Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 8 (10): 1919-1925, 2008
ISSN 1812-5654

A Protocol for Digital Signature Based on the Elliptic Curve Discrete Logarithm Problem

Morteza Nikooghadam, Mohammad Reza Bonyadi, Ehsan Malekian and Ali Zakerolhosseini
Faculty of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran

Abstract: Digital signature and cryptography algorithms based on the Elliptic Curve Discrete Logarithm Problem (ECDLP) have recently received significant attention by researchers due to their high performances. In this research, a novel protocol for digital signature based on the ECDLP has been presented which in comparison with the other protocols is shown to be more efficient. An acceptable security level of the proposed protocol similar to other protocols is also verified. The performance and the time complexity of the proposed protocol in comparison to previous protocols is analyzed and some advantages outlined.

Key words: Elliptic curves, public key cryptography, galois fields, message digest, hash function

INTRODUCTION

The term digital signature was referred to by Diffie and Hellman (1976). Their design intended to present an algorithm having properties identical to hand written signature (i.e., simple to generate and verify but difficult to forge). Nonetheless, the signature could be generated automatically by means of a digital machine to be utilized in digitized documents (Zakerolhosseini and Malekian, 2007).

The most significant difference between the hand written and digital signatures is the uniqueness of hand written signatures for all the documents. However, the digital signatures are dependent on the message and do change accordingly. In general, the mechanisms for implementation of a digital signature can be categorized into four groups as follows (Zakerolhosseini and Malekian, 2007):

- Digital signatures based on the message digest
- Symmetrical key digital signatures, established by a reliable center for signature confirmation
- Digital signatures based on the public key cryptosystems
- Signatures based on transforms that are independent from cryptosystems

The main focus of this research is to improve the first group of digital signatures. In this system, each document is set into a standard form and then a short digest consisting of few bytes is generated. The size of the document does not influence the size of the digest so the size of digest is constant for all the documents. The digest will intricately be affected only by changes in contents and locations of each bit in the document. The digest will be estimated in such a way that any alteration of the document will lead to great changes in its digest. After extracting the digest, the bit-string created will be encrypted by means of signer's private key and the result is appended to the message. In fact, a digital signature is a numerical string which should be extracted from the context of a document through a complex procedure and it will be attached to the document after being encrypted by the signer's private key (Zakerolhosseini and Malekian, 2007).

In order to verify, the receiver must decrypt the encrypted digest using the signer's public key and re-compute the digest of the received message. Then, the receiver compares the two results and if they are the same, the document will be accepted; otherwise the document is deemed to have changed. Figure 1 shows the general procedure (Zakerolhosseini and Malekian, 2007):

Estimation of the message digest: To estimate the digest, the algorithms must satisfy the following conditions (Zakerolhosseini and Malekian, 2007):

- When a message m is available, computing its digest e must be computationally fast
- If a digest is available, the major context of the document can not be reproduced (one-way procedure)
- In practice, make several messages having the same digest impossible

The public key cryptography: Many of the public key cryptography methods have been broken, but some of

Corresponding Author: Ali Zakerolhosseini, Faculty of Electrical and Computer Engineering, Shahid Beheshti University, Velenjak, Tehran, Iran Tel: +989121236539 Fax: +98-21-55404240

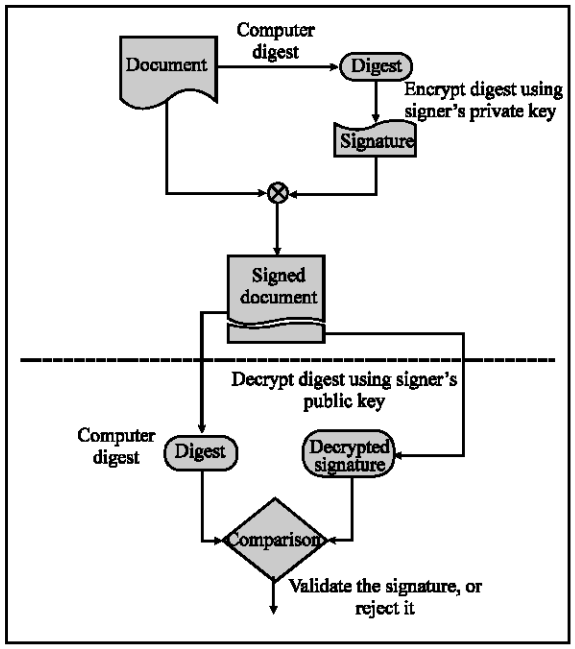

Fig. 1: The procedures for digital signature based on the message digest

them have remained tenacious (Vanstone, 1997). These methods are categorized fundamentally into three groups:

- The Integer Factorization Problem (IFP)
- The Discrete Logarithm Problem (DLP)
- The Elliptic Curve Discrete Logarithm Problem (ECDLP)

Increasing the key length in digital signature algorithms based on the ECDLP causes an exponential increase in the security which is the most important feature of the algorithms in this category. Thus, in order to satisfy the security requirements, the Elliptic Curve Cryptosystems (ECC) need a smaller key size compared to other algorithms (Vanstone, 1997). This feature of having a small key size simplifies the calculations significantly and also reduces the power dissipation (Caelli *et al*., 1999). Therefore, the algorithms based on the ECDLP are receiving much attention in applications such as communications, sensor networks and wireless devices. In recent years, the ECC has been widely established among international standards, for instance, ISO 11770-3, IEEE P1363, ANSI X9.62 and FIPS 186-2, etc. At the end of this research, the performances of the categories mentioned earlier are compared.

The elliptic curves over finite fields: Elliptic curves introduced by Miller (1985) and Koblitz (1987) play an important role in cryptography (Hwang and Liao, 2005; Tzeng and Hwang, 2004). Let GF(2^m) be a finite field of 2^m elements, where m is an integer. An elliptic curve over GF (2^m) is defined as Zheng and Imai (1998) and Johnson *et al.* (2001):

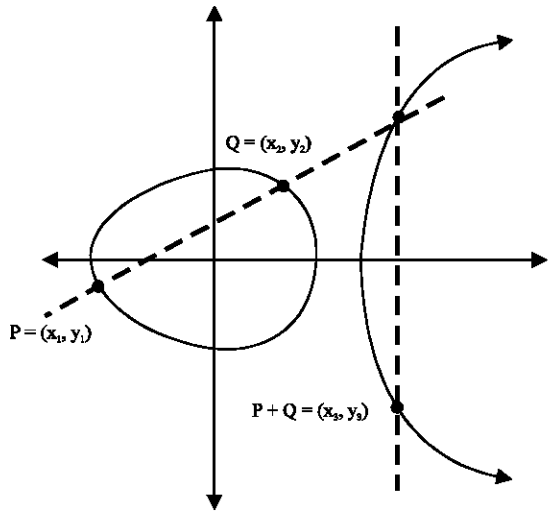

Fig. 2: Addition on elliptic curves

$$y^2 + xy = x^3 + ax^2 + b \quad \text{with } a, b \in GF(2^m),\ b \neq 0 \quad (1)$$

An elliptic curve over GF(2^m) consists of all points (x,y), where x, y ϵ GF(2^m) together with the point of infinite o, do satisfy (1) earlier. The addition of two points and doubling a point on an elliptic curve in a geometrical space, are shown in Fig. 2 and 3, respectively.

Considering an elliptic curve C on GF(2^m), the addition of points follows specific rules indicated below (Johnson *et al.*, 2001; Coombes, 1999; Yu and Chen, 2005):

- O + O = O.
- P + O = P for all values of P = (x, y) ϵ C. Namely, C has O as its identity element.
- P + Q = O for all values of P = (x, y) ϵ C and Q = (x, -x-y) ϵ C namely the inverse of (x, y) is simply (x, -x-y)

- **Adding two distinct points:**

For all $P = (x_1, y_1) \epsilon C$ and $Q = (x_2, y_2) \epsilon C$ with $x_1 \# x_2$, $P + Q = (x_3, y_3)$ is defined as:

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a \quad \text{Where} \quad \lambda = \frac{y_2 + y_1}{x_2 + x_1}$$
$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$$

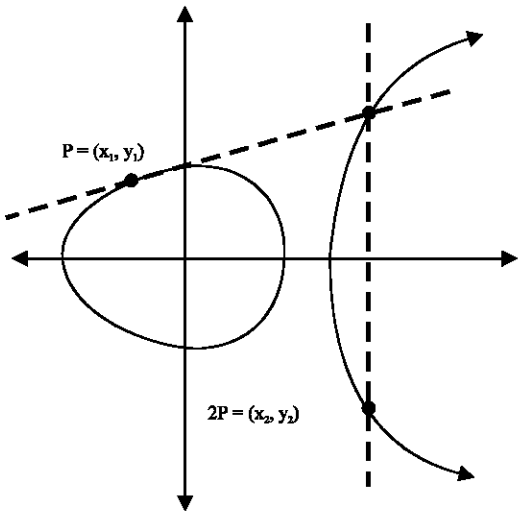

Fig. 3: Doubling a point

- **Doubling a point:**

For any $P = (x_1, y_1) \in C$ with $y_1 \neq 0$, $2P = (x_2, y_2)$ is defined as:

$$x_2 = \lambda^2 + \lambda + a$$
$$y_2 = \lambda(x_1 + x_3) + x_3 + y_1$$
$$\text{where } \lambda = x_1 + \frac{x_1}{y_1}$$

The scalar multiplication: The scalar multiplication is a fundamental operation in ECCs. The operation is simply an accumulation of a point P to itself for k times (k is an m bit long scalar) (Johnson *et al*., 2001):

$$Q = kP = \underbrace{P + P + ... + P}_{k}$$

The discrete logarithm problem on elliptic curves: Let P and Q be the two points on an elliptic curve with an order of n where n is a prime. The point Q = kP where k < n. Given the points P and Q, estimating the value of k is computationally infeasible (Smart, 1999).

Previous works: Here, some previous algorithms based on the ECDLP are briefly investigated. The weakness of these protocols is examined which is the motive behind designing a new protocol. Generally, there are four operational phases in digital signatures based on the ECDLP as listed below:

- The system initialization phase
- The key generation phase
- The signature generation phase
- The signature verification phase

In Table 1, a brief description of two well known algorithms is presented. In both these algorithms it is assumed that entity A wishes to sign a message m and selects a random integer d_A from the interval [1, n-1] as the private key and publishes $Q_A = d_A G$ as the public key.

In Elliptic Curve Digital Signature Algorithm (ECDSA) (Johnson *et al*., 2001), The problem is due to resetting "s" to zero in step 4 of signature generation phase, that leads to a jump to start of the phase and in turn, re-computing some costly operations like scalar multiplication, modular inversion and modular multiplication. This jump to start of the phase is due to s having a value of zero and in that case the s^{-1} in the verification phase can not be computed. Also as stated in EC El-Gamal Digital Signature Scheme (Rabah, 2005), resetting "s" to zero leads to incorrect result in the verification phase. Therefore, in the algorithms stated in Table 1 if s is set to zero, the signature generation phase must become recurring.

In order to correct this weakness and also to improve the performance of such algorithms, some approaches such as the method presented in Chung *et al.* (2007) and the proposed method, are presented here. A comparison of these methods is also presented here.

THE PROPOSED METHOD

Here, a new and improved protocol based on the ECDLP is proposed. Initially, the structure of the protocol is described and then, the effectiveness and the security of the protocol are investigated. This algorithm is divided into four steps similar to other digital signature protocols.

Step 1: System initialization phase

- A field size q = p which defines the underlying finite field Fq, where either q = p in case that p is an odd prime, or $q = 2^m$ that q is a prime power
- Specifying an appropriate elliptic curve by selecting two parameters a and b of elliptic curve equation E over Fq: $y^2 + xy = x^3 + ax^2 + b$.
- The base point $G = (_{Gx,Gy})$, is a finite point on elliptic curve having the largest order n
- The order n of the base point G, is a large prime number in E(Fq). N = #E(Fq) is divisible by n

Step 2: Key generation phase

Signer "A" generates the public and private keys, as follows:

- Select a random integer d from the interval [1, n-1] as the private key
- Compute Q = dG as the public key

Table 1: The signature generation and verification phases of two known algorithms

Algorithms	Signature generation phase	Signature verification phase
EC	Select a random integer k from the interval [1, n-1];	Compute $V_1 = sF$;
El-Gamal	Compute $F = kG = (x_1, y_1)$, $r = x_1 \bmod n$; if $r = 0$ then go to step 1;	Compute $V_2 = h(m)G + rQ_A$, where $r = x_1$.
(Rabah, 2005)	Compute $e = h(m)$, where h is a hash function ;	Accept if and only if $V_1 = V_2$.
	Compute $s = k^{-1}(e + d_A r) \bmod n$. If $s = 0$ then go to step 1, signature is (F, s)	
ECDSA	Select a random integer k from the interval [1, n-1];	Compute $w = s^{-1} \bmod n$;
(Johnson *et al*., 2001)	Compute $F = kG = (x_1, y_1)$, $r = x_1 \bmod n$; if $r = 0$ then go to step 1;	Compute $u_1 = h(m)w \bmod n$ and $u_2 = rw \bmod n$;
	Compute $e = h(m)$;	Compute $u_1 G + u_2 Q_A = (x_1, y_1)$;
	Compute $s = k^{-1}(e + d_A r) \bmod n$. If $s = 0$ then go to step 1, signature is (r, s)	Compute $v = x_1 \bmod n$;
		Accept the signature if and only if $v = r$.

Step 3: Signature generation phase

Signer "A" generates a signature for the message m, as follows:

- Select a random integer k from the interval [1, n-1], where k # d (d is the private key for signer "A")
- Compute $F = kG = (x_0, y_0)$ and $r = x_0 \bmod n$. If r = 0 then go to step 1
- Convert the message m into an integer e using the hash-function operation, e = Hash(m)
- Compute $s = (dre + k) \bmod n$
- (s, F) is the signature generated by signer "A" for the message m

Step 4: Signature verification phase

The verifier confirms the validity and authenticity of the signature m, as:

- Compute the digest of received message as $e^* = \text{Hash}(m^*)$,
- Compute $v = s^* G$
- Compute $u = e^* r^* Q + G^*$
- If v = u then validate the signature; otherwise reject

Note: It is assumed the message received for the verification is m* and the received signature is (s^*, F^*). The use of these new symbols indicates that message and signature may have been altered.

Proof: Assume the message and its signature have not been altered, i.e., $F^* = F$, $s^* = s$, $e^* = e$. Since an elliptic curve over $GF(2^m)$ forms an Abelian group (Lawrence, 2003) under an addition on points, then the consistency of the scheme will be evaluated by Eq. 2.

$$\begin{aligned} v = s^*G = sG = (dre + k)G = dreG + kG \\ = e^*r^*dG + (kG) = e^*r^*(dG) + (F^*) = u \end{aligned} \quad (2)$$

In this protocol, the modular inversion operation in signature generation and verification phases could be avoided. Also, compared to other schemes as shown in Table 1, when "s" in step 4 of the signature generation phase becomes zero, the verification phase can still be performed using a signature that it's "s" is equal to zero and thus the signature generation phase does not repeat.

Having $s = 0$ results in $sG = o$ (point of infinity) (Rabah, 2005). If this case occurs, then according to (2), u = v = O (point of infinity) and the verification phase is also will be performed correctly. This feature increases the performance of the protocol. The security of this protocol depends on the difficulty of solving the ECDLP and the resistances of hash-function against attacks as well other similar protocols.

Signature generation by means of a reliable method: In all corresponding protocols, it is required to embed and hide the digest into the signature such that it would be impossible to detect the digest in the signature. In other word, digest's presence in the signature cannot be tracked or detected. Hence, no one can forge the signature. In all known similar protocols, this uniqueness is achieved by means of two parameters that are unknown to everyone except to the owner of the signature. These parameters are the private key d and the integer k where k is randomly generated by the signer for each message and d # k.

In general, if X and Y are some unknown values that are generated by d and k, then the signature will be Xe + Y. As an instance, in ECDSA (Johnson *et al*., 2001) and EC EL-Gamal Digital Signature Scheme (Rabah, 2005), the signature is employed as $(k^{-1})e + (drk^{-1})$. However, in the proposed scheme the signature is (dr)e + k which employs the parameters d, k and r. The effect of employing r parameter along with the unknown parameters d and k for generating the signature will be appeared later.

The impossibility of signature forging: Here, we examine how forging the signature is not possible in the proposed protocol. In general, assume a forged signature for the message m is $(s + y = s^*, F^*)$, instead of the original signature (s, F). We shall prove that no values can be found for y and F^* such that satisfies the Eq. 3 for the verification.

$$(dre + k + y)G = r^*e^*(dG) + F^* \quad (3)$$

Initially it will be presented that even with selection of y appropriately and arbitrarily, no value can be found for F^*. From Eq. 3, the relation of Eq. 4 can be emerged. In this equation, in order to estimate an appropriate value of F^*, the attacker has access to all the left side parameters except the $r^* e^*(dG)$ value, where r^* is the x-coordinate of F^*.

$$re(dG) - r^*e^*(dG) + kG + yG = F^* \quad (4)$$

Hence, since the value of F* is not available, then r* will be unknown as well. Therefore, the left term of the Eq. 4 can not be computed. Consequently, calculating an appropriate value for F* is not possible. Another approach by the attacker may be by means of selecting the y value appropriately and eliminating the r^* from the left side of the equation. For this approach, the attacker has to select the y value equal to dr^*e^*. However, since the value of d is not available for signature forger, the r^* can not be eliminated and the appropriate value of F* can not be generated.

It will be presented at this section that if the value of F^* is chosen appropriately and arbitrarily with no precondition, it would be impossible to calculate the value of *y* alone and therefore rendering the forging impossible. From Eq. 3 the relation Eq. 5 is concluded:

$$yG = r^*e^*(dG) + F^* - (dre + k)G \quad (5)$$

The parameters on the right of Eq. 5 are available to the forger, in other words, the forger has the value of the product yG. However, according to ECDLP it is not possible to estimate the value of y. Hence, even with an arbitrary value of F^*, y value cannot be estimated and making the forging impossible.

RESULTS AND DISCUSSION

Table 2 defines our notation. From Koblitz *et al.* (2000), the time complexity for various operation units in terms of time complexity of modular multiplication is shown in Table 3.

The time complexity of the proposed protocol and some other protocols in terms of modular multiplication operation, modular inverse operation and one-way hash function is shown in Table 4. The proposed protocol does not require any inverse computation for the signature generation phase and the verification phase. Initially, required computational cost for each protocol has been estimated by means of adding the execution time of required operations. Latter based on Table 3, all the estimated times have been exhibited in terms of the required execution time for modular multiplication or inversion.

Table 2: Definition of given notations

Notation	Definition
T_{MUL}	Time complexity for the execution of a modular multiplication
T_{EXP}	Time complexity for the execution of a modular exponentiation
T_{ADD}	Time complexity for the execution of a modular addition
T_{EC_MUL}	Time complexity for the execution of a multiplication in an elliptic curve point
T_{EC_ADD}	Time complexity for the execution of an addition of two points in an elliptic curve
T_{inv}	Time complexity for the execution of a modular inversion
T_{HASH}	Time complexity for the execution of a one-way hash-function operation

Table 3: Unit conversion of various operations in terms of T_{MUL}

Time complexity of an operation unit	Time complexity in terms of modular multiplication
T_{EXP}	240 T_{MUL}
T_{EC_MUL}	29 T_{MUL}
T_{EC_ADD}	0.12 T_{MUL}
T_{ADD}	Negligible
T_{HASH}	Negligible

Table 4 shows that the proposed method in comparison to other protocols, has less time complexity for the signature generation and verification phases.

Since the method developed in Chung *et al.* (2007), similar to our method, does not require re-computing signature when s = 0, a further comparison is essential. Assuming the time complexity of the hash function is neglected, then we can estimate the speedup of the proposed protocol in comparison with Chung *et al.* (2007), as follows:

$$\text{Speedup} = \frac{\text{Time complexity of the Chung } et\ al.\text{'s protocol}}{\text{Time complexity of proposed protocol}}$$

Hence the speedup of signature generation and verification phases can be calculated as (6) and (7), respectively below:

$$\text{Speedup} = \frac{60.12T_{MUL} + T_{HASH}}{31T_{MUL} + T_{HASH}} \approx \frac{60.12T_{MUL}}{31T_{MUL}} \approx 1.93 \quad (6)$$

$$\text{Speedup} = \frac{87.24T_{MUL} + T_{HASH}}{59.12T_{MUL} + T_{HASH}} \approx \frac{87.24T_{MUL}}{59.12T_{MUL}} \approx 1.47 \quad (7)$$

Therefore, in comparison to the method in Chung *et al.* (2007), the signature generation and signature verification phases of our method is improved

Table 4: Required time complexity in unit of T_{MUL}

Various schemes	Signature generation phase	Time complexity in T_{MUL}	Verification phase	Time complexity in T_{MUL}	Mathematical base	†
Li *et al.* (2005)	$2T_{EXP}+6T_{MUL}+T_{INV}+T_{Hash}$	$246T_{MUL}+T_{INV}+T_{Hash}$	$4T_{EXP}+5T_{MUL}+T_{Hash}$	$965T_{MUL}+T_{Hash}$	IFP	-
Nyang and Song (2000)	$2T_{EXP}+T_{MUL}+T_{Hash}$	$481T_{MUL}+T_{Hash}$	$2T_{EXP}+T_{MUL}+T_{Hash}$	$481T_{MUL}+T_{Hash}$	DLP	-
EC ElGamal (Rabah, 2005)	$2T_{MUL}+T_{INV}+T_{EC\_MUL}+T_{Hash}$	$31T_{MUL}+T_{INV}+T_{Hash}$	$3T_{EC\_MUL}+T_{EC\_ADD}+T_{Hash}$	$87.12T_{MUL}+T_{Hash}$	ECDLP	YES
ECDSA (Johnson *et al.*, 2001)	$2T_{MUL}+T_{INV}+T_{EC\_MUL}+T_{Hash}$	$31T_{MUL}+T_{INV}+T_{Hash}$	$2T_{MUL}+T_{INV}+2T_{EC\_MUL}+T_{EC\_ADD}+T_{Hash}$	$60.12T_{MUL}+T_{Inv}T_{Hash}$	ECDLP	YES
Chung *et al.* (2007)	$2T_{MUL}+2T_{EC\_MUL}+T_{EC\_ADD}+T_{Hash}$	$60.12T_{MUL}+T_{Hash}$	$3T_{EC\_MUL}+T_{EC\_ADD}+T_{Hash}$	$87.24T_{MUL}+T_{Hash}$	ECDLP	NO
Proposed scheme	$2T_{MUL}+1T_{EC\_MUL}+T_{Hash}$	$31T_{MUL}+T_{Hash}$	$2T_{EC\_MUL}+1T_{EC\_ADD}+T_{MUL}+T_{Hash}$	$59.12T_{MUL}+T_{Hash}$	ECDLP	NO

†: Re-computing signature when s = 0 in step 4 of signature generation phase

by 93 and 47%, respectively. Therefore, it is clear that the proposed protocol can substantially raise the efficiency of signature generation and signature verification.

In general, elliptic curve discrete logarithm problem has a stronger mathematical base than the integer factorization (Nyang and Song, 2000) and discrete logarithm problem (Li *et al.*, 2005; Pointcheval and Stern, 2000; Hwang *et al.*, 2002; Hwang *et al.*, 2001). In some well known algorithms like RSA (Hwang *et al.*, 2000), the key length has to be 1024 bits for achieving high security (Rivest *et al.*, 1978). The same level of security is achieved in ECC by a key length of 160 bits. Therefore with the same level of security, the speed of ECC is several times faster than RSA cryptosystems.

CONCLUSION

In this research, the weakness of other protocols for digital signature based on the ECDLP is highlighted. A protocol based on the ECDLP is proposed for overcoming this weakness. The time complexity of the proposed protocol is compared with the previous works and results indicate the proposed protocol based on the ECDLP is more efficient than algorithms based on the IFP and DLP. Furthermore, the results also indicate the protocol has less time complexity even compared to other family protocols based on the ECDLP.

REFERENCES

Caelli, W.J., E.P. Dawson and S.A. Rea, 1999. Elliptic curve cryptography and digital signatures. Comput. Security, 18 (1): 47-66.

Chung, Y.F., K.H. Huang, F. Lai and T.S. Chen, 2007. ID-based digital signature scheme on the elliptic curve cryptosystem. Comput. Standards Interfaces, 29 (6): 601-604.

Coombes, K.R., 1999. Elliptic curves and logarithmic derivatives. J. Pure Applied Algebra, 138 (1): 21-38.

Diffie, W. and M.E. Hellman, 1976. New directions in cryptography. IEEE Trans. Inform. Theor., 22 (6): 644-654.

Hwang, M.S., I.C. Lin and K.F. Hwang, 2000. Cryptanalysis of the batch verifying multiple RSA digital signatures. Information, 11 (1): 15-19.

Hwang, M.S., C.C. Lee and E.J. Lu, 2001. Cryptanalysis of the batch verifying multiple DSA-type digital signatures. J. Applied Sci., 1 (3): 287-288.

Hwang, M.S., C.C. Chang and K.F. Hwang, 2002. An ElGamal-like cryptosystem for enciphering large messages. IEEE. Trans. Knowledge Data Energy, 14 (2): 445-446.

Hwang, S.J. and H.C. Liao, 2005. Security of Tzeng-Hwang's authenticated encryption scheme based on elliptic curve discrete logarithm problems. Applied Math. Comput., 168 (1): 717-721.

Johnson, D., A. Menezes and S. Vanstone, 2001. The elliptic curve digital signature algorithm (ECDSA). Int. J. Inform. Security, 1 (1): 36-63.

Koblitz, K., 1987. Elliptic curve cryptosystems. Math. Comput., 48 (177): 203-209.

Koblitz, N., A. Menezes and S. Vanstone, 2000. The state of elliptic curve cryptography. Design Code Cryptogr., 19 (2-3): 173-193.

Lawrence, C., 2003. Elliptic Curves Number Theory and Cryptography. CRC Press, Washington.

Li, L.H., S.F. Tzeng and M.S. Hwang, 2005. Improvement of signature scheme based on factoring and discrete logarithms. Applied Math. Comput., 161 (1): 49-54.

Miller, V., 1986. Uses of elliptic curves in cryptography. In: Advances in Cryptology-Crypto '85, Lecture Notes in Computer Science, Vol. 218, Springer, Berlin Heidelberg New York, pp: 417-426.

Nyang, D.H. and J.S. Song, 2000. Knowledge-proof based versatile smart card verification protocol. Comput. Commun. Rev., 30 (3): 39-44.

Pointcheval, D. and J. Stern, 2000. Security arguments for digital signatures and blind signatures. J. Cryptol., 13 (3): 361-396.

Rabah, K., 2005. Elliptic curve elgamal encryption and signature schemes. Inform. Technol. J., 4 (3): 299-306.

Rivest, R.L., A. Shamir and L. Adleman, 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM., 21 (2): 120-126.

Smart, N., 1999. The discrete logarithm problem on elliptic curves of trace one. J. Cryptol., 12 (3): 193-196.

Tzeng, S.F. and M.S. Hwang, 2004. Digital signature with message recovery and its variants based on elliptic curve discrete logarithm problem. Comput. Standards Interfaces, 26 (2): 61-71.

Vanstone, S.A., 1997. Elliptic curve cryptosystem-the answer to strong, fast public-key cryptography for securing constrained environments. Inform. Security Tech. Rep., 12 (2): 78-87.

Yu, Y.L. and T.S. Chen, 2005. An efficient threshold group signature scheme. Applied Math. Comput., 167 (1): 362-371.

Zakerolhosseini, A. and E. Malekian, 2007. Data Security. Nass, Iran.

Zheng, Y. and H. Imai, 1998. How to construct efficient signcryption schemes on elliptic curves. Inform. Process. Lett., 68 (5): 227-233.