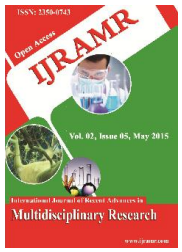
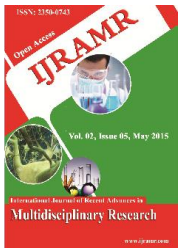


International Journal of Recent Advances in Multidisciplinary Research
Vol. 02, Issue 05, pp.0431-0435, May, 2015



Research Article

SURVEY ON ELLIPTICAL CURVE CRYPTOGRAPHY

***Seelam Lakshmi Sravani**

7-2-m8, Sriram Hills, Khammam, India

ARTICLE INFO

Article History:
Received 19th February 2015
Received in revised form
21th March, 2015
Accepted 25th April, 2015
Published online 31th May, 2015

Keywords:
Communication,
Encryption,
Cryptography,
Provides Security.

ABSTRACT

The main objective of our survey is based on elliptical curve cryptography. It provides security for encryption and decryption of data. We have gone through several papers, each paper describes about some protocols. some of the papers are related to Text based, Wireless Communication and java as implementation tool .The above methods has its own protocols. One is based on customizable cryptography. It produces hardware designs for ECC. Encryption is a process of encoding messages or information in such a way that only authorized parties can read it. Decryption is a process of decoding data that has been encrypted into a secret format. It requires a secret key or password.

INTRODUCTION

Elliptic Curve Cryptography (ECC) was discovered in 1985 by Victor Miller (IBM) and Neil Koblitz (University of Washington) as an alternative mechanism for implementing public-key cryptography. Public key algorithms is a mechanism for sharing keys among large numbers of participants or entities in a comple information system. It is unlike other popular algorithms such as RSA, ECC is based on discrete logarithms that is much more difficult to challenge at equivalent key lengths.

Elliptic curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. One of the main benefits in comparison with non-ECC cryptography (with plain Galois fields as a basis) is the same level of security provided by keys of smaller size. Elliptic curves are applicable for encryption, digital signatures, pseudo-random generators and other tasks. They are also used in several integer factorization algorithms that have applications in cryptography, such as Lenstra elliptic curve factorization Public-key cryptography is based on the intractability of certain mathematical problems.

Corresponding author: Seelam Lakshmi Sravani
7-2-m8, Sriram Hills, Khammam, India

Early public-key systems are secure assuming that it is difficult to factor a large integer composed of two or more large prime factors. For elliptic curve based protocols, it is assumed that finding the discrete logarithm of a random elliptic curve element with respect to a publicly known base point is infeasible: this is the "elliptic curve discrete logarithm problem" or ECDLP. The security of ECC depends on the ability to compute a point multiplication and the inability to compute the multiplicand given the original and product points. The size of the elliptic curve determines the difficulty of the problem. The first benefit promised by ECC is a smaller key size, reducing storage and transmission requirements, i.e. that an elliptic curve group could provide the same level of security afforded by an RSA-based system with a large modulus and correspondingly larger key. For present cryptographic purposes, an *elliptic curve* is a plane curve over a finite field (rather than the real numbers) which consists of the points satisfying the equation along with a distinguished point at infinity, denoted ∞. (The coordinates here are to be chosen from a fixed finite field of characteristic not equal to 2 or 3, or the curve equation will be somewhat more complicated). Elliptic curves are applicable for encryption, digital signatures, pseudo-random generators and other tasks. They are also used in several integer factorization algorithms that have applications in cryptography, such as Lenstra elliptic curve factorization. NIST recommended fifteen elliptic curves. Specifically, FIPS 186-3 has ten recommended finite fields:

- Five prime fields F_p for certain primes *p* of sizes 192, 224, 256, 384, and 521 bits. For each of the prime fields, one elliptic curve is recommended.
- Five binary fields F_2^m for *m* equal 163, 233, 283, 409, and 571. For each of the binary fields, one elliptic curve and one Koblitz curve was selected.

The NIST recommendation thus contains a total of five prime curves and ten binary curves. The curves were ostensibly chosen for optimal security and implementation efficiency.
In 2013, the *New York Times* stated that Dual Elliptic Curve Deterministic Random Bit Generation (or Dual _EC_DRBG) had been included as a NIST national standard due to the influence of NSA, which included a deliberate weakness in the algorithm and the recommended elliptic curve. RSA Security in September 2013 issued an advisory recommending that its customers discontinue using any software based on Dual EC_DRBG. In the wake of the exposure of Dual EC_DRBG as "an NSA undercover operation", cryptography experts have also expressed concern over the security of the NIST recommended elliptic curves, suggesting a return to encryption based on non-elliptic-curve groups.

Proposed Algorithms

I. Text Based

To do operations with EC points in order to encrypt and decrypt the points have to be generated first. The algorithm *'gen Points'* describes the process of generating the points for the given parameters 'a', 'b', and 'p'. Also the algorithm *'ECC'* describes the process of encryption and decryption on EC field.

Algorithm gen Points (a, b, p)
{
x-O;
While(x <p)
y2=(~ + ax + b) mod p;
if (y2 is a perfect square in GF(p))
output(x, sqrt (y)) (x,-sqrt (y));
x=x + I;
}
Algorithm ECC
{
II Key Distribution
II Let UA and UB be legitimate users
UA= {PA, n A} *II Key* pair for UA
UB= {P B ,n B} *II Key* pair for UB
II Send the Public key of U i, to UA
Send(PB,UA);

II Send the Public key of U x to UB
Send (PA, UB);
II Encryption at A
Pm1=a Pm
I la: Ascii value of text
II Pm: random point on EC
PB=n B * G

IIG is the base point of EC
Lin B is the private key
Cipher Text={k G, P ml +k*PB}
II Decryption at B

Let k G be the first point and
P ml + k*PB be the second point
N B k G = I l g * first point;
Calculate P ml = P ml + k PB- n B k G;
Calculate the Pm value from P ml
using discrete logarithm

II. Wireless Communication

Network Server Initialization

SERVER		CERTIFICATION AUTHORITY
• Choose $d_s \in [2, n-2]$		• Choose $k_s \in [2, n-2]$
• $Q_s = d_s \times P$		• $R_s = k_s \times P$
• Send	$\xrightarrow{Q_s}$	• Receive
		• Choose unique I_s
		• $r_s = R_s.x$
		• $s_s = k_s^{-1}(H(Q_s.x, I_s, t_s) + d_{ca} \cdot r_s)$
	$\xleftarrow{Q_{ca}, I_s, (r_s, s_s), t_s}$	• Send
• Receive		
• $e_s = H(Q_s.x, I_s, t_s)$		
• Store $Q_s, Q_{ca}, I_s, (r_s, s_s), e_s, t_s$		

User Terminal Initialization

USER		CERTIFICATION AUTHORITY
• Choose $d_u \in [2, n-2]$		• Choose $k_u \in [2, n-2]$
• $Q_u = d_u \times P$		• $R_u = k_u \times P$
• Send	$\xrightarrow{Q_u}$	• Receive
		• Choose unique I_u
		• $r_u = R_u.x$
		• $s_u = k_u^{-1}(H(Q_u.x, I_u, t_u) + d_{ca} \cdot r_u)$
	$\xleftarrow{Q_{ca}, I_u, (r_u, s_u), t_u}$	• Send
• Receive		
• $e_u = H(Q_u.x, I_u, t_u)$		
• Store $Q_u, Q_{ca}, I_u, (r_u, s_u), e_u, t_u$		

Mutual Authentication and Key Agreement

USER		SERVER
• Receive	$\xleftarrow{Q_s}$	• Send
• Send	$\xrightarrow{Q_u}$	• Receive
• $Q_k = d_u \times Q_s = (d_u \cdot d_s) \times P$		• $Q_k = d_s \times Q_u = (d_s \cdot d_u) \times P$
• $Q_k.x$: The mutually agreed key		• $Q_k.x$: The mutuallly agreed key
		• Generate a random number g
		• $C_0 = E(Q_k.x, (e_s, (r_s, s_s), t_s, g))$
• Receive	$\xleftarrow{C_0}$	• Send
• $D(Q_k.x, C_0)$		
• $C_1 = E(Q_k.x, (e_u, (r_u, s_u), t_u, g))$		
• Send	$\xrightarrow{C_1}$	• Receive
		• $D(Q_k.x, C_1)$
		• If g and t_u are valid, then
• $c = s_s^{-1}$		• $c = s_u^{-1}$
• $u_1 = c \cdot e_s$		• $u_1 = c \cdot e_u$
• $u_2 = c \cdot r_s$		• $u_2 = c \cdot r_u$
• $R = u_1 \times P + u_2 \times Q_{ca}$		• $R = u_1 \times P + u_2 \times Q_{ca}$
• $v = R.x$		• $v = R.x$
• If $v \neq r_s$, then abort		• If $v \neq r_u$, then abort
• $k_m = Q_k.x + g$		• $k_m = Q_k.x + g$
• k_m: The unique secret key		• k_m: The unique secret key

As is customary in most security protocols, we assume that there is a certificate authority (CA) which creates and distributes certificates to the users and servers on their request. These certificates contain a temporary identity assigned by the CA for the requesting party, the public key of the Requesting party, and the expiration date of the certificate. The concatenated binary string is then signed by the CA's private key to obtain the certificate for the requesting party. By using a certificate the identity of a particular party is bound to its public key. The acquisition of the certificate is performed when the users and servers rst subscribe to the service. The certificates are updated at regular intervals, for example, in the beginning of each month after paying the monthly charge. In a wireless environment, it is often necessary to request service outside of users home net-works. In this case, the visited network checks the certificate's expiration date with the users home network in order to decide whether it needs to provide service to the requesting party. Thus, the authentication and communication protocols should be designed in such a way that the users can easily be authenticated on-line via their home networks.

III. Java as Implementaton Tool

First of all, the points are generated for the elliptic curve based on the values of prime modulo p and predefined Coefficients a, b (It is to be noted that _a‘ and _b‘ remain constant throughout the application of an elliptic curve).

Gen Points (prime, a, b)
{
Step 1: initialize x = 0;
Step 2: while (x<p)
y2 = (x3 + ax + b) mod prime;
if (LHS = RHS)
output (sqrt (x), sqrt (y));
x = x+1;
}
Algorithm ECC
{
Algorithm for Key Distribution
Step 1: //For user A
PUB = G*P
UA = (PUA, PA) // User A key pair
Step 2: // For User B
PUB = BP*PB
UB = (PUB, PB) //User B key pair// BP is the Base Point.
Step 3: //Send the Public key of UB to UA Send (PUB, UB);
Step 4: //Send the Public key of UA to UB Send (PUA, UA);
Algorithm for text Encryption
Step 1: Calculate APL = p*AP;
//p = Ascii value of text
//AP: random point on EC
Step 2: // Calculate k BP
K BP = k*BP
//BP is the Base Point
Step #: // Send Cipher test to receiver, i.e. User B
Cipher Text, CM = {k BP, APL+ k * PUB} **Algorithm for text Decryption**
Let k BP be the first point
APL+ k PUB be second point
Step 1: Calculate PB k BP = PB * first _point
//this yields us an equivalent point to k PUB
Step 2: Calculate APL = (APL+ k*PUB) – PB k BP
Now using discrete logarithm concept
Step 3: Evaluate value of sent text from APL
APL = r AP
//r is the value to be calculated using the discrete logarithmic concept. r = p, i.e. the original ASCII value.

IV. Public Key Cryptosystem Technique with Generator g for Image Encryption

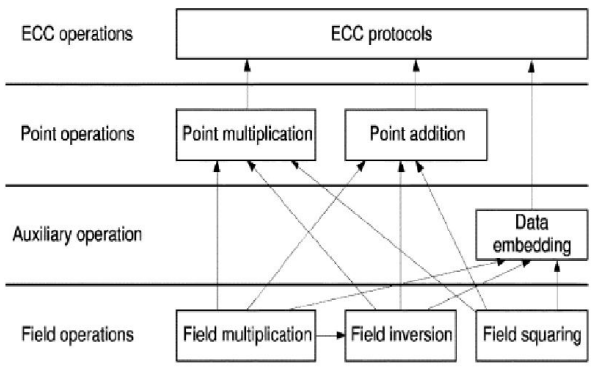
ECC can be used for encryption and decryption. Consider the user A want to encrypt a sw image for the user B, and then the following steps are involved.

Step1.Take any RGB color image as sw.
Step2. A encodes the sw image as sw P = (x, y) =(g**5**,g**3**). similarly others points are calculated using equation (4) with generator g.
Step3. A choose a random number K and produce the
Cipher text C [k G,P k P]
S w s w B
= ´ + ´ and
sends this cipher text
s w C to B.
Step4. To decrypt the s w image, B computes
B n´ k´ G.
Step5. B again computes
S w B s w B P + k´ P - n ´ K´ G= p - k(n ´ G)+ k P
= s w B P - k´ n + k´ n = s w P. In other words, we can say B picks the first co-ordinate KG of **s w C** , multiply that with his private key and then subtract this form the second point s w B P + k´ P .

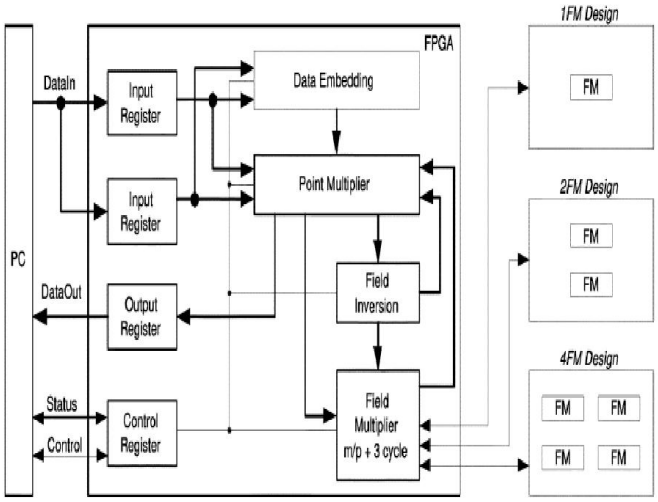
V. ECC based on customizable system

This method is about producing hardware designs for ECC systems over finite field GF(2^m) using the optimal normal basis for representation of numbers. Our field multiplier design is based on a parallel architecture containing multiple m-bit serial multipliers; by changing the number of such serial multipliers, designers can obtain implementations with different tradeoffs in speed, size and level of security**.** we have developed a parametric model for estimating the number of cycles for our generic ECC architecture. The resulting hardware implementations are among the fastest

Reported: for a key size of 270 bits, a point multiplication in a Xilinx XC2V6000 FPGA at 35 MHz can run over 1000 times faster than a software implementation on a Xeon computer at 2.6 GHz.



Interactions between different operations



Data path of the customizable ECC system (FM denotes a field multiplier)

Since the time to perform a multiplication is usually much longer than the time to perform a shift operation, the number of Cycles for this inversion algorithm can be approximated by the following equation where T multi represents the number of Cycles to perform a field multiplication.

Algorithm is described as below:

Input: $a \in GF(2^m)$ to be inverted
Output: $x = a^{-1}$
- $x \leftarrow a$; $s \leftarrow \log_2(m) - 1$
- while $s \geq 0$
  - $r \leftarrow$ right shift m by s bits
  - $y \leftarrow$ left shift x by $\lfloor r/2 \rfloor$
  - $y \leftarrow$ multiply x by y
  - if x is odd
    - $y \leftarrow$ left shift y by 1 bit
    - $y \leftarrow$ multiply x by y
  - $x \leftarrow y$
  - $s \leftarrow s - 1$
- $x \leftarrow$ left shift x by 1 bit
- return x

In this Customizable ECC we compare the performance of various software and hardware implementations for point multiplication, which is the bottleneck for ECC systems. The comparison for serial and parallel designs on different m and p values, where p refers to the degree of parallelization, is presented in Table II. Note that Place-and- Route (P&R) results mean the results that are obtained from the Celoxica DK3 and Xilinx ISE 6.2 tools, and measured results refer to the measured results from hardware realization. It can compute a point multiplication up to 1150 times faster than a software ECC application on a Xeon 2.66-GHz computer. On-going and future work includes functional extensions and optimizations such as speed improvement, resource minimization resource minimization, and run-time customization of ECC designs.

Conclusion

We have described an authentication and key agreement protocol for wireless communication based on elliptic curve cryptographic techniques. The proposed protocol is a public-key type with the feature of o_-line certi_cation procedure. It is a well-known fact that the public-key cryptography concept solves the key distribution and storage problems, which are typical in secret-key settings. The protocol provides certain security services, e.g., nonrepudiation, anonymity of user, service expiration mechanism using time certi_cates, as most recent secret and public-key based protocols also provide.

REFERENCES

Koblitz, N. 1987. Elliptic Curve Cryptosystems, *Mathematics ofComputation,* volA8, pp.203 -209

Agnew, G. B., R. C. Mullin and S. A. Vanstone, 1993. An implementation of elliptic curve cryptosystems over F2155. IEEE Journal on Selected Areas in Communications, 11(5):804{813.

Rajaram Ramasamy, R., M. Amutha Prabakar, M. Indra Devi and M. Suguna, 2009. Knapsack based ECC encryption and decryption‖, International Journal of Network Security, Vol. 9, No. 3, PP. 218-226.

Darren Hankerson, Julio Lopez Hernandez and Alfred Menezes, Software implementation of Elliptic Curve Cryptography over Binary Fields‖

Cryptographic Hardware and Embedded Systems — CHES 2000, Lecture Notes in Computer Science Volume 1965, 2000, pp 1-24.

Kristin Lauter, 2006. "The Advantages of Elliptic Cryptography for Wireless Security", *IEEE Wireless Communications,* pp. 6 2 – 67.

Maria Celestin Vigila, S. and K. Muneeswaran, 2009. Implementation of text Based cryptosystem using elliptic curve cryptography‖, IEEE.

Sravana Kumar, D., CH. Suneetha and A. Chandrasekhar, 2012. Encryption of data using Elliptic Curve over Finite Field‖, IJDPS, Vol. 3, No. 1.

Rajaram Ramasamy, R., M. Amutha Prabakar, M. Indra Devi and M. Suguna, 2009. Knapsack based ECC encryption and decryption‖, International Journal of Network Security, Vol. 9, No. 3, PP. 218-226.

Padma, Bh., D. Chandravathi and P. Prapoorna Roja, 2010. Encoding and Decoding of a Message int the Implementation of Elliptic Curve Cryptography using Koblitz's method‖, IJCSE, Vol. 02, No. 05.

William Stallings, 2010. Cryptography and Network Security, Prentice Hall, 5th Edition.

Vivek Kapoor, Vivek Sonny Abraham and Ramesh Singh, 2008. Elliptic Curve Cryptography‖, ACM Ubiquity, vol. 0, Issue 20, May 20-26.

Darren Hankerson, Julio Lopez Hernandez and Alfred Menezes, Software implementation of Elliptic Curve

Cryptography over Binary Fields‖, Cryptographic Hardware and Embedded Systems — CHES 2000, Lecture Notes in Computer Science Volume 1965, 2000, pp 1-24.

Malan, D.J., M. Welsh and M.D. Smith, 2004. "A public-key infrastructure for key distribution in TinyOS based on

elliptic curve cryptography," *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on* , vol., no., pp.71,80, 4-7.

Aydos, M., T. Yanik and C.K. Kog, 2001. "High-speed implementation of an ECC based wireless authentication protocol on an ARM microprocessor," *lEE Proc Commun., Vol.* 148, No.5, pp. 273-279.

Kristin Lauter, 2006. "The Advantages of Elliptic Cryptography for Wireless Security", *IEEE Wireless Communications,* pp. 6 2 – 67.

Kristin Lauter, 2006. "The Advantages of Elliptic Cryptography of Wireless Security," IEEE Wireless Communications, pp.62-67.

Chaur, 2004. Chin Chen "RSA scheme with MRF and ECC for Data Encryption," 0-7803-8603-5/04 IEEE.

Kefa Rabah, 2006. "Elliptic Curve Cryptography over Binary Finite Field GF (2m)". Information Technology Journal 5(1) pp. 204-229, ISSN 1812-5638.

Luminita Scripcariu and Mircea Daniel Frunza, 2005. "A New Image encryption Algorithm based on Iversiable Functions defined on Galois Fields, " pp. 243-246l, ISSN 0-7803-9029-6/05, IEEE.

Philip P. Dang and Paul M. Chau, 2000. "Image Encryption for Secure Internet Multimedia Application", IEEE Transaction on Consumer Electronics, Vol. 46, No.3 pp. 395-403.

****