A New Approach for Secured Transition using Prime Field Elliptic Curve Cryptography System

Muhammad Firoz Mridha
Department of Computer Science
Stamford University Bangladesh
Dhaka, Bangladesh

ABSTRACT
The demands of secured electronic transactions are increasing rapidly. Prime Field Elliptic curve cryptosystems (PFECC) are becoming most popular because of the reduced number of key bits required in comparison to other cryptosystems. PFECC is emerging as an attractive alternative to traditional public-key cryptosystems. PFECC offers equivalent security with smaller key sizes resulting in faster computations, lower power consumption, as well as memory and bandwidth savings. While these characteristics make PFECC especially appealing for small devices, they can also alleviate the computational burden on secure web servers.

Keywords -Prime Field Elliptic curve cryptosystems, public key cryptosystems, RSA, Modular Arithmetic, Key Distribution center.

1. INTRODUCTION
Secure transaction is an intrinsic requirement of today's world of on-line transactions. Whether exchanging financial, business or personal information, people want to know with whom they are communicating (authentication) and they wish to ensure that the information is neither modified (data integrity) nor disclosed (confidentiality) in transit. The Secure Sockets Layer (SSL) protocol [1] is the most popular choice for achieving these goals.

Elliptic curves were first proposed as a basis for public key cryptography in the mid 1980s independently by Koblitz [2] and Miller [3]. Elliptic curves provide a public key cryptosystem based on the difficulty of the elliptic curve discrete logarithm problem (defined later in this section), which is so called because of its similarity to the discrete logarithm problem (DLP) over the integers modulo a prime p. This similarity means that most cryptographic procedures carried out using a cryptosystem based on the DLP over the integers modulo can also be carried out in an elliptic curve cryptosystem. Another benefit of ECC is that they can use a much shorter key length than other public key cryptosystems to provide an equivalent level of security. For example, 160 bit elliptic curve cryptosystems (ECC) are believed to provide about the same level of security as 1024 bit RSA [7, p.51]. Also, the rate at which ECC key sizes increase in order to obtain increased security is much slower than the rate at which integer based discrete logarithm (dl) or RSA key sizes must be increased for the same increase in security. ECCs can also provide a faster implementation than RSA or dl systems, and use less bandwidth and power [4].

This paper is organized as follows: the literature review related to the ECC in Section 2 and 3 while the existing system, proposed model and implementation are structured in detailed in Section 4, Section 5 and Section 6.

2. PUBLIC KEY CRYPTOGRAPHY METHODS
The origins of public-key cryptography stem from a paper published in *1968* by Wilkes [5]. It describes a new *one-way cipher* used by R. M. Needham to verify passwords on a computer without storing any information that could be used for an intruder to impersonate a legitimate user. In Needham's system, when the user first sets his password, or whenever he changes it, it is immediately subjected to the enciphering process, and it is the enciphered form that is stored in the computer. Whenever the password is typed in response to a demand from the supervisor for the user's identity to be established, it is again enciphered and the result compared with the stored version. It would be of no immediate use to a would-be malefactor to obtain a copy of the list of enciphered passwords, since he would have to decipher them before he could use them. For this purpose, he would need access to a computer and even if full details of the enciphering algorithm were available, the deciphering process would take a long time.

Purdy [6] gave the first detailed description of such a *one-way function* in 1974. In his paper, he let the original passwords and their enciphered forms be the integers modulo a large prime *p*, that is in *Zp*, and the one-way function be a map from *Zp* to *Zp*. The map is given by a polynomial *f(x)* which is easy to evaluate by computer but not feasible to calculate the inverse. As an example, Purdy used $p = 2^{64}$-59 and

$$f(x) = x^{2^{24}+17} + a_1 x^{2^{24}+3} + a_2 x^3 + a_3 x^2 + a_5,$$

Where the coefficients a_i were arbitrary 19-digit integers.

Public-key cryptography was conceived by Diffie and Hellman [7] in 1976 when they described a protocol whereby two people, Alice and Bob, can derive and securely share private information over an insecure communications channel. This information can then be used as their key in a private-key cryptosystem such as DES.

3. ECC BASICS
Elliptic curve cryptosystems (ECC) are based on the group of points on an elliptic curve over a finite field. They rely on the difficulty of finding the value of a scalar, given a point and that scalar multiple of that point. This corresponds to solving the DL problem. However, it is more difficult to solve the elliptic curve DL problem than its original counterpart. Thus, elliptic curve

cryptosystems provide equivalent security as the existing public-key cryptosystems, but with much smaller key lengths. Therefore, they have smaller bandwidth and memory requirements which makes them extremely desirable for embedded systems such as smart cards, as well as use on personal computers and workstations. In addition, another benefit is that each user may select a different curve E even though the underlying field K remains the same for all users. Thus, the hardware, which depends on the field K, remains the same and the curve E can be changed periodically for extra security.Traditionally, ECC has been developed over finite fields which have either prime order, or characteristic 2. Prime fields have the advantages of using integer operations whereas binary extension fields, that is, curves with characteristic 2, can use the exclusive or and shift operations instead of addition and multiplication respectively, which lead to significant improvements in speed.

Elliptic curve cryptosystems are more secure for a given bandwidth, for the same security, they require a much smaller bandwidth which makes them ideal for embedded processor applications such as smart cards. The arithmetic processor on a smart card is generally restricted to an area of approximately $20mm^2$. For *512-bit RSA* encryption, the chip involved has about *50,000* gates and the chip to perform the arithmetic in the field F_2^{593}, which is necessary for this cryptosystem, has approximately 90,000 gates which is given in [8]. An elliptic curve cryptosystem with greater security and a bandwidth of 200 bits would only require arithmetic in the field F_2^{200} which corresponds to about 15,000 gates [9] and would only occupy less than 20% of the 20mm2 assigned to the processor. In fact, an entire elliptic cryptosystem over F_2^{255} can be fabricated and use up less than 4% of the area for a smart card processor.

Table 1: Analogies between Discrete Logarithm and ECC

Setting	Discrete Logarithm $GF(q)^*$	Elliptic Curve Curve E over $GF(q)$
Basic Operation	multiplication in $GF(q)$	addition of points
Main Operation	exponentiation	scalar multiplication
Base Element	generator g	base point G
Base Element Order	prime r	prime r
Private Key	s (integer modulo r)	s (integer modulo r)
Public Key	w (element of $GF(q)$	w (point on E)

The elliptic curve is defined by a Diophantine equation, slightly different and more complicated than the circle: $y^2+axy+by=x^3+cx^2+dx+e$, where x and y are variables and a, b, c, d and e are constants.

For our purpose, it is sufficient to limit ourselves to equations of the form: $y^2=x^3+ax+b$. However, x, y, a, and b are not necessarily real numbers instead they may be values from any fields. For cryptographic purposes we always use a finite field, that is x, y, a, and b are chosen from a finite set of distinct values.

3.1. Elliptic Curve Groups over Finite Fields

The finite field F_p uses the numbers from *0* to *p-1* and computations end by taking the reminder on division by *p*. For example, in F_{23} the field is composed of integers from *0* to *22*, and any operation within this field will result in an integer also between *0* and *22*.

An elliptic curve with the underlying field of F_p can formed by choosing the variables *a* and *b* within the field of F_p. The elliptic curve includes all points *(x,y)* which satisfy the *elliptic curve* equation modulo *p* (where *x* and *y* are numbers in F_p).

For example: $y^2 \bmod p = x^3 + ax + b \bmod p$ has an underlying field of F_p if *a* and *b* are in F_p). If $x^3 + ax + b$ contains no repeating factors (or, equivalently, if $4a^3 + 27b^2 \bmod p$ *is not 0*), then the elliptic curve can be used to form a group. An elliptic curve group over F_p consists of the points on the corresponding elliptic curve, together with a special point *O* called the point at infinity. There are finitely many points on such an elliptic curve.

3.2.Modular Arithmetic

Modulus operation returns the reminder after integer division. It creates equivalency classes. For example, 5 MOD 3 = 2 MOD 3; here the equivalency classes for 2 MOD 3: {…... -1, 2, 5, 8, 11…}.

Operations in modular arithmetic:

- Addition: P+Q MOD R = (P+Q) MOD R = (P MOD R) + (Q MOD R) MOD R.
- Multiplication: P×Q MOD R=(P×Q) MOD R = (P MOD R) × (Q MOD R) MOD R.
- Subtraction (Addition of negation): P-Q MOD R = P+(-Q) MOD R.
  (Addition of negation can be calculated by: -M MOD N=(|M| MOD N)-N.)
- Division (Inverse multiplication): P/Q MOD R=P×Q_i MOD R, if Q×Q_i MOD R=1.
  (Inverse can be found by using the Extended Euclidian Algorithm.)

3.3. Prime Field ECC for Message Transaction

In public key cryptography, key length is an important factor. It is very much desirable to keep smaller key size but achieve greater security. This paper proposes a better technique on *Prime Field Elliptic Curve Cryptography (PFECC) for Electronic Message Transaction,* which is expected to become the next generation public key cryptography. For two parties A and B, a secure message transaction technique has been designed and developed through the *proposed elliptic curve cryptography* based on *prime field* computation. The complexity of the process has also been calculated and found better than *conventional ECC* and *RSA*, widely used public key cryptography. The *proposed PFECC* holds greater flexibility in choosing cryptographic system. Compared to *RSA, PFECC* requires smaller key size for an equivalent amount of security achievement and compared to *conventional PFECC, proposed PFECC* achieves greater security for equivalent key size. Thus *PFECC* devices require less storage, less power, less memory, less time and sometime less bandwidth.

3.4. Prime Field ECC for Session Key Distribution

Frequent key changes are very much desirable for secure electronic communications in symmetric key encryption. Because it is needed to limit the amount of data compromised if an intruder or attacker learns the communicating keys. In public key cryptography, there is an important factor, which is the key length. It is optimal to keep smaller key size but gain large security. For imposing the better security, each time of communication a temporary key, called session-key, is used for symmetric key cryptography. Session-key distribution is the process of delivery a key to two parties who wish to exchange data without allowing others to know the key. For two parties A and B, a key distribution technique using *Key Distribution Center (KDC)* has been chosen. For this a generated session key has been delivered among the communicating parties through the *Improved Prime Field ECC* and the complexity of the process has also been calculated and found better than *conventional ECC* and *RSA*, widely used public key cryptography.

3.5. Session Key

A session key is an ephemeral secret, i.e., one whose use is restricted to a short time period such as a single telecommunications connection or a session, after which all trace of it is eliminated. Motivation for use of session keys or ephemeral keys includes the following:

- To limit available cipher text under a fixed key for cryptanalytic attack.
- To limit exposure, with respect to both time period and quantity of data, in the event of session key compromise.
- To avoid long term storage of a large number of distinct secret keys in the case where one terminal communicates with a large number of others, by creating keys only when actually required.
- To create independence across communication sessions or applications.

3.6. Role of Key Distribution Center (KDC)

A key distribution center is responsible for distributing keys to pairs of users as needed. Each user must share a unique key with the key distribution center for purposes of key distribution.

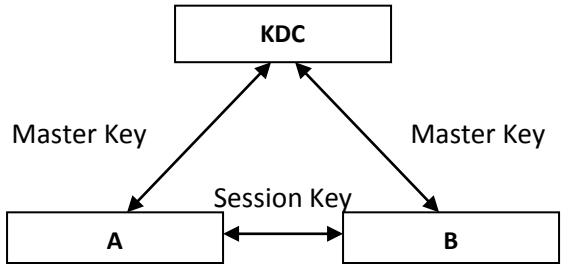

Figure 1: Key distribution

At least two levels of keys must be used:

- Communication between end systems is encrypted using a temporary key, often referred to as a session key.
- Session key are transmitted in encrypted form, using a master key that is shared by the KDC and an end system or user.

4. PROPOSED MODEL

In our proposed model, Session key distribution through *PFECC*, concentrate with the *Step 2* of the *figure 3*, where *KDC* distributes session key for two parties. In this point of view *KDC* acts as the *sender* and *end users* act as the *receiver*.

An underlying finite prime field *Fp* is chosen. An elliptic curve *E* defined over *Fp*, and a base point *P* on *E* is chosen. The order of the point *P* is denoted by *n*.

The field *Fp* and curve *E*, comprise the system parameters, and are public information.

The point *P* is chosen by the *KDC* and transmits it secretly to the receiver for the current communication. The Encryption and Decryption process is formed in the *Figure 2*.

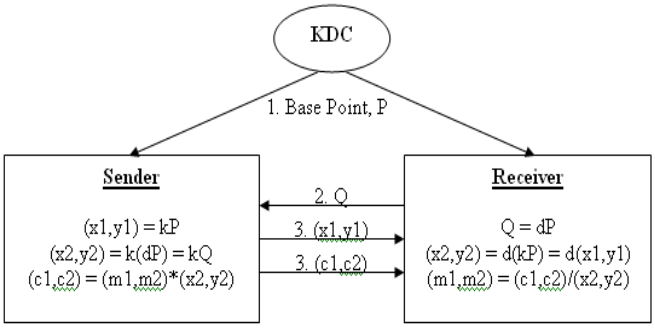

Figure 2: Proposed System Model

In our proposed model, during the key generation receiver will perform the following operations.

1. Select a random integer d in the range [1, n-1].
2. Compute the point Q = dP.
3. The entity's public key consists of the point Q.
4. The entity's private key is the integer d.

4.1. Algorithm: Encryption Process

Base point P, Integer n (order of P),
Integer k (A's private key), Point (x1, y1) (A's private key),
Point Q (B's public key), Point (x2, y2) (encryption key),
Message M, Point (m1, m2) (encoded message, m1ε Fq, m2 ε Fq), Point (c1, c2) (encrypted message), Data c (transmitted data to B).

1. (m1, m2) = M.
2. Read: k = [1, n-1].
3. (x1, y1) = kP.
4. (x2, y2) = kQ.
5. (c1, c2) = (m1, m2) × (x2, y2).
6. c = ((x1, y1), (c1, c2)).

4.2. Algorithm: Decryption Process:

Base point P, Integer n (order of P),
Integer d (B's private key), Point Q (B's public key),

Data c = ((x1, y1), (c1, c2)) (transmitted data from A),
Point (x1, y1) (A's public key), Point (c1, c2) (encrypted message),
Point (x2, y2) (decryption key), Point (m1, m2) (decrypted encoded message)
Message M (decoded).

1. (x2, y2) = d (x1, y1).
2. (m1, m2) = (c1, c2) × $(x2, y2)^{-1}$.
3. M = (m1, m2).

5. IMPLEMENTATION

Let, we consider the following system parameters to implement the proposed model.

Fp : F28
E : $y2 = x3 + x + 13$
P = (9,10)
n = E(F27) = 30

The 34 multiples of P are all distinct, so we can represent letters of the Roman alphabet by distinct points on the elliptic curve to encode the message in terms of EC points.

Here, A wants to send B the message "ODD". It will do this one letter at a time. Let's start with 'O', which is corresponds to the point (6, 9). After A indicates that It wants to send B a message, B choose his secret key s_P=7 and sends A s_PP = 7(9, 10) = (6, 24). A then choose her secret key k=3 and compute kP = 3(9, 10) = (23, 19). Using B's transmission of s_PP = (6, 24), A computes ks_PP=3(6, 24) = (27, 21) and uses this to encrypt 'O'. Now (ks_PP) × (5, 9) = (11, 3), so A sends B a pair of information: kP which is (23,19) and the encrypted message (11, 3).

Table 2: EC points for encoding

n	Np	Letter	n	Np	Letter
1	(10,11)	A	18	(20,29)	R
2	(19,30)	B	19	(6,23)	S
3	(23,20)	C	20	(27,11)	T
4	(5,23)	D	21	(28,22)	U
5	(26,17)	E	22	(29,19)	V
6	(18,19)	F	23	(23,10)	W
7	(7,25)	G	24	(21,30)	X
8	(25,30)	H	25	(17,24)	Y
9	(17,9)	I	26	(25,3)	Z
10	(20,2)	J	27	(6,7)	
11	(23,23)	K	28	(17,13)	
12	(29,14)	L	29	(25,15)	
13	(28,11)	M	30	(4,9)	
14	(27,22)	N			
15	(6,9)	O			
16	(20,4)	P			
17	(11,1)	Q			

B receives (23, 19) and (11, 3) from A. Since B knows that (23, 19)=kP, It can compute the encryption key, ks_PP = s_P(kP) = 7(23, 19) = (27, 21) and hence its associated decrypted key (27^{-1}, 21^{-1}) = (23, 3). Now B applies the decryption key to A's encrypted message to recover m = (23, 3) × (11, 3) = (6, 9). Finally B looks up (5, 9) in the table and recover the letter 'O'. When A sends the next letter to B, it should probably use a different value of k, or ask B to change his private key (the latter is somewhat more cumbersome since it involves more transmissions) and as well as he asks KDC to change the base point. This provides for greater security, since the same letter might have different-looking encryptions depending on where it occurs in any particular message. Also, the rest part of A's transmission (the quantity kP) will always be the same (hence redundant) if it uses the same value of k each time.

5.1. Comparisons of PEECC with RSA

Thus, PEECC devices require less storage, less power, less memory, and often less bandwidth than other public key systems. Current key-size recommended by NIST for legacy public schemes is 2048 bits. A vastly smaller 224-bit PEECC key offers the same level of security. This advantage only increase with security level—for example, a 3072 bit legacy key and a 256 bit PEECC key are equivalent.

Table 3 and figure 6 show the computational complexity for breaking the private key for elliptic curve cryptosystem, using the Pollard ρ method, is 3.8×10^{10} MIPS-years (i.e. millions of instructions per second times the required number of years) for an elliptic curve key size of only 150 bits. If the ECC key length is increased to 234 bits, the system will impose a computational complexity of 1.6×10^{28} MIPS-years (still with the Pollard ρ method) [10].

Table3: Comparisons between PEECC & RSA for key breaking complexity

1.1 PEECC		1.2 RSA	
Key Length	Breaking Complexity	Key Length	Breaking Complexity
150 bits	3.8×10^{10} MIPS-years	1024 bits	3×10^{11} MIPS-years
234 bits	1.6×10^{28} MIPS-years	2048 bits	3×10^{20} MIPS-years

Table 4: Key Size Ratio

PEECC Key Size(Bits)	RSA Key Size(Bits)	Key Size Ratio(Bits)
163	1024	1:6
256	3072	1:12
384	7680	1:20
512	15360	1:30

Table 5: Comparisons between conventional & proposed PFECC

Conventional ECC	Proposed PFECC
Security: S	Security: t(S) where t = Base point security

6. CONCLUSIONS

This paper proposed an improved model for both session-key distribution and message transactions through Prime Field Elliptic Curve Cryptography. The main advantages of using this model over other cryptosystems include the following:

- Small key-size of 160-260 bits as compared to 512-1024 bits with traditional schemes such as RSA.
- Extra security imposed by distributing the base point secretly through KDC.
- High edibility and enhanced security through periodically changing the curve, without requiring extensive hardware changes.
- Simplicity of implementation facilitates a wide range of applications including m-commerce (WAP), smart card system (EMV), e-commerce and banking applications (SET) and internet based applications (SSL).

7. REFERENCES

[1]. Johannes Buchmann, Michael J. Jacobson JR, Edlyn Teske. On Some Computational Problems in Finite Abelian Groups, Mathematics of Computation, Volume 66 Number 220, October 1997, pp. 1663-1687.

[2]. Neil Koblitz. Elliptic curve cryptosystems. In Mathematics of Computation, volume 48, pages 203–209, 1987. C356

[3]. Victor S. Miller. Use of elliptic curves in cryptography. In Advances in Cryptology—Proceedings of Crypto 85, volume 218 of Lecture Notes in Computer Science, pages 417–426. Springer-Verlag, 1986. C356.

[4]. Erik De Win, Serge Mister, Bart Preneel, and Michael Wiener. On the performance of signature schemes based on elliptic curves. In Algorithmic Number Theory: Third International Symposium, ANTS-III, Proceedings, volume 1423 of Lecture Notes in Computer Science, pages 252–266.Springer-Verlag, 1998. C357.

[5]. M. V. Wilkes. Time-Sharing Computer Systems, Elsevier, 1968.

[6]. G.Purdy. A High-Security Log-In Procedure, Communications of the ACM, 17:442-445, 1974.

[7]. W. Di_e, M.E. Hellman. New Directions in Cryptography, IEEE Transactions on Information Theory, IT-22:644-654, 1976.

[8]. P. Ivey, S.Walker, J. Stern, S. Davidson. An ultra-high speed public key encryption processor, Proceedings of IEEE Custom Integrated Circuits Conference, Boston, 19.6.1-19.6.4, 1992.

[9]. Alfred Menezes. Elliptic Curve Public Key Cryptosystems, Kluwer Academic Publishers, USA, 1993.

[10]. Certicom. An Introduction to Information Security, The First in a Series of ECC Whitepapers, March 1997.

[11]. S. Goldwasser, S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information, Proceedings of 14th ACM Symposium on Theory of Computing, pp. 365-377.

[12]. Neal Koblitz. Algebraic Aspects of Cryptography, Algorithms and Computation in Mathematics, Volume 3, Springer-Verlag, New York, 1998.

[13]. Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone. Handbook of Applied Cryptography, CRC Press, USA, 1997.

[14]. R.L. Rivest, A. Shamir, L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM, 21(2):120-126, February 1978.

[15]. T. Rosati. A high speed data encryption processor for public key cryptography, Proceedings of IEEE Custom Integrated Circuits Conference, San Diego, 12.3.1 - 12.3.5, 1989.