International Journal of Advanced Research and Development
ISSN: 2455-4030, Impact Factor: RJIF 5.24
www.advancedjournal.com
Volume 2; Issue 4; July 2017; Page No. 50-55

A mutual authentication protocol based on elliptic curve cryptography

Manoj Kumar

Department of Mathematics and Statistics, Gurukula Kangri Vishwavidyalaya, Haridwar, Uttrakhand, India

Abstract
Elliptic curve cryptography (ECC) is an alternative approach for RSA scheme. 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. In the present paper we proposed a mutual authentication scheme based on ECC and zero knowledge proofs. The required computational time of our scheme is generally similar to other existing schemes.

Keywords: elliptic curves, cryptosystem, authentication protocols, public key cryptography, zero-knowledge proofs, finite fields

1. Introduction

Recent years have brought a host of cryptographic schemes that make use of zero knowledge proofs on elliptic curves. In the world of elliptic curve cryptography [14], the zero knowledge proofs were initially considered by Goldwasser, Micali and Rackoff [4]. Zero knowledge protocols are instances of an interactive proof system, where claimant and verifier exchange messages (typically depending on random events). On the other hand, the ECC was first developed independently by Victor Miller [15] and Neal Koblitz [7] 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. A lot of work has been done on ECC [2, 3, 5, 7, 9, 12, 13, 14, 15, 16, 20, 22]. Recently, Kumar and Gupta [10] proposed cryptographic schemes based on elliptic curves over the ring $Z_p[i]$. Very recently, Kumar *et al* [11] studied a novel and secure multi party key exchange scheme using trilinear pairing map based on elliptic curve cryptography. The National Institute of Standards and Technology (NIST) approved ECC for use by the U.S. government [17]. 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.
The detail study of fundamentals of ECC and its mathematical background can be found in [8, 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:
A general equation of 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 $GF(p)$ i.e.

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

where p is a prime number. If the discriminant $\Delta = (4a^3 + 27b^2)(\mathrm{mod}\, p)$ 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

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

where $O \neq (0,0)$.
The following figure is an elliptic curve satisfying the equation

$$y^2 = x^3 - 3x + 3$$

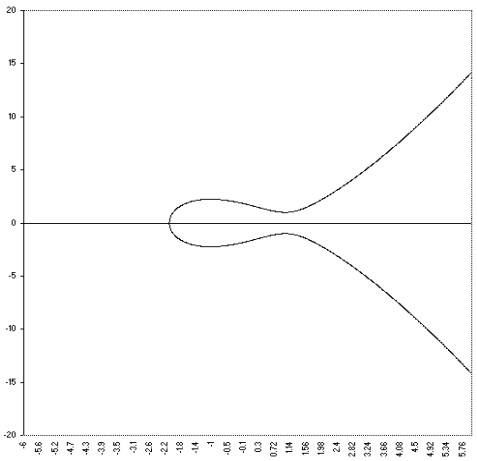

Fig 1: Elliptic curve over R^2

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 [5, 20] 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[1] with further recommendations for the

selection of underlying finite fields and elliptic curves, the revised standard is called FIPS 186-2[17]. 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 $GF(2^m)$, $m = 163, 233, 283, 409, 571$. Table-1 shows NIST guidelines [18] on choosing computationally equivalent symmetric and public key sizes.

Table 1: Equivalent Key Sizes (in bits)

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

2. 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 [6]. Generally ECDSA involves the three parts:

2.1 ECDSA Key Generation

The user A follows three steps

1. Choose a random integer d_1 such that $2 \le d_1 \le n-2$.
2. Calculate $Q = d_1 P$.
3. d_1 and (E, P, n, Q) are the private and public keys of user A, respectively.

2.2 ECDSA Signature Generation

The user A signs the message m using the five steps

1. Choose a random integer d_2 such that $2 \le d_2 \le n-2$.
2. Calculate $d_2 P = (x_1, y_1)$ and $r = x_1 \bmod n$. If $r = 0$ then go to step **1**.
3. Calculate $d_2^{-1} \bmod n$.
4. Calculate $s = d_2^{-1}(H(m) + d_1 r) \bmod n$. If $s = 0$ then go to step **1**.
5. (r, s) is the signature for the message m.

2.3 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

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

2. Calculate $u_1 = H(m)c \bmod n$ and $u_2 = r c \bmod n$.

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

4. 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 [6].

3. 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. The concept of zero knowledge was first introduced by Goldwasser *et al.* [4].

The aim of the zero knowledge is to prove the knowledge of a secret without revealing it. Schnorr's protocol [21] is one of the most popular zero knowledge protocol. Let p and q be two primes number such that q divides $(p-1)$. Let $g \ne 1$ 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 [21].

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

1. If α is the prover's secret information then user makes public $Z = \alpha P$.
2. The prover picks a random number r and sends $X = rP$ to the verifier.
3. The verifier picks a random number e and sends it to the prover.
4. The prover computes $Y = (\alpha e + r) \bmod n$ and sends it to the verifier.
5. The reciver receives y and accepts if $yP + eZ = X$.

3.1 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 takes all responsibility of the group member's verification by providing to both sender and receiver a random session number known as nonce.

3.2 Authentication between User and Server

The authentication between user and server need to be executed in real time. It consists of the four steps

1. U → S : $(\text{Sends})_s\ (X, N_1)$
2. S → U : $(\text{Sends})_U (e, N_1)$
3. U → S : $(\text{Sends})_s\ (y)$
4. 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.

3.3 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

1. A Selects two random numbers n_A, k_A in $GF(p)$ and a point P_A on elliptic curve E_p.
2. B Selects two random numbers n_B, k_B in $GF(p)$ and a point P_B on elliptic curve E_p.
3. A sends $X_A = n_A P_A$ to B.
4. B sends $X_B = n_B P_B$ to A.
5. A sends $Y_A = k_A X_B$ to B.
6. B sends $Y_B = k_B X_A$ to A.
7. A computes the session key $Pub = e(Y_A + Y_B)$.
8. B computes the session key $Pub = e(Y_A + Y_B)$.
9. The private key will be $Sec = d(Y_A + Y_B)$.

4. Security Analysis of Proposed Protocol

As we know that security of classic cryptosystem is based on DLP (discrete logarithm problem), while the security of elliptic curve cryptosystem is based on ECDLP (elliptic curve discrete logarithm problem). According to ECDLP, for given $Q, P \in E_p(a,b)$ such that $Q = kP$ it is very difficult to determine the value of k. In our protocol A sends message $X_A = n_A P_A$ to B. Similarly B sends message $X_B = n_B P_B$ to A. After receiving the message, A calculates $Y_A = k_A X_B$ and sends it to B. Likewise B Calculates $Y_B = k_B X_A$ and sends it to A. After that both A and B Compute the common session key $K = e(Y_A + Y_B)$. Under the assumption that it is very difficult (almost impossible) to determine the elliptic curve discrete logarithm problem, both parties A and B believe that only they can calculate the shared session key K exactly. Further, private key is obtained by multiplication of d and $(Y_A + Y_B)$. In the same time the multiplication provides an increased security so that the protocol will not suffer from the man-in-the middle attack. One can easily verify that protocol has only two pass key agreements. It means no communication overhead will be added. In our protocol session key is computed from the random numbers generated by the two communicating parties every time. If the random number generator or algorithm is good enough then the probability that the same session keys are generated in two runs of our protocol is neglectable. From this point, it is impossible to learn other keys when knowing several session keys. In our protocol, the secret information of both communicating parties is bounded with certificate. Therefore, using its secret information or other secret information it gets through certain manner, the attacker can't impersonate other party besides the owners of secrete information.

5. Comparison

During group communication very sensitive situations arise because all the users have the same keys for one session. If one key is compromised all the other users are compromised. To prevent such type of situation we use ECC which provides the required security for such type of situations. Although, such type of situation can be prevent by using different keys for each user but this approach cannot be follow, because it is more important that all messages can be read by any member of the group. Since all the messages have to be known by all the members of the group, therefore authentication is the most important phase because if an intruder succeeds to sign in all the information can be compromised. As we know that DSA includes many operations and takes much more times for messages authentication therefore we did not use this approach in our protocol. For authentication we use secret knowledge which takes three times fewer steps and provides at least the same level of security. Furthermore to increase the security level of encryption, we use the Schnorr's protocol. Our protocol can be implemented without using ECC. In that case, RSA [19] method will be used for encryption while classical Schnorr's protocol will be used for authentication. Without using ECC, the keys used for encrypting and decrypting the messages, must be comparatively larger to obtain the same level of security. Thus, using ECC our proposed protocol prevents high complexity and it is more secure and efficient for group communication.

6. Conclusions

The proposed 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.

References

1. ANSI X9 62, Public Key Cryptography for the Financial Services Industry, The Elliptic Curve Digital Signature Algorithm, 1999.
2. Chillali A, Elliptic Curve over Ring, International Mathematical Forum. 2011; **6**:1501-1505.
3. Deepthi PP, Sathidevi PS. New stream ciphers based on elliptic curve point multiplication, Computer communications. 2009; 32:25-33.
4. Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof systems, Society for Industrial and Applied Mathematics, Journal on Computing. 1989; 18(1):186-208.
5. Hankerson DR, Menezes AJ, Vanstone SA. Guide to Elliptic Curve Cryptography, Springer-Verlag, 2003.
6. IEEE P1363, Standard Specifications for Public Key Cryptography, Draft Version. 1998, 7.
7. Koblitz N. Elliptic curve cryptosystem, Mathematics of computation. 1987; 48:203-209.
8. Koblitz N. A course in Number Theory and Cryptography, New York, NY: Springer-Verlag, Second edition, 1994.
9. Kumar M. A secure and efficient authentication protocol based on elliptic curve Deffie-Hellman algorithm and zero knowledge property, International Journal of Soft Computing and Engineering. 2013; 3(5):137-142.
10. Kumar M, Gupta P. Cryptographic schemes based on elliptic curves over the ring Zp[i], Applied Mathematics. 2016; 7(3):304-312.
11. Kumar M, Gupta P, Kumar A. A novel and secure multi party key exchange scheme using trilinear pairing map based on elliptic curve cryptography, International Journal of Pure and Applied Mathematics, Article No. AP2016-, 2017; 16(1):31-439.
12. Kumar S, Suneetha C, Chandrasekh A. Encryption of Data Using Elliptic Curve over Finite Fields, International Journal of Distributed and Parallel Systems. 2012; 3(1):301-308.
13. Marin L, Jara A, Gomez AS. Shifting primes: Optimizing elliptic curve cryptography for 16-bit devices without hardware multiplier, Mathematical and Computer Modeling. 2013; 58:1155-1174.
14. Menzes AJ. Elliptic Curve Public Key Crypto System, Boston, MA: Kluwer Academic Publishers, 1993.
15. Miller V. Uses of elliptic curves in cryptography, Crypto LNCS 218: Advances in Cryptography, Springer-Verlag. 1985,1986.
16. Moosavi SR, Nigussie E, Virtanen S, Isoaho J. An elliptic curve based mutual authentication scheme for RFID implant systems, Procedia Computer Science. 2014; 32:198-206.
17. NIST/ US. Dept. of Commerce, Digital Signature Standards (DSS), FIPS Publication. 2000. 186-2.
18. NIST, Special Publication 800-57: Recommendation for key management Part 1: General Guideline, 2003.
19. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public key cryptosystems, Communications of the ACM. 1978; 21:120-126.
20. Rosing M. Implementing ECC, Manning Publications Co, 1999.
21. Schnorr CP, Efficient signature generated by smart cards, Journal of Cryptography, 1991; 4(3):161-174.
22. Srivastava K. and Nand G., Elliptic Curves for Data Provenance, Procedia Computer Science. 2015; 45:470-476.