's private key, so it can't provide public verifiability.

5. THE ELLIPTIC CURVE VARIANT OF THE PROPOSED SCHEME

Since 1985 that elliptic curves applied independently by Miller [25] and Koblitz [26] to introduce a new public key cryptography, many researchers tried to employ it on different data types and improve the efficiency by proposing various techniques. In fact, the most attractive advantage of ECC that motivated cryptographers to use it was the greater security and more computationally efficient performance with equivalent key size in comparison with another public key as the RSA, ElGamal, ect. Shorter keys result in less storage requirements. The ECDLP 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 the 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_r, Y_r) is the recipient key pair. The EC variant of the proposed proxy signcryption scheme is as follows.

5.1 The Proposed Scheme Construction

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.

5.1.2 Key Generation

The original user U_i chooses his private key $x_i \in Z_q^*$ and computes the public key $Y_i = x_i .G$. 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_r chooses his private key $x_r \in Z_q^*$ and computes the public *key* as $Y_r = x_r .G$. Both the proxy U_p and recipient U_r use Diffie-Hellman protocol [only one time - offline-before the transmission of any message] to exchange a shared secret key Y_{sh} as follows:

The proxy U_p computes: $Y_{sh} = x_p . Y_r = x_p . x_r . G$

The recipient U_r computes: $Y_{sh} = x_r . Y_p = x_r . x_p .G$

5.1.3 Proxy Designation Protocol

The original user U_i delegates his signing power to a proxy signcrypter U_p as follows: U_i first chooses a random integer $d \in Z_q^*$ and computes

$$T = d . G = (\alpha , \beta) = (T_x, T_y)$$

$$\sigma \equiv \left(\frac{d}{x_i} - H(m_W, \alpha)\right) mod\ q$$

The original user U_i sends (σ, m_W , α) to U_p. Upon receiving (σ, m_W , α), U_p computes:

$Y_{i\ .}(\sigma + H(m_w, \alpha)$ and performs check its validity as follows:

$(T_x`, T_y`) = Y_{i\ .} (\sigma + H(m_w, \alpha))$. Checks if $T_x` = \alpha$ If is not equal to the right-hand side, the proxy requests a new (σ, m_W, α) to be sent again. The verification of the above equation proceeds as follows:

$$Y_{i\ .}(\sigma + H(m_w, \alpha) = x_i .G.(\frac{d}{x_i} - H(m_W, \alpha) + H(m_W, \alpha))$$
$$= G . d = T$$

After the proxy authenticates the original signer, the proxy computes the secret key as follows: $x_{pr} \equiv (\sigma + x_p) mod\ q$ as his proxy signature, secret key. Then he computes and publishes the corresponding proxy public key:

$$Y_{pr} = x_{pr}. G = (\sigma + x_p). G = \left(\frac{d}{x_i} - H(m_W, \alpha) + x_p\right). G$$

5.1.4 Proxy Signcryption

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

1. $K = w . G = (x_1, y_1)$
2. $Z = Y_{sh} + K = (x_2, y_2)$
3. $C = E_{x_2}(m)$
4. $V = H(C, x_1)$
5. $S \equiv \left(\frac{w}{x_{pr}} - V\right) mod\ q$

The proxy sends $\delta = (C, V, S)$ to the receiver.

5.1.5 Proxy Unsigncryption and Verification

The receiver decrypts the message and checks the signature validity as follows:

1. Recover the key K by computing:
   $K` = Y_{pr\ .} (S + V) = (x_1`, y_1`)$
2. Verify that the received signature is valid by computing:
   $V` = H(C, x_1`)$ and accepts if: $V` = V$
3. Compute $Z = Y_{sh} + K` = (x_2`, y_2`)$
   where Y_{sh} is the shared secret key between the proxy and the receiver.
4. Decrypt the ciphertext $m = D\ x_{2`} (C)$.

5.2 Proof of Correctness

The following equation demonstrate the correctness of the proposed scheme:

$$K` = Y_{pr\ .} (S + V) = x_{pr} . G\ (\frac{w}{x_{pr}} - V + V) = G.w = (x_1, y_1)$$

5.3 Performance Analysis

The performance of the proposed proxy signcryption scheme based on the ECDLP is analyzed and compared to the scheme in [20]. It is found that the proposed scheme involves fewer computations than the scheme in [20] Table 3 defines the notation that is used in comparison. Table 4 and Figure 2 shows the comparison of the proposed scheme based on the ECDLP to that of [20].

Table 3. Comparison notation

Symbol	Operation
$T_{EC\text{-}mult}$	Time required for executing multiplication operation on elliptic curve *E*
$T_{EC\text{-}add}$	Time required for executing addition operation on elliptic curve *E.*
T_{mult}	Time required for executing a modular multiplication in a finite field.
T_h	Time required for executing one-way hash function.
$T_{encr}/T_{decr.}$	Time required for executing an/a encryption/decryption operation.

Table 4. The proposed signcryption scheme compared with the scheme in [20]

Phase	**The Scheme in [20]**	**Ours**
Proxy Designation	$3T_{EC\text{-}mult}+1T_{EC\text{-}add}+2T_h+2T_{mult}$	$3T_{ECmult}+2T_h+2T_{mult}$
Signcryption	$1T_{EC\text{-}mult}+1T_h+1T_{mult}+1T_{encr}$	$1T_{EC\text{-}mult} + 1T_{EC\text{-}add}+1T_h+1T_{mult}+1T_{encr}$
Unsigncryption	$3T_{EC\text{-}mult} +3T_{EC\text{-}add}+1T_{ht}+1T_{decr}$	$1T_{EC\text{-}mult} +1T_{EC\text{-}add}+1T_h+1T_{decr}$
Total No. of Operations	$7T_{EC\text{-}mult} +4T_{EC\text{-}add}+3T_h+3T_{mult}+1T_{encr}+1T_{decr} =19$	$5T_{EC\text{-}mult} +2T_{EC\text{-}add}+4T_h+3T_{mult}+1T_{encr}+1T_{decr} =16$

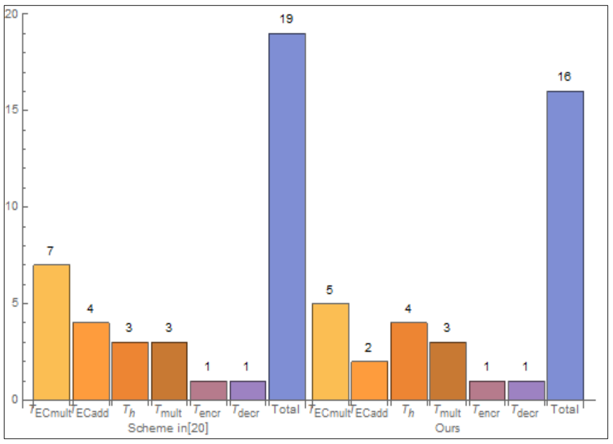

Fig 2: The comparison of the proposed EC variant with the scheme in [20]

To simplify the estimation of computational costs, we consider only the major operation which is the multiplication operation on elliptic curve *E*. The proposed scheme needs 5 modular exponentiations compared to 7 modular exponentiations of the scheme in [20].

Thus, the computational cost saving $\frac{7-5}{7} = 28.5\%$.

6. NUMERICAL EXAMPLE

Here is a numerical example as a proof-of-concept, which has been implemented using Mathematica 10.2 program. In this example, the parameters used are among the 256-bit recommended domain parameters for elliptic curves.

- *p* is the prime specifying the base field.
- *a* and *b* are coefficients of the equation: $y^2 = (x^3 + ax +b)$ mod *p* defining the elliptic curve.

273245448356212950287978634733803,100736746495
537631506687310442098478662301202803857003310
38661251300441884764}

Y_{shared} = (Shared secret key between the proxy signer and the receiver)
{22668527831036485363505488892878201594283203
170481426910562157676151201977476, 50411069104
219184216468076044458349021210461405232174612
753542058157281640929}.

6.3 Proxy Key Generation

d
$= 20666999796610830832276857422478850305089400100800938081655008090327617096.$

T=
{11603919161609063432713514605716968075070369

- $G = (x, y)$ is the base point, i.e. a point in *E* of prime order, with *x* and *y* being its *x*-and *y*-coordinate, respectively.
- *q* is the prime order of the group generated by *G*.

6.1 Setup

p =
76884956397045344220809746629001649093037950200943
055203735601445031516197751

a=
56698187605326110043627228396178346077120614539475
214109386828188763884139993

b=
17577232497321838841075697789794520262950426058923
084567046852300633325438902

$G = (x , y)$ =
(63243729749562333355292243550312970334778175571054
726587095381623627144114786,
38218615093753523893122277964030810387585405539772
602581557831887485717997975)

q =
76884956397045344220809746629001649092737531784414
529538755519063063536359079.

6.2 Key Generation

x_i = (original signer secret key) =
866595994033399539216311608927453897378463889
937410554442261607268691789707

x_p = (proxy signer secret key) =
463453778675747388075947166319093770187249652
4845891485128629547185876098

x_r = (receiver secret key)
123678921189508511736974980106190281333165457
11097957018197085350687390.

m_W
$= 3456257372414217562668310384556007687103260450141912601460306622985447687$

m (the message) is the word "cryptography"

Y_i = (original user public key)
{11904646479248176634565247724299112168189478
474674455710539116488162623935825, 44681411079
611309987496826974264395221829781697036802065
1438874124193537602}

Y_p = (proxy public key)
{10320723656485208543977832058624414770291794
164779781912467197746386375905257,756466550078
490487658239638330816901411621124446655969874
08764108769259816312}

Y_r= (Receiver public key) =

515679940565694322503179044255255, 65885113768
109763301671143780299605591757178714021240525
402700944123985384042}

σ
$= 53627026819731548112194644046903243715109886530587663184429883548840741013131.$

$Y_{i\ .}\ (\sigma + H\ (m_w\,, \alpha))$ = (verify the original user)
{11603919161609063432713514605716968075070369
515679940565694322503179044255255, 65885113768
109763301671143780299605591757178714021240525
402700944123985384042} $= T$

x_{pr}
$= 5363166135751830558607540351856643465281175902711250907591501217838792688$

Y_{pr}=
{13041773698909967771619958039067633822074615 920127266461559756095977587635876, 71415885185 446878481495340818835758425154559969902361955 609906145144359426222}

6.4 Signcryption Generation

w=
788900075432367789997765544324567887766654909
876;
$K = w\ .\ G = (x_1, y_1)$
{64162821985873772948677605470576070657495075 682514546267932976986529785182744, 50167066442 448277394769285182928629584394033706869005929 373169304387007097917} .
$Z = Y_{sh} + K = (x_2\ , y_2)$
868313498169102583121830943634542722517782788 52995973178495134662680987160220, 100578135546 667461611237361227386978605604495112101180542 126711362544288738846}.
$C = E_{x_2}(m) =$
868313498169102583121830943634542722517782788
53006125466284158168649280628453.
$V = H\ (C, x_1)$
= 286660571245873686003929741075727652702.
$$S \equiv \left(\frac{w}{x_{pr}} - V\right) mod\ q =$$
683197827462845915709681071797691902394651462
78528156492117057839563175962207.

6.5 Unsigncryption and Verification

$Y_{pr\ .}\ (S + V) = (x_{1`}, y_{1`})$=
{64162821985873772948677605470576070657495075 682514546267932976986529785182744, 50167066442 448277394769285182928629584394033706869005929 373169304387007097917}.

$V` = H\ (C, x_1`) =$
286660571245873686003929741075727652702 = V

The receiver accepts the received ciphertext.

$Z = Y_{sh} + K` = (x_{2`}, y_{2`}) =$
868313498169102583121830943634542722517782788 52995973178495134662680987160220, 100578135546 667461611237361227386978605604495112101180542 126711362544288738846}.

$m = D\ x_{2`}\ (C) = cryptography$

7. CONCLUSIONS

As people in modern societies are busier than any human era and computer network has profound impact on how people work and live through fast and convenient information exchange, people need more help from each other to accomplish more work via network connections in limited period of time. Therefore, privilege delegation mechanism has become a necessary service in modern enterprises and organizations. Proxy signcryption scheme provides a secure privilege delegation mechanism for a person to delegate his privilege to his proxy agent to accomplish things. This paper introduces a proxy signcryption scheme in which the original signer delegates his signing rights to a proxy agent. This scheme has many applications such as in e-cash systems. The security properties of the proposed scheme are investigated revealing that it meets all security requirements. The use of a warrant facilitates identifying both the original as well as proxy agents. Additionally, warrants are used to specify the signing capacity of the proxy agent to prevent misuse of the delegation. Moreover, the proxy signature is distinguishable from the original signer to protect a malicious proxy agent. The proposed scheme as well as its elliptic-curve based variant is compared with the scheme in [20] and it is found that the proposed scheme reduces the computational burden so that it is more efficient to support the class of applications targeted by it such as e-commerce using mobile computing and communication devices that require large number of individual short messages for the completion of a transaction. These include banking services stock, trading international roaming information for GSM etc.

8. REFERENCES

[1] Diffie W, Heliman M. New direction in cryptography. IEEE Transactions on Information Theory 1976; 22 (6): 644-654.

[2] Mambo M, Usuda K, Okamoto E. Proxy signature: delegation of the power to sign messages. IEICE Transaction on Fundamentals 1996; E79-A (9): 1338-1353.

[3] Kim S, Park S, Won D. Proxy signatures, revisited. In Proceedings of Information and Communications Security. (ICICS' 97), Han Y, Okamoto T, Qing S (eds), LNCS 1334. Springer- Verlag:Beijing, China, 1997; 223-232.

[4] Lee B, Kim H, Kim K. Secure mobile agent using strong non-designated proxy signature. In Proceedings of Information Security and Privacy (ACISP'01), Varadharajan V, Mu Y (eds), LNCS 2119. Springer Verlag: Sydney, Australia, 2001; 474-486.

[5] Lee B, Kim H, Kim K. Strong proxy signature and its applications, In Proceedings of the Symposium on Cryptography and Information Security (SCIS'01), Oiso, Japan,2001;603-608.

[6] Kim S, Park S, and Won D. Proxy signatures, revisited. In Proceedings of ICICS 97, LNCS 1334, Springer-Verlag; 1997; 223-232.

[7] Zheng Y. Digital Signcryption or How to Achieve Cost (Signature & Encryption) << Cost (Signature) + Cost (Encryption)", Advances in Cryptology – CRYPTO' 97, volume 1294 of Lecture Notes in Computer Science, Springer-Verlag; 1997; 165-179.

[8] Lee JM, Mao W. Two birds one stone: signcryption using RSA. In Topics in Cryptology (CT-RSA'03), Joye M (ed), LNCS 2612. Springer-Verlag: San Francisco, CA, USA, 2003; 211–225.

[9] Malone-Lee J. Identity based signcryption. Available from: http://eprint.iacr.org/2002/098.pdf [Accessed on 30 May 2011].

[10] Libert B, Quisquator JJ. A new identity based signcryption scheme from pairings. In Proceedings of IEEE Information Theory Workshop (ITW'03). Elsevier: Paris, France, 2003; 155–158.

[11] Chow SSM, Yiu SM, Hui LCK, Chow KP. Efficient forward and provably secure ID based signcryption scheme with public verifiability and public ciphertext authenticity. In Procceedings of Information Security and Cryptology (ICISC'03), Lim JI, Lee DH (eds), LNCS 2971. Springer-Verlag: Seoul, Korea, 2004; 352–369.

[12] Boyen X. Multipurpose identity based signcryption: a Swiss army knife for identity based cryptography. In Advance in Cryptology (CRYPTO'03), Boneh D (ed), LNCS 2729. Springer-Verlag: Santa Barbara, California, USA, 2003; 383–399.

[13] Chen L, Malone-Lee J. Improved identity-based signcryption. In Public Key Cryptography (PKC'05), Vaudenay S (ed), LNCS 3386. Springer-Verlag: Les Diablerets, Switzerland, 2005; 362–379.

[14] Barreto PSLM, Libert B, McCullagh N, Quisquater JJ. Efficient and provably-secure identity based signatures and signcryption from bilinear maps. In Advance in Cryptology (ASIACRYPT'05), Roy BK (ed), LNCS 3788. Springer-Verlag: Chennai, India, 2005; 515–532.

[15] Yu Y, Yang B, Sun Y, Zhu S. Identity based signcryption scheme without random oracles. Computer Standards and Interfaces 2009; 31(1): 56–62.

[16] Jin Z, Wen Q, Du H. An improved semantically-secure identity-based signcryption scheme in the standard model. Computers and Electrical Engineering 2010; 36: 545–552.

[17] Liu Z, Hu Y, Zhang X, Ma H. Certificateless signcryption scheme in the standard model. Information Sciences 2010; 180: 452–464.

[18] Gamage G, Leiwo J, and Zheng Y. An efficient scheme for secure message transmission using proxy signcryption. Technical report 98-01, Monash University, 1998.

[19] Elkamchouchi H, Abouelseoud Y, and Shouaib W. A new proxy Signcryption scheme using warrants.International Journal of Intelligent Engineering Informatics, Vol.1, No. 3, April 2011.

[20] Elkamchouchi H, Abouelseoud Y, Abu Elkhair E. An efficient proxy signcryption scheme based on the discrete logarithm problem. International Journal of Technology and Computing (IJITMC). Vol. 1 No.2 May 2013.

[21] Ming Y, Wang Y. Proxy signcryption scheme in the standard model. Security and Communication Networks 2015; 8; 1431-1446.

[22] Lin H, Wu T and Huang S. An Efficient Strong Designated Verifier Proxy Signature Scheme for Electronic Commerce. Journal of Information Science and Engineering 28;2012;771-785.

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

[24] The Elliptic Curve Discrete Logarithm Problem http://www.certicom.com/index.php/index.php/52-the-elliptic-curve-discrete-logarithm-problem.

[25] Miller M. Uses of elliptic curves in cryptography. Advances in Cryptography-Crypto '85.1986; 417-426.

[26] Koblitiz N. Elliptic curve cryptosystems. Mathematics of computation. Vol. 48; No. 177; 1987; 203-208.