JOURNAL OF COMPUTER ENGINEERING & TECHNOLOGY (JCET)

ISSN XXX-XXX (Print)
ISSN XXX-XXX (Online)
Volume 1, Issue 1, July-December (2013), pp.10-17

JCET
© I A E M E

A SURVEY OF ELLIPTIC CURVE CRYPTOGRAPHY USING CHARACTER CONVERSION METHOD

Arun Kumar Gandhi, Dr. S.S.Tyagi

ABSTRACT

In the recent years, elliptic curve cryptography (ECC) has gained widespread exposure and acceptance and has already been include in many security standards. It is widely recognized that data security will play a central role in the design of future IT system. Elliptic key cryptography recently gained lot of attention in industries when we talk about security. This paper discusses the basic operation how ECC is more secure than other public key cryptosystems and also how security can be enhanced in ECC by using Character-Conversion-Method.

Keywords: Elliptic Curve Cryptography, Security

I. INTRODUCTION

Encryption is the process of transforming plaintext data into cipher text in order to conceal its meaning and so preventing any unauthorized recipient from retrieving the original data. Hence, the main task of encryption is to ensure secrecy. Companies usually encrypt their data before transmission to ensure that the data is secure during transit.[9] The encrypted data is sent over the public network and is decrypted by the intended recipient. There are many encryption algorithms are developed and widely used for information security. They can be categorized into symmetric (private) and asymmetric (public) keys encryption.

Symmetric keys encryption only uses one key to encrypt and decrypt data. The key should be distributed before transmission between entities. Keys play a very important role because if weak key is used in algorithm then everyone may decrypt the data. Strength of Symmetric key encryption depends on the size of used key.[13]

For the same algorithm, encryption using longer key is harder to break than the one done using smaller key. There are many examples of strong and weak keys of cryptography algorithms like RC2, DES, 3DES, RC6, Blowfish, and AES. RC2 and DES use one 64-bit key. Triple DES (3DES) uses three 64-bits keys while AES uses various (128,192,256) bits keys. Blowfish uses various (32-448) key. RC6 uses various (128,192,256) bit keys where default is 128 bits. [1-3].

Asymmetric key encryption is used to solve the problem of key distribution. In Asymmetric keys, two keys are used: private and public keys. Public key is used for encryption and private key is used for decryption (E.g. RSA and Digital Signatures). Public key is known to the public while private key is known only to the user. There is no need for distributing them prior to transmission. However, public key encryption is based on mathematical functions, computationally intensive and is not very efficient for small mobile devices such as cell phone, PDA, and so on. In Asymmetric key encryption, private key and public key are used. Public key is used for encryption and private key is used for decryption (E.g. RSA, Digital Signatures and ECC). [11]

In public key cryptography each user or the device taking part in the communication have a pair of keys, a public key and a private key, and a set of operations associated with the keys to do the cryptographic operations [9]. Only the particular user/device knows the private key whereas the public key is distributed to all users/devices taking part in the communication. Since the knowledge of public key does not compromise the security of the algorithms, it can be easily exchanged online.

The rest of this paper is organized as follows. We review the the asymmetric algorithms in section II describe their operations in section III & proposal for enhance the security level of Elliptic Key Cryptography from various attacks in section IV.

II. PUBLIC KEY CRYPTOSYSTEMS

In this section, we have an overview and cryptanalysis public key cryptosystems RSA, DH and ECC.

A. THE RSA CRYPTOSYSTEM

We begin with a brief review of the RSA scheme. It is based on the hard mathematical problem of integer factorisation, i.e. given a number that is the product of two large prime numbers, factorise the number to find the primes.

RSA Key generation

Given the ―public exponent e, generate two large prime numbers p and q, such that (p-1) and e have no common divisor greater than 1 and (q-1) and e have no prime divisor greater than 1. Let n = pq, the product of p and q. Solve (for d) the equation ed = 1 mod (p-1) (q-1). The public key is the pair of numbers {n,e} and the private key is the pair {n,d}.

RSA Encryption and Decryption

Encryption uses a public key, so that the ciphertext corresponding to plaintext m is $c = m^e \pmod n$. Decryption uses the corresponding private key, so $m = c^d \pmod n$.

RSA – and the Integer factorization problem

As we know RSA is the first asymmetric cryptosystem to have seen widespread use is also one of the most accessible illustrations of this principle in action. RSA gets its security from the difficulty of factoring very large numbers. The difficulty of getting the plaintext message back from the ciphertext and the public key is related to the difficulty of factoring a very large product of two prime numbers [4].

As an illustration of this: imagine you were to take two very large prime numbers — say, 200 digits long, and were then to multiply them together. Now the result you get has two particular properties:

(i) It is very large (about 400 digits in length),
(ii) It has two, and exactly two factors, both prime numbers the two primes you just multiplied together

You can easily give the two prime numbers from which you start find the product. But finding the primes given only the product is more difficult. So much more, in fact, that once the numbers get adequately large, it is almost impossible to find them you simply cannot assemble enough computing power to do so. So the multiplying of two large prime numbers together is the (relatively) easy forward function in this asymmetric algorithm. Its inverse the factor finding operation is considerably more difficult, and in practical terms, it‘s intractable [3, 4].

The RSA system employs this fact to generate public and private key pairs. The keys are functions of the product and of the primes. Operations performed using the cryptosystem is arranged so that the operations we wish to be tractable require performing the relatively easy forward function — multiplication. Conversely, the operations we wish to make difficult — finding the plaintext from the ciphertext using only the public key — require performing the inverse operation — solving the factoring problem.

The Diffie-Hellman/DSA Cryptosystems and the

Discrete Logarithm Problem

Diffie-Hellman (DH) along with the Digital Signature Algorithm (DSA) based on it is another of the asymmetric cryptosystems in general use. ECC, in a sense, is an evolved form of DH. So to understand how ECC works, it helps to understand how DH works first. DH uses a problem known as the discrete logarithm problem as its central, asymmetric operation. The discrete log problem concerns finding logarithm of a number within a finite field arithmetic system. Prime fields are fields whose sets are prime - that is, they have a prime number of members. These are of particular interest in asymmetric cryptography because, over a prime field, exponentiation turns out to be a relatively easy operation, while the inverse computing the logarithm is very difficult. To generate a key pair in the discrete logarithm (DL) system, therefore, you calculate:

$y = (g^x) \bmod p$ …….. (1)

Where p is a large prime — the field size x and g are smaller than p. y is the public key. x is used as the private key. In DH, again, the operations we wish to make ‗easy‘, or tractable, we harness to the operation in the field which is (relatively) easy exponentiation. So encryption using the public key is an exponentiation operation. Decryption using the private key is as well. Decryption using the public key, however, would require performing the difficult inverse operation solving the discrete logarithm problem. The discrete logarithm problem, using the values in the equation above, is simply finding x given only y, g and p.

Expanding that thought slightly: someone has multiplied g by itself x times, and reduced the result into the field (performed the modulo operation) as often as necessary to keep the result smaller than p. Now, knowing y, g and p, you‘re trying to find out what value of *x* they used. It turns out that for large enough values of p, where p is prime, this is extraordinarily difficult to do much more difficult than just finding y from g, x and p. If you grasp what‘s going on in the operations above, you‘re now in a position to grasp the basic math behind the DSA and discrete logarithm systems.

And, by extension, you also understand some of the principles behind ECC. ECC — as we‘ll discuss in greater detail a little later — also uses a discrete log problem in a finite group. The difference is that ECC defines its group differently. And it is, in fact, the difference in how the group is defined and particularly how the mathematical operations within the group are defined that give ECC its greater security for a given key size.

B. ELLIPTIC KEY CRYPTOGRAPHY

Elliptic Curve Cryptography (ECC) is a public key cryptography. In public key cryptography each user or the device taking part in the communication generally have a pair of keys, a public key and a private key, and a set of operations associated with the keys to do the cryptographic operations [6]. Only the particular user knows the private key whereas the public key is distributed to all users taking part in the communication.

Some public key algorithm may require a set of predefined constants to be known by all the devices taking part in the communication. _Domain parameters'in ECC is an example of such constants. Public key cryptography, unlike private key cryptography, does not require any shared secret between the communicating parties but it is much slower than the private key cryptography.

The mathematical operations of ECC is defined over the elliptic curve $\mathbf{y^2 = x^3 + ax + b}$ where $\mathbf{4a^3 + 27b^2 \neq 0}$. Each value of the _a'and _b'gives a different elliptic curve. All points (x, y) which satisfies the above equation plus a point at infinity lies on the elliptic curve. The public key is a point in the curve and the private key is a random number. The public key is obtained by multiplying the private key with the generator point G in the curve [7,8]. The generator point G, the curve parameters _a'and _b', together with few more constants constitutes the domain parameter of ECC. One main advantage of ECC is its small key size. A 160-bit key in ECC is considered to be as secured as 1024-bit key in RSA.

Discrete Logarithm Problem

The security of ECC depends on the difficulty of Elliptic Curve Discrete Logarithm Problem. Let P and Q be two points on an elliptic curve such that kP = Q, where k is a scalar. Given P and Q, it is computationally infeasible to obtain k, if k is sufficiently large. k is the discrete logarithm of Q to the base P. Hence the main operation involved in ECC is point multiplication. I.e. multiplication of a scalar k with any point P on the curve to obtain another point Q on the curve.

III. ECC FOUNDATIONS

Scalar Point Multiplication

The dominant operation in ECC cryptographic schemes is scalar point multiplication. This is the operation which is the key to the use of elliptic curves for asymmetric cryptography the critical operation which is itself fairly simple, but whose inverse (the elliptic curve discrete logarithm problem defined below) is very difficult. ECC arranges itself so that when you wish to performance operation the cryptosystem should make easy encrypting a message with the public key, decrypting it with the private key, the operation you are performing is point multiplication [3].

Point multiplication is simply calculating *kP*, where *k* is an integer and *P* is a point on the elliptic curve defined in the prime field. In point multiplication a point P on the elliptic curve is multiplied with a scalar k using elliptic curve equation to obtain another point Q on the same elliptic curve. I.e. KP=Q. Point multiplication is achieved by two basic elliptic curve operations.

- Point addition, adding two points J and K to obtain another point L i.e., L = J + K.
- Point doubling, adding a point J to itself to obtain another point L i.e. L = 2J.

Point addition

Point addition is the addition of two points J and K on an elliptic curve to obtain another point L on the same elliptic curve. Consider two points J and K on an elliptic curve

as shown in figure (a). If K ≠ -J then a line drawn through the points J and K will intersect the elliptic curve at exactly one

More point –L. The reflection of the point –L with respect to x-axis gives the point L, which is the result of addition of points J and K. Thus on an elliptic curve L = J + K. If K = -J the line through this point intersect at a point at infinity O. Hence J + (-J) = O. This is shown in figure (b). O is the additive identity of the elliptic curve group. A negative of a point is the reflection of that point with respect to x-axis.

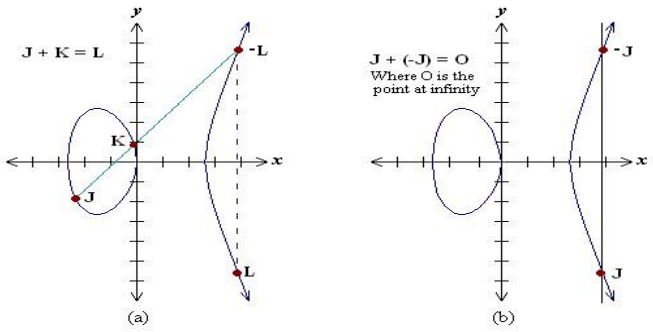

Figure 1: Addition of two points

O is the additive identity of the elliptic curve group. A negative of a point is the reflection of that point with respect to x-axis.

Point doubling

Point doubling is the addition of a point J on the elliptic curve to itself to obtain another point L on the same elliptic curve.

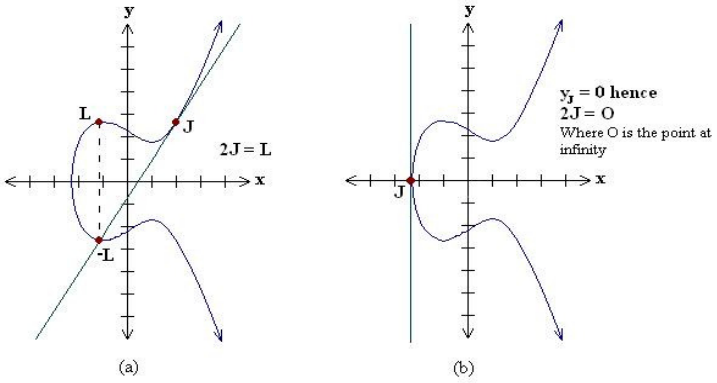

Fig 2: Doubling of points

To double a point J to get L, i.e. to find L = 2J, consider a point J on an elliptic curve as shown in figure (a). If y coordinate of the point J is not zero then the tangent line at J will intersect the elliptic curve at exactly one more point –L. The reflection of the point –L with respect to x-axis gives the point L, which is the result of doubling the point J.

Thus L = 2J. If y coordinate of the point J is zero then the tangent at this point intersects at a point at infinity O. Hence 2J = O when yJ = 0. This is shown in figure (b)

Elliptic Curve Example:

Let the equation of the curve is $y^2 \bmod p = x^3 + ax + b \bmod p$ Inputs: a, b, p (is key of the ECC algorithm). Choose two non-negative integers a, b and a large prime number such that $4a^3 + 27b^2 \bmod p! = 0$. For Example, the following figure (fig 1) shows the elliptic curve, $y^2 \bmod 41 = x^3 + x + 1 \bmod 41$. Here points P, Q lie on the curve and P+Q gives another point that lie on the line that connects P and Q as shown in the fig 1 below.

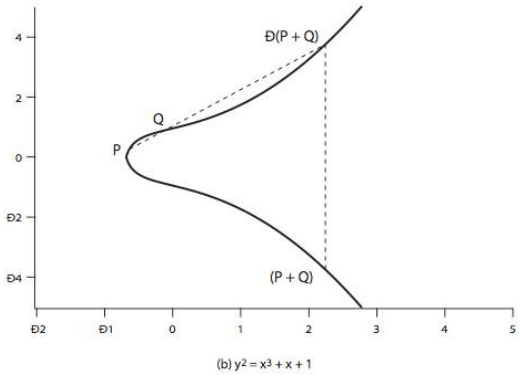

Figure: 3

The set of points on the above curve are

{

[0 1],[0 40] ,[3, 20], [3, 21], [5, 7], [5, 34], [6, 10],[6, 31] ,[7, 8], [7, 33] ,[9, 1],[9, 40],

[11,20],[11,21],[19,14],[19,27],[24,14],[24,27],

[27,20],[27,21],[28,13],[28,28],[31,4],[31,37], [19 5],[19 18]

}

Multiplication of a point with a positive integer k is

Defined as the sum of copies of P, k times. This operation is called Point Multiplication in ECC. So 3P=P+P+P. [9]

The above points from the Group i.e.Ep (a,b) . Each X and Y coordinates ranges between 0 and 41. The addition of the two points on the curve and the inverse of a point on the curve are defined in the field using modular arithmetic. The point at the infinity is identity point on the curve.

IV. SECURITY IN ECC

As we discussed earlier that the security of ECC depends on the difficulty of Elliptic Curve Discrete Logarithm Problem Now for our assumption USER A and USER B communicate each other and sending message M securily.

1. Assume that Xp = 11 and Yp = 20 Where Xp and Yp are the base - points
2. Also Xm = 24 and Ym = 27

Where Xm & Ym are the message points

3. User A sends kap and l to User B.

Where kap is (35, 5) of User A and l = any random integer from 0 to 15.

4. Now User B computes the following points.

(i) l * P is (28,28)

(ii) l*kap is (5,7)

(iii) l*kap+M is (3,21) where M is the message point.

5. Now User B sends 2 points
C1 = l*P is (5, 7)
C2 = (l*ka*P+M) to User A.
C2 – C1 = M which is the message point.
M is (24, 27)

1. Character - Conversion – Method : The Proposed Method

Now for our assumption USER A and USER B communicate each other and sending message M securely.

1. Assume that Xp and Yp are the base points.
2. And also Xm & Ym are the message points.
3. User A sends kap and l to User B. Where kap is public key of User A and l = any random integer from 0 to 15.

Suppose we take a character L. DEFINE: Char L;
l = random (15) + 3;
l = l + 64;
L = l;
Printf (—L = % c, where L is an character □, L); l = L – 64;

4. Now User B computes the following points.
i) l * P
ii) l*kap
iii) l*kap+M , where M is the message point.

5. Now User B sends 2 points
i) C1 = l*P
ii) C2 = (l*ka*P+M) to User A.
iii) C2 – C1 = M which is the message point.

2. Encoding and Decoding through Message - Point

Let us suppose a text file or any image has to be encrypted with the message M. All the points on the elliptic curve can be directly mapped to an ASCII value, select a curve on which we will get a minimum of 128 points, so that we fix each point on the curve to an ASCII value. For example, _EGYPT‘ can be written as sequence of ASCII characters that is _69‘ _71‘ _89‘ _80‘ _84‘. We can map these values to fixed points on the curve. It will improve the security of our message. The above two methods are useful to prevent our secret information from various attackers.

V. CONCLUSION

The above two methods are very efficient in terms of security when we talk about — brute-force □ attack and — man-in the-middle-attack □. In first method integer value is replaced by a character and every time a character takes less space as compared to an integer. This will results in the memory savings, fast computation and low processing overhead. And in second method the original message M encrypted with any text. It also results in the enhancement of security level of ECC. And this, in the end, is the reason ECC is a stronger option than the RSA and discrete logarithm systems for the future. And this, in the end, is

why ECC is such an excellent choice for doing asymmetric cryptography in portable, necessarily constrained devices right now. As an example: as of this writing, a popular, recommended RSA key size for most applications is 2,048 bits. For equivalent security using ECC, you need a key of only 224 bits.

REFERENCES

[1] Darrel Hankerson, Julio Lopez Hernandez, Alfred Menezes, Software Implementation of Elliptic Curve Cryptography over Binary Fields, 2000.

[2] M. Brown, D. Hankerson, J. Lopez, A. Menezes, Software Implementation of the NIST Elliptic Curves Over Prime Fields, 2001.

[3] Certicom, Standards for Efficient Cryptography, SEC 1: Elliptic Curve Cryptography, Version 1.0, September 2000.

[4] Certicom, Standards for Efficient Cryptography, SEC 2: Recommended Elliptic Curve Domain Parameters, Version 1.0, September 2000

[5] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996

[6] I. F. Blake, G. Seroussi, and N. P. Smart. Elliptic curves in cryptography, volume 265 of London Mathematical Society Lecture Note Series.Cambridge University Press, Cambridge, 2000. Reprint of the 1999 original.

[7] D. Shanks. Class number, a theory of factorization, and genera. In 1969 Number Theory Institute (Proc. Sympos. Pure Math., Vol. XX, State Univ. New York, Stony Brook, NY, 1969), pages 415 440. Amer. Math. Soc., Providence, RI, 1971.

[8] J. H. Silverman. The arithmetic of elliptic curves, volume 106 of Graduate Texts in Mathematics. Springer-Verlag, New York, 1986.

[9] W. Trappe and L.Washington. Introduction to cryptography with coding theory, (2nd Ed.). Prentice Hall, Upper Saddle River, NJ, 2006.

[10] L. C. Washington. Elliptic curves. Number theory and cryptography, (2nd ed.). Chapman & Hall/CRC, New York, NY, 2008. [11] A. S. Tanenbaum, "Modemn Operating Systems", Prentice Hall, 2003.[3] M. J. B. Robshaw, "Block Ciphers", Technical Report, RSALaboratories, Number TR - 601, July 1994.

[12] H.M. Heys and S.E. Tavares, "On the Security of the CAST Encryption Algorithm," Proceedings of the Canadian Conference on Electrical and Computer Engineering, Halifax, Nova Scotia, Sep 1994, pp. 332-335.

[13] R. Rivest, ―The encryption algorithm, in Fast Software Encryption, ser. LNCS, vol. 1008. Springer-Verlag, 1995, pp. 86–96.

[14] M.A. Viredaz and D.A. Wallach, ―Power Evaluation of a Handheld Computer: A Case Study, WRL Research Report, 2001/1.

[15] P. Ruangchaijatupon, P. Krishnamurthy, "Encryption and Power Consumption in Wireless LANs-N,‘‘ The Third IEEE Workshop on Wireless LANs - September 27-28, 2001-Newton, Massachusetts.

Arun Kumar Gandhi received B.Tech degree in Computer Science & Engineering from Kurukshetra University in 2007 and M.Tech. in CSE from Manav Rachna International University,Faridabad in 2011 .Presently, he is working as Assistant professor in Computer Engineering department in Manav Rachna International University, Faridabad. His area of interest is Cryptography & Network Security.

Prof Dr. S.S Tyagi is currently working as Professor and Head, Department of CSE in Manav Rachna International University, Faridabad, India. He did his P.hD from Kurkshetra University, M.E from BITS Pilani , and B.Tech, from Nagpur University in Computer Sc. and Engineering. He has published many papers in national andinternational Journals and guiding Ph.D Scholars.