A Secure and Efficient Authentication Protocol Based on Elliptic Curve Diffie-Hellman Algorithm And Zero Knowledge Property

Manoj Kumar

Abstract— Elliptic curves have been extensively studied for over hundred years, originally pursued mainly for aesthetic reasons; elliptic curves have recently become a tool in several important applied areas, including coding theory, pseudo-random bit generation and number theory algorithms. Actually ECC is an alternative approach for traditional public key cryptography like RSA, DSA and DH. It provides the highest strength-per-bit of any cryptosystem known today with smaller key size resulting in faster computations, lower power assumption and memory. Another advantage is that authentication protocols based on ECC are secure enough even if a small key size is used. It also provides a methodology for obtaining high speed, efficient and scalable implementations of protocols for authentication and key agreement. The present paper consists of an introduction to elliptic curves and an authentication protocol based on ECC and zero knowledge property. The protocol is developed for group communication where every person of the group has a secret information and the communication starts when all this information is put together. If one person is not online, the others cannot communicate.

Keywords and phrases-Elliptic Curves, Cryptosystem, Authentication Protocols, Public key Cryptography, Finite Fields.

I. INTRODUCTION

The ECC was first developed independently by Victor Miller [15] and Neal Koblitz [11] in 1985. Compared to its traditional counter parts, ECC offers the same level of security using much smaller keys. This result in faster computations and savings in memory, power and bandwidth those are especially important in constrained environments. More significantly, the advantage of ECC over its competitors increases, as the security needs increase over time. The National Institute of Standards and Technology (NIST) approved ECC for use by the U.S. government [16]. Several standards organizations such as Institute of Electrical and Electronics Engineers (IEEE), American National Standards Institute (ANSI), Open Mobile Alliance (OMA) and Internet Engineering Task Force (IETF) have ongoing efforts to include ECC as a required or recommended security mechanism.

A. Some Notations and Basics of ECC

The detail study of fundamentals of ECC and its mathematical background can be found in [1, 12,14]. Some basic concepts of ECC that are essential to understand the mathematical descriptions of elliptic curves used in the cryptographic schemes, are described in brief, as follows.

Scalar: Any element of the finite field $GF(p)$ or $GF(2^k)$, where p is a prime number and k is a positive integer, is known as a scalar. Generally lowercase letters $(m, n, k, p, q, r, s \text{ etc})$ are used to denote scalars.

Point on Elliptic Curve: A point on elliptic curve is defined as an ordered pair (x, y) of scalars, satisfying the equation of that particular elliptic curve. Generally, a point is denoted by uppercase letters (P, Q, R etc.). An alternative notation for a point P is $P \equiv P(x, y)$.

Scalar Multiplication: A new scalar can be obtained by the multiplication of two or more scalars.

Point Multiplication: For a given scalar k and a point P, the point multiplication is defined as

$$kP = P + P + P + ... + \text{upto } k \text{ terms}$$

which is again a new point on elliptic curve. In case the scalar k is a very large value, we can also use the Montgomery's point multiplication method which is described as

INPUT: An integer k and a point $P \in E_P(a,b)$

OUTPUT: $Q = kP \in E_P(a,b)$

Montgomery's Point Multiplication Algorithm

Given $P \in E_P(a,b)$ and $k = (k_{r-1}, k_{r-2}, ..., k_2, k_1, k_0)_2$, compute $Q = kP$

1. $k \leftarrow (k_{r-1}, k_{r-2}, ..., k_2, k_1, k_0)_2$
2. $P_1 \leftarrow P$
3. $P_2 \leftarrow 2P_1$
4. for $i = r-2$ down to 0 do
5. if $k_i = 1$, then
6. $P1 \leftarrow P_1 + P_2$
7. $P2 \leftarrow 2P_2$
8. else
9. $P2 \leftarrow P_2 + P_1$
10. $P1 \leftarrow 2P_1$
11. end if
12. end for
13. $Q \leftarrow P_1$
14. return (Q)

Actually elliptic curves are not ellipses. They are so called because they are described by cubic equation, similar to those are used for calculating the circumference of an ellipse.

Manuscript Received on November, 2013.

Manoj Kumar, Department of Mathematics and Statistics, Gurukul Kangri Vishwavidyalaya, Haridwar, Uttrakhand-249404, India.

In general an elliptic curve is given by

$$y^2 + a\,x\,y + b\,y = x^3 + cx^2 + d\,x + e$$

where a,b,c,d,e are some fixed real numbers and the variables x, y take the values in the set of real numbers.

For our purpose, we take the following elliptic curve E defined over a finite field $GF(p)$

$$y^2 (\text{mod } p) = (x^3 + a\,x + b)(\text{mod } p) \qquad (1)$$

where p is a prime number. If the discriminant $\Delta = (4a^3 + 27b^2)(\text{mod } p)$ is non zero, then the elliptic curve E is defined as the set of points (x, y) satisfying the equation (1) including the point O called the zero point or the point at infinity on the elliptic curve *i.e.* For given p, a and b, E is defined as

$E_p(a,b) = \{(x,y): y^2(\text{mod } p) = (x^3 + a\,x + b)(\text{mod } p)\} \cup \{O\}$

, where $O \neq (0,0)$.

For all the points $P(x_1, y_1), Q(x_2, y_2) \in E_p(a,b)$, we have the following properties

1. $P + O = O + P = P$
2. $-P = (x_1, -y_1)$
3. $P(x_1, y_1) + Q(x_2, y_2) = R(x_3, y_3)$
4. $P(x_1, y_1) + P(x_1, y_1) = 2P(x_1, y_1) = R(x_3, y_3)$

   where $x_3 = (\lambda^2 - x_1 - x_2)(\text{mod } p)$,

   $y_3 = [\lambda(x_1 - x_3) - y_1](\text{mod } p)$,

and $$\lambda = \begin{cases} \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(\text{mod } p) & \text{if} \quad P \neq Q \\ \left(\frac{3x_1^2 + a}{2y_1}\right)(\text{mod } p) & \text{if} \quad P = Q \end{cases}.$$

The performance of an elliptic curve depends upon the scalar multiplication [7, 10, 18] operation.

In the case of binary field, an elliptic curve is also given by (1). Here we have

5. $-P = (x_1,\ x_1 + y_1)$
6. $P(x_1, y_1) + Q(x_2, y_2) = R(x_3, y_3)$

where $x_3 = \lambda^2 + \lambda + a + x_1 + x_2$,

$y_3 = \lambda(x_1 + x_3) + (y_1 + x_3)$ and $\lambda = \frac{y_2 + y_1}{x_2 + x_1}$.

7. $P(x_1, y_1) + P(x_1, y_1) = 2P(x_1, y_1) = R(x_3, y_3)$

where $x_3 = \lambda^2 + \lambda + a$, $y_3 = (\lambda + 1)x_3 + x_1^2$ and

$\lambda = \frac{x_1^2 + y_1}{x_1}$.

The security of ECC depends on the ECDLP (Elliptic Curve Discrete Logarithm Problem) which states that for given two points P and Q on elliptic curve, it is relatively very hard (almost impossible) to determine the value of k, such that $Q = kP$. To avoid this problem the elliptic curve must be chosen carefully. In February 2000, FIPS 186-1 was revised by NIST to include the ECDSA(Elliptic Curve Digital Signature Algorithm) as specified in ANSI X9.62[2] with further recommendations for the selection of underlying finite fields and elliptic curves, the revised standard is called FIPS 186-2[16]. FIPS 186-2 has 10 recommended finite fields, 5 for prime fields $GF(p)$, $p = 192, 224, 256, 384, 521$ and 5 for binary fields $GF(2^m)$, $m = 163, 233, 283, 409, 571$. For each of the prime fields, one randomly selected elliptic curve was recommended, while for each of the binary fields one elliptic curve (randomly selected) and one Koblitz curve was selected. In order to allow for efficient modular reduction, the prime p for the prime fields $GF(p)$ were chosen to either be a Mersenne prime or a Mersenne-like prime with bitwise a multiple of 32. For binary fields $GF(2^m)$, m was chosen so that there exists a Koblitz curve of almost prime order over $GF(2^m)$. Table-1 shows NIST guidelines [17] on choosing computationally equivalent symmetric and public key sizes.

Symmetric	ECC	RSA / DH / DSA	MIPS Yrs to attack	Protection Lifetime
80	160	1024	10^{12}	Untill 2010
112	224	2048	10^{24}	Untill 2030
128	256	3072	10^{28}	Beyond 2031
192	384	7680	10^{47}	Beyond 2031
256	512	15360	10^{66}	Beyond 2031

Table1: Equivalent Key Sizes (in bits)

From the above table it is obvious that, at higher key sizes, RSA performance issues become even more acute. Since the performance advantage of ECC over RSA grows approximately as the cube of the key size ratio, wider adoption of ECC seems inevitable.

B. Authentication

In our daily life, information security is crucial which is necessitated by the profitable and legal trading confidentiability, integrity and non-reputability of the associated information. A secure communication system contains preferred properties which may be include any or all of the following [20]

Data confidentiability: The protection of data from unauthorized disclosure.

Authentication: The assurance that the communicating entity is the one that it claims to be.

Access Control: The prevention of unauthorized use of resource.

Data Integrity: The assurance that data received are exactly as sent by an authorized entity (i.e. contain no modification, insertion, deletion or replay).

Non-Repudiation: Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

Anti-Replay: The message should not be permitted to

be sent to multiple recipients, devoid of the sender's knowledge.

Proof of delivery: The sender should possess the ability to prove that the message was received by the recipient.

Authentication can be said as any process by which one can verify that someone is who they claim, they are. Privacy protection is supported by authentication which ensures that entities verify and validate one another before disclosing any secret information. Authentication allows access to services and infrastructure by authorized entities only, while denying unauthorized entities access to sensitive data, thereby supporting confidentiability and access control. Some features that made ECC [5, 9] more suitable for authentication are as follows

- For a given key size, ECC offers considerably greater security.
- For a given level of security, much more compact implementations are also made possible by the smaller key size, which means faster cryptographic operations, running on smaller chips or more compact software. This also means less heat production and less power consumption, all of which is of particular advantage in constrained devices, but of some advantage anywhere.
- Extremely efficient, compact hardware implementations are available for ECC exponentiation operations, in which potential reductions in implementation foot print even beyond those due to the smaller key length alone are offered.

All these factors show that ECC is fit for authentication and hence on the basis of the mathematical theory of elliptic curve, we defined a new authentication protocol for group communication.

II. RELATED WORK

Chalkias et al [4] proposed an authentication protocol for exchanging encrypted messages via an authentication server based on ECC using ElGamal's algorithm. The ElGamal's algorithm is used for encryption/decryption and also for the authentication. The protocol proposed by Chalkias briefly described as

Sending Process :(From=Alice, To=Bob)

1. $A \rightarrow S$:Pub_S({ From, To, N_1 })
2. $S \rightarrow A$:Pub_A({ Pub_B, N_1, N_2, Id })
3. $A \rightarrow S$:{ Pub_B({ M, N_2 }), Id }

Receiving Process :(Bob receives the message)

4. $B \rightarrow S$:Pub_S({ To, N_3 })
5. $S \rightarrow B$:{ Pub_B ({ M, N_2 }), Pub_B ({ N_2, N_3 }) },

where M-message, Id-mail identity, S-Server, A(Alice)and B(Bob)-agents, Pub_B-public key of Bob; N_1 , N_2, N_3-nonces, { M_1, M_2 } -compound messages, Pub_B({ M_1, M_2 })-encrypted message.

In the first step the user A sends a request to the server *S*. The request consists of three elements namely From (nick name of sender), To (nick name of receiver) and N_1 a random message, needed for the verification of the server when the server replies. All the three elements are concatenated (using tags) and create a stream which is encrypted with the server's public key and sent to the server. In second step after decryption, the server must reply by providing a mail Id, the N_1, a new nonce N_2 and the receiver's public key to the sender. Meanwhile, the server inserts a new row in his internal mail data base. Every row consists of the following fields{ From, To, Id, Pub_B({ M, N_2 }), Pub_S(N_2) }. In the third step sender verifies the N_1 and if the verification is correct then he/she concatenates the clear message with N_2 and encrypts them with the recipient's public key. Next he/she sends this encrypted message to the server along with the Id.

In receiving process, if the receiver wants to download his message, he/she sends an encrypted stream consisting of his nickname and a new random message N_3, to the server. In last step after decryption, the server sends two streams to the receiver. The first stream is the encrypted message that was stored in database and, the second stream consists of decrypted N_2 and N_3 which is encrypted with recipient's public key. Finally after receiving the streams the receiver verifies the N_3 and then he compares the N_2 with the received two streams. The authors [4] recommend the proposed protocol to be use for controller-pilot data link communications but it does not mean that it cannot be adapted for other communications. They use the ECDSA (Elliptic Curve Digital Signature Algorithm) for authentication which is briefly in the next section.

III. ELLIPTIC CURVE DIGITAL SIGNATURE ALGORITHM (ECDSA)

First an elliptic curve E is defined over $GF(p)$ or $GF(2^m)$ with large group of order n and a point P of large order is selected by communicating parties and made to all users. Then the following key generation primitive is used by each party to generate the individual public and private key pairs. Furthermore, for each transaction the signature and verification primitives are used. A brief discussion of ECDSA is given below, details of which can be found in [8]. Generally ECDSA involves the three parts

1. ECDSA Key Generation: The user A follows three steps

(i). Choose a random integer d_1 such that $2 \le d_1 \le n-2$

(ii). Calculate $Q = d_1 P$

(iii). d_1 and (E, P, n, Q) are the private and public keys of user A, respectively.

2. ECDSA Signature Generation: The user A signs the message m using the five steps

(i). Choose a random integer d_2 such that $2 \le d_2 \le n-2$

(ii). Calculate $d_2 P = (x_1, y_1)$ and $r = x_1 \bmod n$. If $r = 0$ then go to step (i)

(iii). Calculate $d_2^{-1} \bmod n$

(iv). Calculate $s = d_2^{-1}(H(m) + d_1 r) \bmod n$. If $s = 0$ then go to step (i)

(v). (r, s) is the signature for the message m.

3. ECDSA Signature Verification: After receiving the signature

(r,s) of the user A on the message m, the user B verifies the signature using the four steps

(i). Calculate $c = s^{-1} \bmod n$ and $H(m)$

(ii). Calculate $u_1 = H(m)c \bmod n$ and $u_2 = rc \bmod n$

(iii). Calculate $u_1 P + u_2 Q = (x_2, y_2)$ and $v = x_2 \bmod n$

(iv). Signature (r,s) is accepted if $v = r$.

After verifying the signature, the user and the server have to create a secret key for the encryption. The advantages and disadvantages of ECDSA can be found in [8]. Aydos et al[3] proposed an authentication and key agreement protocol for wireless communication based on ECC. They use ECDSA for the authentication and the Diffie-Hellman key exchange scheme to generate session key. Although this protocol has including lower computational burden, lower communication band-width and storage requirements, Mangipudi et al [13] showed that the protocol is vulnerable to the man-in-the middle attack [21] from the attacker within the system. To prevent this attack Mangipudi et al [13] proposed an user authentication protocol which is a variant of Aydos's protocol. Like the Aydos's protocol, the Mangipudi's protocol has two phases namely the initialization phase and the user authentication phase. Unlike the Aydos's protocol, in the first phase of Mangipudi's protocol the certificate authority sends server's public key and expiration time whereas in Aydos's protocol in the same phase the certificate authority sends its own public key. The Mangipudi's protocol can resist man-in-the middle attack. However there are still some vital security deficiencies in this protocol. Firstly this protocol does not provide user's entity authentication to server. Secondly it does not provide forward security because disclosing server's secret key will lead to all previous secret information being disclosed. Thirdly it does not provide explicit (session) key authentication. This makes both communication parties can not know whether the other party exactly calculates the session key. Fourthly it is difficult to implement the renewal for security of wireless communication system based on this protocol. Once server's secret key is disclosed, all previous communication messages between users and server including the user's certificate will be disclosed.

IV. ELLIPTIC CURVE DIFFIE-HELLMAN (ECDH)

ECDH protocol establishes a shared key between two parties. The original Diffie-Hellman algorithm is based on the multiplicative group modulo p, while the ECDH protocol is based on the additive elliptic curve group. First of all a base point $P(x,y)$ of order n is selected on the elliptic curve E defined over the field $GF(p)$ or $GF(2^k)$. In brief the ECDH can be described as follows

User(U)	Server(S)
(i). Select random number $d_U \in [2, n-2]$	**(i).** Select random number $d_S \in [2, n-2]$
(ii). Compute $Q_U = d_U P$	**(ii).** Compute $Q_S = d_S P$
(iii). Send Q_U to server S	**(iii).** Send Q_S to user U
(iv). Receive Q_S	**(iv).** Receive Q_U
(v). Compute $K = K_U = d_U Q_S = d_U d_S P$	**(v).** Compute $K = K_S = d_S Q_U = d_S d_U P$

1. The user U selects a random number $d_U \in [2, n-2]$ and after computing $Q_U = d_U P$ sends it to the server S.
2. Similarly the server S selects a random number $d_S \in [2, n-2]$ and after computing $Q_S = d_S P$ sends it to the user U.
3. After receiving Q_S the user U computes the key $K_U = d_U Q_S = d_U d_S P$.
4. Likewise, after receiving Q_U the server S computes the key $K_S = d_S Q_U = d_S d_U P$.
5. Both the user and the server have the same key $K = K_S = K_U$.

The improved version of ECDH provides a little more flexibility in the sense that established value can be pre-selected by the user and sent to the server. The protocol steps can be modified slightly for sending a secret value from the server to the user as shown below

User(U)	Server(S)
(i). Select random number $d_U \in [2, n-2]$	**(i).** Select random number $d_S \in [2, n-2]$
(ii). Compute $e_U = d_U^{-1} \bmod n$ and $Q_U = d_U K$	**(ii).** Compute $e_S = d_S^{-1} \bmod n$
(iii). Send Q_U to server S	**(iii).** Receive Q_U
(iv). Receive R_S	**(iv).** Compute $R_S = d_S Q_U = d_S d_U K$
(v). Compute $S_U = e_U R_S = e_U d_S d_U K$	**(v).** Send R_S to user U
(vi). Sends S_U to the server S	**(vi).** Receive S_U
	(vii). Compute $T = e_S S_U = e_S d_S K = K$

A detail study of ECDH protocol can be found in [3].

V. PROPOSED PROTOCOL

In this section we will propose an authentication protocol which is very helpful for group communication among big companies where share holders are very busy and cannot attend all the meetings. Most of the authentication protocols based on ECC, use the ECDSA. The reason behind this is that it provides a high level of security. For our proposed protocol we use the zero knowledge to authenticate the users. A zero Knowledge protocol is an interactive technique for one party to prove to another that a statement is true without revealing anything other than the veracity of the statement. The concept of zero knowledge was first introduced by Goldwasser et al [6]. A zero knowledge protocol must satisfy three properties

(i) Completeness: It means that if the statement is true, the honest verifier (that is, one following the protocol property) will be convinced of this fact by an honest prover.
(ii)Soundness: It means that if statement is not true no cheating prover can convince the honest verifier that it is true, except with some small probability.
(iii) Zero Knowledge: It means that if the statement is true, no cheating verifier learns anything other than this fact.
The aim of the zero knowledge is to prove the knowledge of a secret without revealing it. Each user from the group has a secret information and each one has to prove that he/she knows the information without revealing it to the server. Thus the prover is the user and the verifier is the server. Since the secret information of each user is different therefore the server will identify each user through a demonstration of his knowledge. The basic idea of the zero knowledge authentication is that the verifier asks a question related to the secret information in such a way that the answer does not reveal the secret. Schnorr's protocol [19] is one of the most popular zero knowledge protocol. Let p and q be two primes number such that q divides $(p-1)$. Let $g \neq 1$ be an element of order q in Z_p (the multiplicative group of integers modulo p). Also let G_q be the cyclic subgroup of order q generated by g. The integers p, q, g are known and can be common to a group of users. An identity consists of a private /public key pair. The private key w is a random non-negative integer less than q. The public key is computed as $y = g^{-w} \bmod p$.
The protocol is described as below
Common Input: p, q, g, y; A security parameter t.
Secret Input for a Prover: $w \in Z_q$ such that $y = g^{-w} \bmod p$.

1. Commitment by Prover

Prover picks $r \in Z_q$, compute $x = g^r \bmod p$ and send it to the verifier.

Prover $\xrightarrow{x=g^r}$ Verifier

2. Challenge from Verifier

Verifier selects a number $e \in [1\ 2^t]$ and sends it to the prover

Prover $\xleftarrow{e}$ Verifier

3. Response from Prover

Prover computes $s = (r + w.e) \bmod q$ and sends it to the verifier

Prover $\xrightarrow{s=r+w.e}$ Verifier

The verifier checks that $x = g^s\ y^e \bmod p$ and accepts if and only if equality holds.
It is well known that Schnorr's protocol is an honest verifier zero knowledge protocol of knowledge of w, the discrete logarithm of y. The details study of the protocol can be found in [19].
Schnorr's protocol based on elliptic curve is described as below
Let P be a point on the elliptic curve E defined over the finite field F_q, of order n (Prover's secret information key), then
(i) If α is the prover's secret information then user makes public $Z = \alpha P$.
(ii) The prover picks a random number r and sends $X = rP$ to the verifier.
(iii) The verifier picks a random number e and sends it to the prover.
(iv) The prover computes $Y = (\alpha e + r) \bmod n$ and sends it to the verifier.
(v) The reciver receives y and accepts if $yP + eZ = X$.

A. Assumption for proposed protocol

In order to implement our proposed protocol we have to work based on some assumptions.
The first assumption is that every user has to be connected to the server to send a message. The second assumption is that the public keys are published in a public directory and in a web page for security reasons. The third one is that the server has a key pair too. The last and the most important assumption is that the server's public key is the only key that does not need verification. The role of the server is very important because he / she take all responsibility of the group member's verification by providing to both sender and receiver a random session number known as nonce.

B. Authentication between user and server

The authentication between user and server need to be executed in real time. It consists of the four steps
(i) U → S : $(\text{Sends})_s$ (X ,N_1)
(ii) S → U : $(\text{Sends})_U$(e ,N_1)
(iii) U → S : $(\text{Sends})_s$ (y)
(iv) S → accept or reject.
In first step, the user sends a request to the server. The request consists of two elements $X = rP$ and N_1. N_1 is a random message needed for verification of the server when the server replies. These two elements are concatenated using tags and create a stream. This stream is encrypted with the server's key and then encrypted stream is sent to the server. In the second step, after receiving the request, the server decrypts the stream and gets X and N_1 .Then the server sends a random number e and N_1 encrypted with the user's keys to the user. In the third step the user receives the stream and verifies that it is send from the server because only the server knew the random message N_1 . The user encrypts $y = (\alpha e + r) \bmod n$ using server's key and sends it to the server. In the last step the server decrypts the received stream and gets y . The server accepts the user if the computation $yP + eZ = X$ is true otherwise the server will reject the user.
Each user follows the above authentication steps.

C. Communication Process among the group

Once the verification procedure is completed by the user and the server, the communication can start among the authenticated users of the group. All the users use the same key pair (P (public key), S (secret key)) because they all have to know the messages sent from any user of the group. This key pair along with the server's one is used for communication during each session. Two users cannot communicate through this

protocol without the knowledge of the others. If one user receives a message it can be read by all the others because they all can decrypt it. Further if one user's key is found by an intruder all the users are affected. The security level can be increased by keeping a point P (private) on elliptic curve E. The public parameters are the prime number p and a, b defining the elliptic curve $E_p(a,b)$ such that $y^2 = x^3 + ax + b$ with $\gcd(4a^3 + 27b^2, p) = 1$. The RSA algorithm [1] is used to generate the key pair (e,d).

If A and B are two communicating parties then algorithms is described below

(i) A Selects two random numbers X_A, R_A in E_p and a point P_A on elliptic curve.

(ii) B Selects two random numbers X_B, R_B in E_p and a point P_B on elliptic curve.

(iii) A sends $G_A = X_A P_A$ to B.

(iv) B Sends $G_B = X_B P_B$ to A.

(v) A Sends $S_A = R_A G_B$ to B.

(vi) B Sends $S_B = R_B G_A$ to A.

(vii) A Computes the session key $Pub = e(S_A + S_B)$.

(viii) B computes the session key $Pub = e(S_A + S_B)$.

(ix) The private key will be $Sec = d(S_A + S_B)$.

VI. CONCLUSION AND FUTURE WORK

Elliptic curve cryptographic techniques are one of the best ways of sending-receiving encrypted messages or keeping encrypted data. The security of the proposed authentication protocol is based on ECDSA and ECDH protocol principles and the Schnorr's zero knowledge protocol is used for authentication. The protocol provides a methodology for obtaining high-speed, efficient and scalable implementations of authentication protocols. It also provides the highest strength per bit of any crypto system resulting in faster computations, lower power consumption and money. Also it is resistant against man-in-the middle attack. The use of ECC will decrease the storage requirements for the execution of the protocol. The use of ECC with compression techniques will further reduce the storage requirements and it is highly recommended for the future developments with regard to the network security protocols. The proposed protocol is a step in this direction. The future work on the present paper will be implementing the protocol in real-time and providing the performance results.

ACKNOWLEDGMENT

The author is very grateful to the reviewers for making fruitful suggestions that greatly improved the paper in the present form.

REFERENCES

1. R. Afreen and S.C. Mehrotra, A review on elliptic curve cryptography for embedded systems, International Journal of Computer Science and Information Technology, Vol. 3, No. 3, June 2011.
2. ANSI X9.62, Public Key Cryptography for the Financial Services Industry, The Elliptic Curve Digital Signature Algorithm, 1999.
3. M. Aydos, B. Sunar and C. K. Koc, An elliptic curve cryptography based authentication and key agreement protocol for wireless communication, Proceedings of the 2nd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, Dallas, Texas, 1-12, 1998.
4. K. Chalkias, G. Filiadis and G. Stephanides, Implementing authentication protocol for exchanging encrypted messages via an authentication server based on ECC with ElGamal's algorithm, World Academy of Science Engineering and Technology, 7, 137-142, 2005.
5. N. Constantinescu, Authentication protocol based on elliptic curve cryptography, Annals of the University of Craiova, Mathematics and Computer Science Series, Volume37(2), 2010, 83-91.
6. S. Goldwasser, S. Micali and C. Rackoff, The knowledge complexity of interactive proof systems, Society for Industrial and Applied Mathematics, Journal on Computing, vol. 18, no. 1, 186-208, 1989.
7. D. R. Hankerson, A. J. Menezes and S.A. Vanstone, Guide to Elliptic Curve Cryptography, Springer-Verlag, 2003.
8. IEEE P1363, Standard Specifications for Public Key Cryptography, Draft Version 7, Sep, 1998.
9. Introduction from http://www.device forge.com /articles/ AT4234154468.html.
10. V. S. Iyenger, Novel elliptic curve scalar multiplication algorithms for faster and safer public-key cryptosystems, International Journal on Cryptography and Information Security, Vol. 2, No. 3, September 2012.
11. N. Koblitz, Elliptic curve cryptosystem, Mathematics of computation, 48:203-209, 1987.
12. N. Koblitz, A course in Number Theory and Cryptography, New York, NY: Springer-Verlag, Second edition, 1994.
13. K. Mangipudi, N. Malneedi and R. Katti, Attacks entific papers and solutions on Aydos-Savas-Koc's wireless authentication protocol, Proceedings of 2004 IEEE Global Telecommunications conference, 2229-2234, 2004.
14. A. J. Menzes, Elliptic Curve Public Key Crypto System, Boston, MA: Kluwer Academic Publishers, 1993.
15. V. Miller, Uses of elliptic curves in cryptography, Crypto 1985, LNCS 218: Advances in Cryptography, Springer-Verlag, 1986.
16. NIST/ U. S. Dept. of Commerce, Digital Signature Standards (DSS), FIPS Publication 186-2 Feb, 2000.
17. NIST, Special Publication 800-57: Recommendation for key management Part 1: General Guideline, Draft Jan, 2003.
18. M. Rosing, Implementing ECC, Manning Publications Co., 1999.
19. C. P. Schnorr, Efficient signature generated by smart cards, Journal of Cryptography, 4, no.3, 161-174, 1991.
20. W. Stallings, Crytography and Network Security, Prentice Hall, Third Edition, 2003.
21. L. Yongliang, W. Gao, H. Yao and X. Yu, Elliptic curve cryptography based wireless authentication protocol, International Jouranl of Network Security, 5, no.3, 327-337, 2007.

AUTHOR PROFILE

Author's Profile-The author is presently working as an assistant professor in the Department of Mathematics and Statistics, Gurukul Kangri Vishwavidyalay, Haridwar (UK), for more than 10 years. He possesses Ph. D. as well as M. Phil. degree in Mathematics. He has also qualified CSIR-UGC NET examination. He is presently doing research work in the field of "Cryptography and Network Security" as well as in the field of "Approximation Theory". He has published four research papers in various Journals.