Mutual Authentication for Wireless Communication Using Elliptic Curve Digital Signature Based on Pre-known Password

Tumpa Roy[1], Poonam Sisodia[1], Divye Upadhyay[1], and Kamlesh Dutta[2]

[1] GLNA Institute of Technology, Mathure- 281406
tumpa.nit@gmail.com,
poonamsisodi22@gmail.com,
divs08divyaa@gmail.com
[2] National Institute Of Technology Hamirpur
Hamirpur, Himachal Pradesh-177005
India
kdnith@gmail.com

Abstract. The appearance of public access wireless networks enables ever-present Internet services, whereas it inducing more challenges of security due to open air mediums. As one of the most widely used security mechanisms, authentication is provide for secure communications by preventing unauthorized usage and negotiating credentials for verification. In the intervening time, it generates heavy overhead and delay to communications, further deteriorating overall system performance. First, a system model based on challenge/response authentication mechanism by using the elliptic curve cryptographic digital signature is introduced, which is wide applied in wireless environment to reduce the computational cost, communication bandwidth and the server overload . Then, the concept of security levels is proposed to describe the protection of communications with regard to the nature of security.

Keywords: Elliptic curve cryptography (ECC), security, wireless communication, Public key cryptography (PKC), Authentication, verification.

1 Introduction

You Wireless communications is advancing rapidly in recent years. After 2G (e.g. GSM) widely deployed in the world, 3G mobile communication systems are spreading step by step in many areas. At present, some countries have already launched investigations beyond 3G (B3G) and 4G. Along with the wireless communications' rapid development, the secure access authentication of the users within wireless networks is becoming very critical, and so, more and more attention is focused on it. As the wireless industry explodes, it faces a growing need for security. Applications in sectors of the economy such as healthcare, financial services, and government depend on the underlying security already available in the wired computing environment. Both for secure (authenticated, private) Web transactions and for secure (signed, encrypted) messaging, a full and efficient public key

N. Meghanathan et al. (Eds.): Advances in Computing & Inform. Technology, AISC 176, pp. 11–22.
springerlink.com

infrastructure is needed. Three basic choices for public key systems are available for these applications:

- RSA
- Diffie-Hellman (DH) or Digital Signature Algorithm (DSA) modulo a prime p
- Elliptic Curve Diffie-Hellman (ECDH) or Elliptic Curve Digital Signature Algorithm (ECDSA).

RSA is a system that was published in 1978 by Rivest, Shamir, and Adleman, based on the difficulty of factoring large integers. Whitfield Diffie and Martin Hellman proposed the public key system now called Diffie-Hellman Key Exchange in 1976. DH is key agreement and DSA is signature, and they are not directly interchangeable, although they can be combined to do authenticate key agreement. Both the key exchange and digital signature algorithm are based on the difficulty of solving the discrete logarithm problem [15] in the multiplicative group of integers modulo a prime p. Elliptic curve groups were proposed in 1985 as a substitute for the multiplicative groups modulo p in either the DH or DSA protocols. For the same level of security per best currently known attacks, elliptic curve based systems [7,10] can be implemented with much smaller parameters, leading to significant performance advantages. Such performance improvements are particularly important in the wireless arena where computing power, memory, and battery life of devices are more constrained. In this article we will highlight the performance advantages of elliptic curve systems [8] by comparing their performance with RSA in the context of protocols from different standards.

Authentication is the act of establishing or confirming something as authentic, that is, that claims made by or about the subject are true. There are several methods concerning strong authentication. The main difference consists whether secret-key or public-key cryptography is used. In secret-key cryptography the signer and the verifier must share a secret where the problem of the key exchange must be solved. The main difference consists whether secret-key or public-key cryptography is used. In secret-key cryptography the signer and the verifier must share a secret where a public key is distributed for signature verification. The method using public-key cryptography is known as a digital signature. The protocols used for authentication consists of zero-knowledge protocols and challenge-response protocols. The Diffie-Hellman protocol [9] is used in wireless communication.

Deffie-hellman algorithm has five parts:

1. Global Public Elements
2. User A Key Generation
3. User B Key Generation
4. Generation of Secret Key by User A
5. Generation of Secret Key by User B

Global Public Elements:

q is a Prime number
$\alpha, \alpha < q$ and α is a primitive root of q

The global public elements are also sometimes called the domain parameters.

User A Key Generation:

Select private X_A, where $X_A < q$
Calculate public Y_A ,where $Y_A = \alpha .X_A \bmod q$

User B Key Generation:

Select private X_B, where $X_B < q$
Calculate public Y_B, where $Y_B = \alpha. X_B \bmod q$

Generation of Secret Key by User A:

$K = (Y_B).X_A \bmod q$

Generation of Secret Key by User B:

$K = (Y_A).X_B \bmod q$

If user A and user B are genuine then they can communicate to each other. The ECC version of algorithm is used in wireless communication for authentication proof.

2 Preliminaries

2.1 Elliptic Curve Cryptography

Elliptic curves [11] take the general form of the equation:

$$Y_2 + axy + by = x_3 + cx_2 + dx + e$$

where a, b, c, d and e are real numbers satisfy some conditions which depends on the field it belongs to, such as real number or finite field. Finite field may be F(p) or $F(2^m)$

The F(p) Field:

The elements of Fp [13] should be represented by the set of integers: {0, 1,. . . p – 1} With addition and multiplication defined as follows:

Addition: If $a, b \in F(p)$, then $a + b = r$ where r is the remainder of the division of a + b by p and $0< r < p-1$. This operation is called addition modulo p.

Multiplication: if $a, b \in F(p)$, then $a . b = s$ where s is the remainder of the division of a . b by p and $0< s < p-1$. This operation is called multiplication modulo p.

The $F(2^m)$ Field:

The elements of $F(2^m)$ should be represented by the set of binary polynomials of degree m – 1 or less: $a = \alpha_{m-1}x^{m-1} + \ldots + \alpha_1 x + \alpha_0$ with addition and multiplication defined as follows:

Addition: $a + b = c = \{_{cm-1},..c_1,c_0\}$ where $c_i = (a_i + b_i) \bmod 2$. $c \in F(2^m)$.

Multiplication: $a . b = c = \{c_{m-1},..c_1,c_0\}$ where c is the remainder of the division of the polynomial a(x) . b(x) by an irreducible polynomial of degree m. $c \in F(2^m)$.

There is a point 0 called the point at infinity or the zero point [12]. The basic operation of elliptic curve is addition. The addition of two distinct points on elliptic curve can be illustrated by the following figure [3] (figure 1):

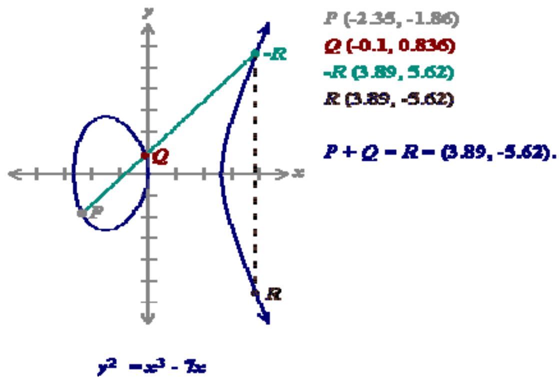

Fig. 1.

Elliptic Curve over F(p):
Let F(p) be a finite field, p > 3, and let a, b ∈ F(p) are constant such that

$$4a^3 + 27b_2 \equiv 0 \pmod{p}.$$

An elliptic curve, E(a,b)(F(p)), is defined as the set of points (x,y) ∈ F(p) * F(p) which satisfy the equation

$$y^2 \equiv x^3 + ax + b \pmod{p}$$

together with a special point, O, called the point at infinity.

Elliptic Curve over F(2m) for some m ≥ 1. :
Elliptic curve E(a,b)(F(2m)) [14] is defined to be the set of points (x,y) ∈ F(2m) * F(2m) which satisfy the equation

$$y^2 + xy = x^3 + ax^2 + b;$$

where a, b ∈ F(2^m) and b≠0, together with the point on the curve at infinity, O.The points on an elliptic curve form an abelian group under a well defined group operation. The identity of the group operation is the point O.

P and Q be two points on E(a,b)(F(p)) or F(2m) and O is the point at infinity.

P+O = O+P = P
If P = (x_1,y_1) then -P = $(x_1, -y_1)$ and P + (-P) = O.
If P = (x_1,y_1) and Q = (x_2,y_2), and P and Q are not O.
Then P +Q = (x_3, y_3) where
$x_3 = \lambda_2 - x_1 - x_2$,
$y_3 = \lambda(x_1 - x_3) - y_1$ and
$\lambda = (y_2-y_1)/(x_2-x_1)$ if $P \neq Q$; $\lambda = (3x_1^2+a)/ 2y_1$ if $P = Q$

2.2 Elliptic Curve Digital Signature Algorithm

The private key in DSA is a number X. It is known only to the signer. The public key in DSA consists of four numbers:

P = a prime number, between 512 and 1024 bits long
Q = a 160-bit prime factor of P-1.
G = h(P - 1)/Q, where H< P -1 and G mod Q> 1.
Y = G X mod P, which is a 160-bit number.

A signature on a document's hash value H consists of two numbers R and S:

R = (G K mod P) mod Q, where K is a randomly-chosen number <Q.
S = (K -1 (H + XR)) mod Q

To verify the signature, a recipient must compute a value V from the known information:

W = S -1 mod Q
U1 = HW mod Q
U2 = RW mod Q
V = ((G U1 Y U2) mod P) mod Q

If V = R, then document was signed by the person with the public key (P, Q, G, Y). The security of DSA is based on the computational infeasibility of finding a solution for the equation S = (K -1 (H + XR)) mod Q, when X is not known.

3 Proposed Protocol

Choosing a finite field Fq. An elliptic curve E defined over Fq with large group order and a point P of large order n is selected and made public, where n is a prime number. Zn is a class of modulo n, where n is the order of p over E(Fq). Given r,t $\in$ Zn , where r+t = o mod n, r is called the additive inverse of t and denoted as r =-t mod n. the server and client share a secret password S and a secret key K. the server and client individually compute two integers t and r. t is derived from Sand (n-1) in any predetermined way and it yields a unique value. The whole protocol divided into two phases:

Key establishment phase,
Verification phase.

3.1 Key Establishment Phase

The steps of the key establishment phase are explain bellow:

e.1 the client choose a random integer r_c which is belongs in between 1 to n-1 ie. $r_c \in$ (1, n-1). And compute Q_c =(r_c+t)P. the client send Q_c to the server.

e.2 The server then select a random integer r_s which is belongs in between 1 to n-1 ie. $r_s \in$ (1, n-1). And compute Q_s =(r_c+t)P. the server send Q_s to the client.

e.3 client compute X=Q_s+rP

$=(r_s+t)P+rP$
$= r_sP+tP+(-t)P$
$=r_sP$

And compute the session key $K_c=r_cX=r_cr_sP$

e.4. Server compute $Y=Q_c+rP$
$=(r_c+t)P+rP$
$= r_cP+tP+(-t)P$
$=r_cP$

And compute the session key $K_s=r_sY=r_cr_sP$

The session key computed by the server and client individually are same ie. Kc=Ks.

The figure 2 show the key establishment procedure between the client and server.

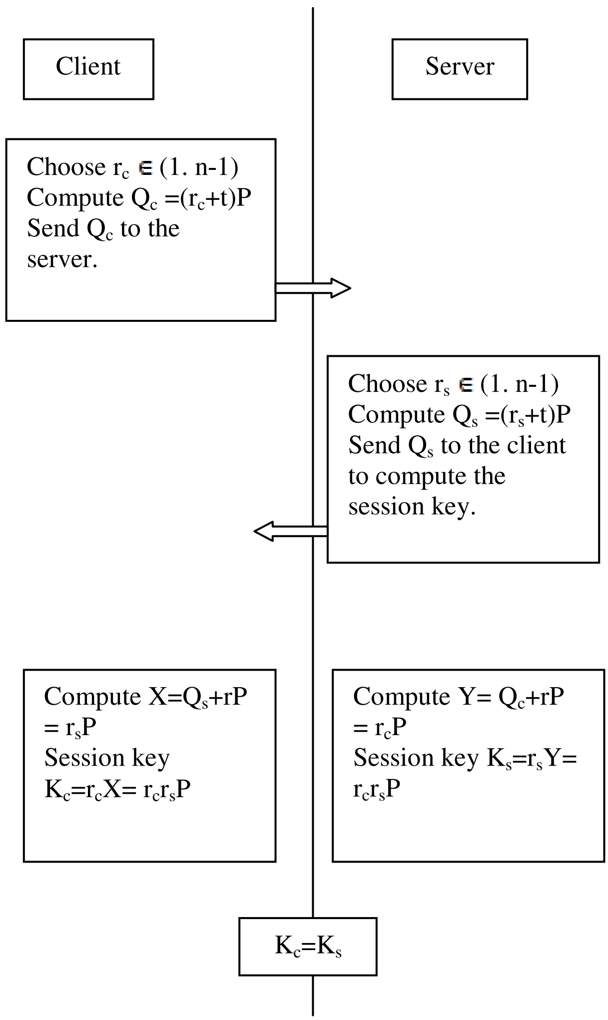

Fig. 2. Key Establishment Phase

3.2 Verification Phase

v.1 The client compute $K*K_c=K*r_c *r_s*P$ where K is the secret key which is known by the server and client. Client send the $K*K_c$ to the server to proof its validation.

v.2 server checks whether $K*K_c=K*K_s$ hold or not. if it dose server believes that it and the client have obtain the same session key i.e $K_c=K_s$ and the client is not duplicate because it knows the secret key which is only known by the server and the clinent. Since the server knows r_s , it believes it has obtain the accurate Q_C. Since client knows r_c , server believes client obtain the correct Q_c ie server condensed that the K_s is valid and the server compute $K*Q_c$ and send it to the client.

v.3 client checks $K*Q_c$. If $K*Q_c$ is correct, client believes that B has obtain the correct Q_c . since only server knows the the secret key K which is shared between the server and client and t is known by the server. So the server is not duplicate. The server knows the t beside client. Client believes that it has obtain the correct Qs and they have obtain the same session key $K_c=K_s$. Client convinced that the K_c is valid.

After the verification procedure has been completed by both sides, the client and the server are now ready to use the session key.

The figure 2 show the Verification procedure between the client and server.

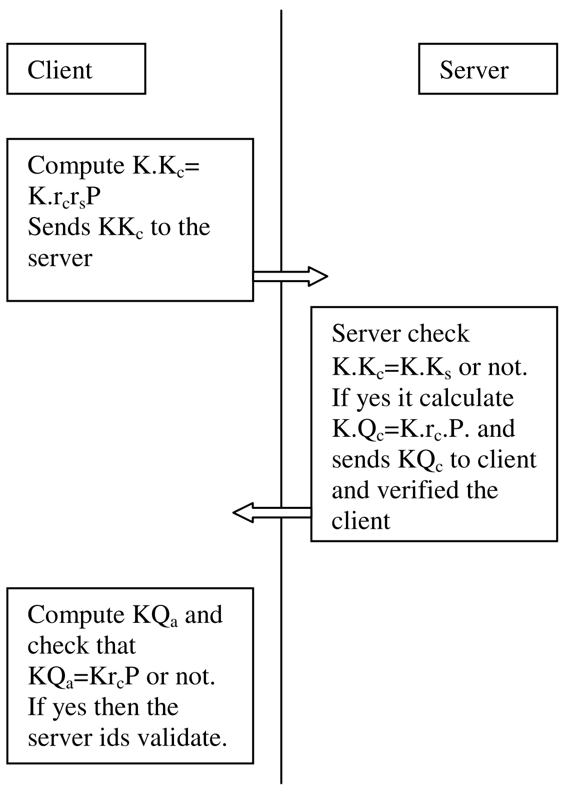

Fig. 3. Validation Phase

4 Analysis of the Proposed Protocol

4.1 Security Analysis

In this section, we scrutinize our proposed key agreement protocol in detail so as to ensure that our protocol is able to achieve the desired security attributes of a key agreement protocol and also able to resist against the known cryptographic attacks.

Known session key security (KSK-S). As shown in our protocol description, the session key is derived from the ephemeral keys (r_c, r_s) of the specific session and the long term keys (S,K) of the protocol entities. This would result in distinct independent session key in each protocol execution. On top of that, a one-way collision-resistant cryptographic function is used to derive the session key. Thus, obtaining any other session keys would not benefit the adversary in mounting a successful attack against a protocol run without the information set (r_c ,r_s),(t,r) which is required in the computation of the shared secret K. Therefore, we claim that the knowledge of some previous session keys would not allow the adversary to gain any advantage in deriving any future and other previous session keys.

Weak Perfect Forward secrecy (wPFS). Suppose that both client and server's long term secret key and password S and K have been exposed. However, the adversary, with the eavesdropped information of any particular session, would not be able to recover the respective established session key since the adversary does not know the involved ephemeral private key r_c or r_s which are needed in the computation of the shared secret K_c and K_s . And also, the intractability of ECCDHP has significantly thwarted the adversary's attempt in computing K_c and K_s by using S and K . Hence, we claim that our enhanced protocol enjoys weak perfect forward secrecy.

Key-Compromise Impersonation Resilience (KCI-R). Suppose that client's and server's long term private key S, K has been compromised and instead of directly impersonating client, the adversary now wishes to impersonate server in order to establish a session with client. However, the adversary is unable to compute the shared secret K_c with the available information (S, r_s, K) since the required information set is (r_c,S, K). Hence, the adversary is significantly prevented from launching a successful KCI attack against our protocol. Generally, the same situation will result when the long term key S is compromised (the adversary would impersonate client in this case and her effort will be foiled in computing K_c as our key agreement protocol is symmetric. As a result, we claim that this protocol is able to withstand the KCI attack under all circumstances.

Key Replicating Resilience (KR-R). The key replicating attack was first introduced by Krawczyk [1] where the illustration of it involves oracle queries described in Bellare and Rogaway's random oracle model [2,3]. This attack, if successfully carried out by the adversary, would force the establishment of a session, K (other than the Test session or its matching session) to agree on a same session key as the Test session, by means of intercepting and altering the message from both communicating parties during transmission. Since the Test session and K are non-matching, the adversary may issue a Reveal query to the oracle associated with K and she can then distinguish whether the Test session key is real or random. Notice that the message integrity of Qc and Qs has been guaranteed by having each party to calculate K_c and K_s which will be bound to X and Y respectively. Since the adversary has no idea in forging X or Y along with Q_c or Q_b she would not be able to force the establishment of non matching sessions to possess a common session key. As a result, if the adversary reveals client's session key, she would not be able to guess server's session key correctly with non-negligible probability and vice versa. Therefore, we claim that our protocol is secure against the key replicating attack.

Replay Resilience (R-R). In any protocol run, an adversary may attempt to deceive a legitimate participant through retransmitting the eavesdropped information of the impersonated entity from a previous protocol execution. Although the adversary might be unable to compute the session key at the end of the protocol run, her attack is still considered successful if she manage to trick the protocol entity to complete a session with her, believing that the adversary is indeed the impersonated party. In this replay analysis, we reasonably assume that the prime order n of point P is arbitrarily large such that the probability of a protocol entity selecting the same ephemeral key (r_c , $r_s \in [1, n - 1]$) in two different sessions is negligible. Consider a situation where the adversary (masquerading as A) replays A's first message from a previous protocol run between client and server. After server has sent her a fresh (Q_s, Y) in the second message flow, the adversary would abort since she could not produce (by means of forging or replaying) X corresponding to Q_s . Notice that the same replay prevention works in the reverse situation where server's message is replayed. The adversary would fail eventually in generating Y corresponding to the fresh Q_c. Hence, we claim that message replay in our protocol execution can always be detected by both client and server.

Identity authentication. On the one hand, assuming Eve can impersonate B. When Eve receives Q_c, E $\rightarrow$A: $Q_e = (r_e + t)P$. But Eve does not know t and re, and she cannot make the validation message KrcrsP, thus the key validation fails. On the other hand, with (v.2) and (v.3), A and B believe that only knowing t can generate the valid validation messages.

Man-in-the-middle attacks. In the original Diffie-Hellman protocol, Eve can alter the public values such as g_a mod n or g_b mod n with her own values. Thus Eve can share session keys with client or server. In our protocol, when Eve receives Qc= $(r_c + t)P$, she cannot guess rc and t. If she still tries to eavesdrop, she mus t generate $r_eP = (r^{c'} + t)P$ and send it to server; server will obtain a wrong value rc'rsP, which is impossible for Eve to know. Thus Eve cannot share a session key with server or client. Based on ECDH algorithm [4], our protocol with pre-shared password is proposed. It makes use of the difficulty of computing discrete logarithms over elliptic curves. It provides identity authentication, key validation and perfect forward secrecy, and it can foil man-in-the-middle attacks.

4.2 Performance Analysis

Efficiency Analysis

Atay et. al. have conducted detailed studies on Computational Cost Analysis of Elliptic Curve Arithmetic [5]. They have reported the point addition arithmetic is applied on two and three dimensional coordinate systems. The computational cost of each arithmetic operation should be taken into consideration in order to compare the efficiency of algorithms in different coordinate systems. The efficiency is measured as the computational cost, which is in terms of elapsed time. The measured units in Fig. 4 [10] are as follows:

1. Inversion (I) is the multiplicative inverse in modular arithmetic. It has the highest computational cost and one inversion is approximately equals nineteen times of the cost of multiplication cost and denoted as 1I =19M .
2. Multiplication (M) has a lower cost than inversion; therefore all inversions should be converted either to multiplication or to addition.
3. Addition (A) and subtraction (S) have the lowest cost, therefore omitted.

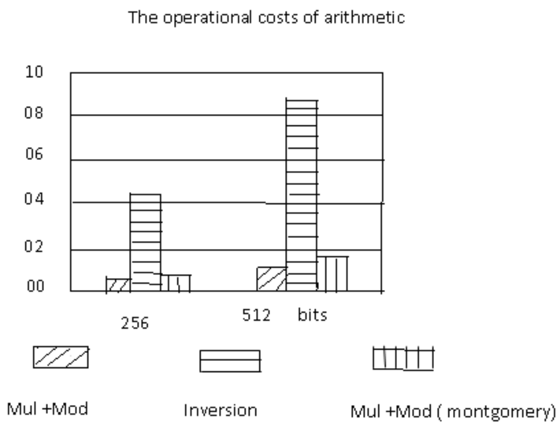

Fig. 4. The operational cost of Arithmetic operation

Computational Cost Analysis

The major advantage of ECC over RSA is ECC needs less computation than RSA but still can achieve the same or even higher level of security. Table 1[6] gives cost-equivalent key sizes. It gives the size, in bits, for equivalent keys. The time to break is computed assuming a machine can break a 56-bit DES key in 100 seconds, and then scaling accordingly.

Table 1.

ECC key	RAS key	Time to break	Machines	Memory
112	430	<5 minutes	105	`Trivial
106	760	600 months	4300	4Gb
192	1020	3 million years	114	170 Gb
256	1620	10^16 years	16	120Gb

5 Conclusion

Attack that monitor side-channel information, Key Replicating Resilience (KR-R).the key replicating attack was first introduced by Krawczyk have recently been receiving much attention in wireless communication. The result presented in this paper conform that the key replacing attack quite powerful and need to be addressed. Any addition to memory or processing capacity increases the cost of each card. ECC needs less computation power, thus it is more suitable than RSA. We have described an authentication and key agreement protocol for wireless communication based on elliptic curve cryptographic techniques. The proposed protocol is a public key type with the feature of signature generation procedure. The new protocols are based on previous classic authentication protocols, including the protection of integrity and session key exchange. This can be used to provide the integrity of the data being transferred during the authentication process in order to prevent from active attacks. The smaller key sizes result in smaller system parameters, smaller public key signatures, bandwidth savings, faster implementations, and smaller hardware processors.

References

1. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
2. Bellare, M., Rogaway, P.: Provably Secure Session Key Distribution: The Three Party Case. In: 27th ACM Symposium on the Theory of Computing - ACM STOC, pp. 57–66 (1995)
3. Blake-Wilson, S., Johnson, D., Menezes, A.: Key Agreement Protocols and their Security Analysis. In: Darnell, M.J. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 30–45. Springer, Heidelberg (1997)
4. Miller, V.S.: Use of Elliptic Curves in Cryptography. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
5. Atay, S., Koltuksuz, A., Hışıl, H., Eren, S.: Computational Cost Analysis of Elliptic Curve Aurithmetic. In: International Conference on Brid Information Technology, ICHIT 2006, vol. 1, pp. 578–582 (2006)
6. Chatterjee, K., Gupta, D.: Secure access of smart cards using Elliptic curves Cryptography. In: 5th International Conferences on Wireless Communications, Networking and Mobile Computing, WiCom 2009, pp. 1–4 (2004)
7. Blake, I.F., Seroussi, G., Smart, N.P.: Elliptic Curves in Cryptography. London Math. Soc. Lecture Note Series, vol. 265. Cambridge Univ. Press (2000)
8. Aydos, M., Sunar, B., Koç, Ç.K.: An Elliptic Curve Cryptography Based Authentication and Key Agreement Protocol for Wireless Communication. In: 2nd Int. Workshop Discrete Algorithms and Methods for Mobility, DIALM 1998, Dallas, TX (1998)
9. Diffie, W., Hellman, M.: New Directions in Cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
10. Strangio, M.A.: Efficient Diffie-Hellmann Two-Party Key Agreement Protocols based on Elliptic Curves. In: Proceedings of the 2005 ACM Symposium on Applied Computing, pp. 324–331 (2005)

11. Miller, V.S.: Use of Elliptic Curves in Cryptography. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
12. Blake, I.F., Seroussi, G., Smart, N.P.: Elliptic curves in cryptography. Cambridge University Press, New York (1999)
13. Chen, L., Yanpu, C., Zhengzhong, B.: An implementation of fast algorithm for Elliptic Curve Cryptosystem over GF(p). Journal of Electronics 21(4), 346–352 (2004)
14. Morales-Sandoval, M., Feregrino-Uribe, C., Cumplido, R., Algredo-Badillo, I.: An area/performance trade-off analysis of a GF(2m) multiplier architecture for Elliptic Curve Cryptography. Computers and Engineering 35, 54–58 (2009)
15. Smart, N.P.: The discrete logarithm problem on elliptic curves of trace one. Journal of Cryptology 12(3), 193–196 (1999)
16. Lin, X., Wong, J.W., Kou, W.: Performance analysis of secure web server based on SSL, pp. 249–261. Springer, Heidelberg (2000)