http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 4 (3): 299-306, 2005
ISSN 1812-5638

Elliptic Curve ElGamal Encryption and Signature Schemes

Kefa Rabah
Department of Physics, Eastern Mediterranean University,
Gazimagusa, North Cyprus, via Mersin 10, Turkey

Abstract: In this research Elliptic Curve ElGamal (ECEG) cryptosystems was studied. The ElGamal signature algorithm is similar to the encryption algorithm in that the public-key and private-key have the same form; however, encryption is not the same as signature verification, nor is decryption the same as signature creation as in RSA. The DSA is based in part on the ElGamal signature algorithm.

Key words: Elliptic curve cryptosystems, ElGamal cryptosystems, digital signature, online content management, wireless devices, internet security

INTRODUCTION

In the computing age of today, we have witnessed the growing popularity of the Internet and networks in our society. With these tools at our fingertips, we are able to communicate and do business even more quickly and efficiently than ever before. For example, businesses can market their products online so customers do not have to leave their homes and banks can conduct transfers and manage accounts with more ease, speed and functionality than with the paperwork of the past. Also, what is probably the most popular means of communication, email, is used by just about everyone each and every day. It is clear that these modern conveniences have made our lives much smoother. However, as we continue to add these conveniences to our lives, we open the door to more numerous, possibly even more dangerous, outlets for attacks. With the prominence of identity theft on the rise, we must all be weary of the security of online communication. One solution is to hide our data by using encryption algorithms that only allow those that we trust and/or exchanged cryptographic session keys with, to view the information. We can also verify the integrity of our data through electronic signatures and electronic certificates-this is the heart of cryptography.

Cryptography offers a set of sophisticated security tools for a variety of problems, from protecting data secrecy, through authenticating information and parties, to more complex multi-party security implementation. With most modern cryptography, the ability to keep encrypted information secret is based not on the cryptographic algorithm, which is widely known, but on a number called a key that must be used with the algorithm to produce an encrypted result or to decrypt previously encrypted information. Currently, there are two popular kinds of cryptographic protocols: symmetric-key and asymmetric-key protocols[1,2]. In the symmetric-key protocols, a common key (the secret-key) is used by both communicating partners to encrypt and decrypt messages. Among these are DES, IDEA and AES[2]. These symmetric-key cryptosystems provide high-speed key and communication but have the drawback that a common (or session) key must be established for each pair of participants. The process of exchanging the cryptographic key is referred to as key distribution and can be very difficult.

In the public-key protocols we have two associated keys; one is kept private by the owner and used either for decryption (confidentiality) or encryption (signature) of messages. The other key is published in the public domain (public-key server) to be used for the reverse operation or decryption. Different public-key cryptographic systems are used to provide public-key security. Among these we can mention the RSA[3], Diffie-Hellman (DH)[4], Digital Signature Algorithm (DSA)[5], ElGamal cryptosystem[6] and in recent years the Elliptic Curve Cryptography (ECC)[7]. These systems provide these services by relying on the difficulty of different classical mathematical problems, hence provide the services in different ways. In this work we will concentrate on elliptic curve cryptography focusing on EC-ElGamal crypto-scheme.

Elliptic Curve Cryptosystem (ECC) is relatively new[7]. Elliptic curves are mathematical constructions from number theory and algebraic geometry, which in recent years have found numerous applications in cryptography. The ECC was first introduced by Victor Miller[8] and independently by Koblitz[9] in the mid 1980s. It is a new branch in cryptography that uses an old, interesting and difficult topic in mathematics or, particularly, algebra: elliptic curves over finite fields. The elliptic curve approach is a mathematically richer procedure than traditional cryptosystems e.g., RSA, DH, ElGamal, DSA etc. The ECC from the very beginning was proposed as an

alternative to established public-key systems such as the traditional cryptosystems. This is because elliptic curves do not introduce new cryptographic algorithms, but they implement existing public-key algorithms using elliptic curves. In this way, variants of existing schemes can be devised that rely for their security on a different underlying hard problem. Today it has evolved into a mature public-key cryptosystem. The U.S. government[10] recently endorsed it as an alternative public key algorithm.

ECC ARITHMETIC OVER FINITE FIELD

The use of elliptic curve groups over finite fields as a basis for a cryptosystem was first suggested by Koblitz[9]. An elliptic curve can be defined over any field (e.g., real, rational, complex). However, elliptic curves used in cryptography are mainly defined over finite fields[7,11-13]. Finite fields, also called Galois fields, are fields consisting of a finite number of elements. The cost, speed and feasibility of elliptic curve cryptosystems depend on the finite field F_q, where $q = p^m$, on which it is implemented. There are usually two finite fields to work on: prime finite field F_p(i.e., m = 1) when p is a prime number >3 and binary finite field GF(2^m) or F_{2^m}.

An elliptic curve $E(F_p)$ consists of elements (x, y) of the form:

$$E: y^2 = x^3+ax+b \text{ with } x, y, a, b \in F_p = \{1,2,3,..., p-2, p-1\} \quad (1)$$

together with a single element denoted by O called the "point at infinity". Abstractly a finite field consists of a finite set of objects called field elements together with the description of two operations-addition, multiplication-that can be performed on pairs of field elements. These operations must possess certain properties. The addition operation in an elliptic curve is the counterpart to modular multiplication in common public-key cryptosystems and multiple addition is the counterpart to modular exponentiation[11-13]. The order of a finite field is the number of elements in the field. There exists a finite field of order q if and only if q is a prime power, then there are, however, many efficient implementations of the field arithmetic in hardware or in software[14].

Now given a message, m, we must first choose a large integer, k and a suitable elliptic curve, $E(F_p)$ defined as above. We must then embed the message m onto a point, P, on the curve. This is not as straightforward as it looks and involves the use of quadratic residues and probabilistic algorithms. In the next sections we are going to develop the methodology to achieve that task.

Construction of the ECC arithmetic over finite field: The core of the ECC is when it is used with Galois Field it becomes a one way function i.e., the math's needed to compute the inverse is not known. Let an elliptic curve group over the Galois Field $E_p(a,b)$ where, p>3 and is prime, be the set of solutions or points P = (x, y) such that $(x, y \in E_p(a,b))$ that satisfy the equation: $y^2 = x^3 + ax + b \pmod p$ for $0 \le x < p$ together with the extra point O called the point at infinity. The constants a and b are non negative integers smaller than the prime number p and must satisfy the condition: $4a^3+27b^2 \neq 0 \pmod p$. For each value of x, one needs to determine whether or not it is a quadratic residue. If it is the case, then there are two values in the elliptic group. If not, then the point is not in the elliptic group $E_p(a,b)$. The number of points on $E_p(a,b)$ is denoted by $\#E(F_p)$. Since 50% of integers mod p are quadratic residues, the number of points will be roughly p+1, counting the point at infinity. Exact number is between $\#E(F_q) \le q+1-t$ where, $|t| \le 2\sqrt{q}$, is called the trace of E[7]. In fact, the general theory says that there will be about p points (x, y) with error bounded by $O(\sqrt{P})$. The order of the group is known to all parties; we can either generate a curve at random and counts its order (Schoof algorithm)[7], choose an order and use a constructive algorithm to derive a curve (method of complex multiplication is most common). In this work will use the naïve approach and also elliptic curve builder, a free open source software[15]. Here we will design systems to use prime order group (sub group) of points on the elliptic curve.

Point addition algorithm: The basic condition for any cryptosystem is that the system is closed, i.e., any operation on an element of the system results in another element of the system. In order to satisfy this condition for Elliptic curves it is necessary to construct nonstandard addition and multiplication operations. Capitals represent points on the curve while lower case represents integers. The addition of two points on an elliptic curve is defined in order that the addition results will be another point on the curve as presented in Algorithm 1.

In either case, when $P_1 = P_2$ (doubling) and $P_1 \neq P_2$ (point addition), major operations are field multiplication

Algorithm 1: Point Addition Equation

Input: $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$
Output: $P_1+P_2=P_3 = (x_3, y_3)$
1. If $P_1 = P_2$: $x_3 = \lambda^2-x_1-x_2$, $y_3 = \lambda(x_1-x_3)-y_1$
where, $\lambda = (3x_1^2+a)/2y_1$ (Point doubling)
2. Else if $P_1 \neq P_2$: $x_3 = \lambda^2+\lambda+x_1+x_2+a$, $y_3 = \lambda(x_1+x_3)+x_3+y_1$
where, $\lambda = (y_2-y_1)/(x_2-x_1)$ (Point addition)
3. Return: (x_3, y_3)

and field inversion. (Squaring and field addition are enough ignorable because of its less computation time.) From these formulas of Algorithm 1, we can determine the number of field operations required for each kind of elliptic curve operation. We see that in affine coordinates, point addition step usually requires 6 addition/subtraction operations, three modular multiplications and one inversion. A doubling step usually requires 7 addition/subtraction operations, four modular multiplications, one squaring and one inversion. A Negation step requires one addition.

Construction of an elliptic curve over F_p: Let the prime number p = 29 and consider an elliptic curve E: $y^2 = x^3-x+16$ defined over F_{29}, with the constants a = -1 and b = 16, which have been checked to satisfy that E is indeed an elliptic curve. We then determine the quadratic residues Q_{29} from the reduced set of residue Z_{29} {1, 2, 3,...., 27, 28}, which is given by Q_{29} = {1, 4, 5, 6, 7, 9, 13, 16, 20, 22, 23, 24, 26, 28}. Which we use to determine the values of E_{29}(-1, 16), i.e.,:

$$E_{29}(-1, 16)=\begin{Bmatrix} (0,4) & (0,25) & (1,4) & (1,25) & (2,14) \\ (2,15) & (5,7) & (5,22) & (6,9) & (6,20) \\ (7,2) & (7,27) & (10,7) & (10,22) & (13,5) \\ (13,24) & (14,7) & (14,22) & (16,6) & (16,23) \\ (18,1) & (18,28) & (21,11) & (21,18) & (22,12) \\ (22,17) & (23,3) & (23,26) & (28,4) & (28,25) \end{Bmatrix}$$

For a given point P = (x_p, y_p), x_p and y_p are called the x and y coordinates of P, respectively. We now discuss efficient algorithms to expedite implementation procedures in elliptic curve cryptosystems.

ALGORITHMS FOR ELLIPTIC SCALAR MULTIPLICATION

Just as modular exponentiation determines the efficiency of RSA cryptographic systems[3], scalar multiplication dominates the execution time of ECC systems. Scalar multiplication is the operation to compute kP, where, k is a random integer and P is an elliptic curve generation point, say (x_p, y_p) and it can be defined as the combination of additions of two points on an elliptic curve. That is, the calculations of the form: Q = kP =P+P+P...+P (k-summands). Here P is a fixed point that generates a large, prime subgroup of E(F_q), or P is an arbitrary point in such a subgroup and, k is an integer in the range of [1, n-1] where, n is the order of the elliptic curve E. Elliptic curves have some properties that allow optimization of scalar multiplications. The important contributors to the run time are the multiplications and inversions[7,11,12]. In all the protocols that are fundamental implementation of ECC, say ECDH, ECElGamal, ECDSA, ECAES etc., the most time consuming part of the computations are scalar multiplications.

To test the algorithm, let P=(5, 7) $\in$ E_{29}(-1, 16). Then 2P = (x_3, y_3) is equal to: 2P = P + P= (x_1, y_1)+(x_1, y_1) = (28, 4). Next, test addition of two different points on the curve, i.e., Q = (1, 25) $\in$ E_{29}(-1, 16) and R = (6, 20) $\in$ E_{29}(-1, 16). Then Q+R = (x_3, y_3) = (23, 26), which is also on the curve. However, for real implementation of ECC, we need to know the order of the elliptic curve group.

Let us now implement the scalar multiplication to form a subgroup of points <P>, where, <P> is the finite cyclic group <P> = {P, 2P, 3P, ..., nP}, with order n, by following the same additive rules and a generator point P. For example, let P = (5,7) $\in$ E_{29}(-1, 16) be a generator point, which we use through repeated addition of point to generate all the point on the curve (Table 1).

Observe from above algorithms that the addition of two elliptic curve points in E(F_p) requires a few arithmetic operations (addition, subtraction, multiplication and inversion) in the underlying field F_p. The most basic operation is adding two points or doubling a point on an elliptic curve. It is more expensive computationally than a basic operation in a symmetric-key cryptosystem (a block encryption/decryption). But it is still much faster than a basic modular multiplication over a cyclic group whose order is of the same security level[2]. The methods, which included subtractions, are more attractive than the corresponding methods, which included divisions in calculating power in finite fields. The reason is division or inversion in finite fields is a more costly operation than multiplication, while subtraction is just as costly as addition in elliptic curve operations.

340 ElGamal elliptic curves cryptosystems: Taher ElGamal was the first mathematician to propose a public-key cryptosystem based on the Discrete Logarithm problem (DLP)[6,16]. He in fact proposed two distinct cryptosystems: one for encryption and the other for digital signature scheme in 1984, well before elliptic curves were introduced in cryptography. Since then, many variations have been made on the digital signature system to offer improved efficiency over the original system. The ElGamal public-key encryption scheme can be viewed as Diffie-Hellman key agreement protocol in key transfer mode[7]. Its security is based on the intractability of the Discrete Logarithm Problem (DLP) and the Diffie-Hellman problem[16].

Suppose two entities, the Bank (B) and the customer Alice (A) wants to communicate between each other over

Table 1: Point addition/scalar point multiplicative values of P (Note that: $kP \equiv P_k$, so that $31P \equiv P_0 \equiv [O]$ etc.).

1P = [5, 7]	2P = [28, 4]	3P = [18, 1]	4P = [22, 12]	5P = [6, 20]	6P = [13, 5]
7P = [2, 14]	8P = [21, 11]	9P = [23, 3]	10P = [10, 7]	11P = [14, 22]	12P = [16, 23]
13P = [7, 27]	14P = [1, 4]	15P = [0, 4]	16P = [0, 25]	17P = [1, 25]	18P = [7, 2]
19P = [16, 6]	20P = [14, 7]	21P = [10, 22]	22P = [23, 26]	23P = [21, 18]	24P = [2, 15]
25P = [13, 24]	26P = [6, 9]	27P = [22, 17]	28P = [18, 28]	29P = [28, 25]	30P = [5, 22]
31P = [0]					

Algorithm 2: EC-ElGamal protocol

Key generation: (A)
1. Select a random integer k_A from [1, n-1].
2. Compute: $A = k_A P$
3. A's public key is $k_A P$ or (E, P, A), A's private key is k_A

Encryption: (B)
1. Select a random integer k_B from [1, n-1].
2. Compute $B = k_B P$ such that $S_{AB} = k_B(k_A P) = k_A(k_B P) = (x_S, y_S)$
3. **If** $x_P = 0 \pmod p$ and $y_P = 0 \pmod p$ **then go to step 2.**
4. Compute: $C_{M1} = x_S M_1$ **and** $C_{M2} = y_S M_2$
(Note the calculation are done, mod p)
5. Send (B, C_{M1}, C_{M2}) to A

Decryption: (A)
A receives (B, C_{M1}, C_{M2}) and does the following
1. Compute $S_{AB} = k_A(k_B P) = (x_P, y_P)$
2. **Compute** $M_1 = C_{M1}/x_P$ **and** $M_2 = C_{M2}/x_P$,
(Note the calculation are done, mod p)
3. Recover the message $M = (M_1, M_2)$

- Note that in step 2 we can also compute $hk_A P$ and $hk_B P$, which can resist the attack on small subgroup. Where h is a co-factor defined in P1363.

an insecure communication network. Next let's assume that the two entities have decided to use the protocol of EC-ElGamal protocol to implement their secure communication. One basic point to note is unlike ECDH protocol[7], this protocol does not create a common key, but using EC-ElGamal protocol a message $M = (M_1, M_2)$, a point on elliptic curve, can be sent from Bank to Alice and vice versa, as per the Algorithm 2.

Performing the decryption, we reverse the embedding process to produce the message, M, from the point P. It is not trivial to find a point M for the message. Note that the difficulty in obtaining the private-key from the public-key is based on the discrete log problem (DLP) for elliptic curves. The DLP states that given a point $Q \in F_P$ and base point P, it is extremely difficult to find an integer, k, such that Q = kP. Interested readers can find detailed discussion on DLP[7,16].

Simple implementation of ElGamal cryptosystem for elliptic curves: Let the prime number p = 29 and consider an elliptic curve E: $y^2 - x^3 - x + 16 \bmod 29$ defined over F_{29}. Here $E_p(a, b) = E_{29}(-1, 16)$ with the order n of the elliptic curve as: $\#E(F_{29}) = n = 31$, which is prime order. The curve has generator point given by $G \equiv P = (5,7)$ such that the multiples kG of the generator point G are (for $1 \le k \le n-1$) including point O located at infinity.

As an example, suppose the Bank (B) chooses public-key set: $B = k_B G = 17\,G = (1,25)$, where, the secret-key $k_B = 17$, giving rise to its public-key ring (E, G, A), which is kept in the public-key server.

Next let Alice selects a random secret-key $k_A = 23$ such that: $A = k_A G = 23G = (21, 18)$ and computes: $S_{AB} = k_A(k_B G) = 23(17G) = 19\,G = (16, 6) = (x_S, y_S)$.

Encryption: She then selects a message point: $M = (M_1, M_2)$ and computes: $C_{M1} = x_S M_1 \bmod p = 16\,(28) \bmod 29 = 13$ and $C_{M2} = x_S M_2 \bmod p = 6(25) \bmod 29 = 5$ and send (A, C_{M1}, C_{M2}) to the Bank (B).

Decryption: Bank receives the message $(A, C_{M1}, C_{M2}) = (23G, 13, 5)$ and computes:

$S_{BA} = k_B(k_A G) = 17(23G) = 19G = (16, 6) = (x_S, y_S)$ which it uses to recover M, i.e.,

$M_1 = C_{M1}/x_S \bmod p = (13/16) \bmod 29 = 28$ and

$M_2 = C_{M2}/y_S \bmod p = (3/16) \bmod 29 = 25$

and recovers the original message chosen point M (28, 25).

Digital signature scheme: In public-key cryptography, communicating entities can use their private keys to encrypt a message and the resultant ciphertext can be decrypted back to the original message using the individual entity's public key[1]. Evidently, the ciphertext so created can play the role of a Manipulation Detection Code (MDC) accompanying the encrypted message, i.e., provide data integrity protection for the message. Here, the public key decryption process forms a step of verification. Since, it is considered that only the owner of the public key used for the MDC verification could have created the MDC using the corresponding private key. Thus, this usage of public key cryptosystem can model the precisely the property of a signature, a digital signature, for proving the authorship of the message[17-19].

Diffie and Hellman were the first researchers to envision the notion of digital signature scheme with their invention of asymmetric key crypto-algorithm through use of key distribution and shared keys[4]. This systems of key distribution leading to shared keys under public key crypto-algorithm, finally meant that only a single entity is able to create a digital signature of a message which can be verified by anybody, it is easy to settle dispute over who has created the signature. This allows the provision of a security called non-repudiation, which means no denial of connection with message. Non-repudiation is a necessary security requirement in electronic commerce application[1].

The ElGamal digital signature scheme: The ElGamal signature algorithm[6] is similar to the encryption algorithm in that the public-key and private-key have the same form; however, encryption is not the same as signature verification, nor is decryption the same as signature creation as in RSA.

Key generation: Entity A (Alice) selects a random integer k_A from the interval [1, n-1] as her private key and computes $A = k_A G$ as her public key, which she places in the public-key server.

Signing scheme

1. Selects random integer k from the interval [1, n-1]
2. Computes $R = kG = (x_R, y_R)$, where, $r = x_R \bmod n$; if r = 0 then goto step 1;
3. Compute e = h(M), where h is a hash function $\{0,1\}^* \rightarrow F_n$
4. Compute $s = k^{-1}(e+rk_A) \bmod n$; if then goto step 1 (R, s) is the signature message M.

Verifying scheme

1. Verify that s is an integer in [1, n-1] and $R = (x_R, y_R) \in E(F_q)$.
2. Compute $V_1 = sR$.
3. Compute $V_2 = h(M)G+rA$, where, $r = x_R$.
4. Accept if and only if $V_1 = V_2$.

Consistency

$V_1 = sR = skG$ $\{(h(M)+k_Ar) \bmod n\}G$, $V_2 = h(M)G+rA = [h(M)+rK_A]G$. And because G's order is n, kG = jG where, $j \equiv k \bmod n$. Hence, $V_1 = V_2$.

Simple implementation of EC ElGamal signature scheme: Let the prime number p =29 and consider an elliptic curve E: $y^2 \equiv x^3-x+16 \bmod 29$ defined over F_{29}. Here $E_p(a, b) = E_{29}(-1, 16)$ with the order n of the elliptic curve as: $\#E(F_{29}) = n = 31$, which is prime order. The curve has generator point given by $G \equiv P = (5,7)$ such that the multiples kG of the generator point G are (for $1 \le k \le n-1$) including point O located at infinity.

Let Alice selects a random secret-key $k_A = 23$ such that: $A = k_AG = 23G = (21,18)$. Next she chooses integer k in the interval [1, n-1] and computes $R = kG = 13G = (7, 27) = (x_R, y_R)$ and also computes $r = x_R \bmod n = 7 \bmod 31 = 7$.

Suppose now Alice wants to send the message, M = 33 = e, which lies in the interval [1, n-1].

She next computes: $s = k^{-1}(e+k_Ar) \bmod n = (13)^{-1}(33+23*7) \bmod 31 = 3$.

(R, s) = (13 G, 3) is the signature of the message M.

Verifying

1. Compute $V_1 = sR = 3R = 3(13G) = 8G = (21, 11)$
2. Compute $V_2 = h(M)G+rA = 33G +7(23G) =8G (21, 11)$, where, $r = x_R \bmod 31 = 7$
3. Accept signature, since $V_1 = V_2 = 8G = (21, 11)$

PRACTICAL APPLICATION OF EC ELGAMAL (ECEG) ENCRYPTION

For this part let's get real and simulate real application. Assume we have a website: www.bauxicat.com, a music content site that specializes in selling downloadable online music. Now the question is? How can we set up a public-key implementation of ECEG to protect our site, so that only registered parties can download music from the site? We start by setting up an elliptic curve cryptographic system. Let's take as an example our usual communicating partners, Alice and Bob. Alice and Bob love to swap music files stored in their computers amongst themselves.

When Alice became a member of Bauxicat's music content rights management (BMRM), she was prompted to download and install BMRM software on her computer. While registering Alice and BMRM both exchanged their public keys. Bauxicat's public-key point is $k_MG = P_M$. During installation session, the BMRM software sneakily generated a private-key m and which it stealthily hid in bits of files (e.g., blackhole.dll, m3ks.dla and PerBox.key). BMRM music point is mG =M. (For more details on data hiding techniques[20]). In order for Alice to play the latest Storm Boy's music lovekills.wma, she must use her private key to decrypt the file. Bauxicat created the license using the ElGamal public-key cryptosystem using a predefined elliptic curve group over the Galois Field E_p (a, b); Alice's license file can now be used to unlock lovekills.wma, but only in her computer, since the license file is not transferable and hence Bob's computer has no access to this file, since there is no shared key between Bob and Bauxicat. So, whenever she shares her music files including the license files etc. Bob gets very crossed because he can't play lovekills.wma. This is mainly because Bob's computer doesn't know Alice's computer's private-key (i.e., the integer m), therefore, Bob's computer can't decrypt the license file to allow him access and enjoy the music.

Implementation of practical application of ECEG scheme: Here we simulate the real ECEG scheme in practice. Let an elliptic curve group over the Galois Field E_p (a, b) where, p>3 and is prime, be the set of solutions or points P = (x,y) such that ($x, y \in E_p$ (a, b)) that satisfy the

equation: $y^2 = x^3 + ax + b$ for $0 \le x < p$ together with the extra point O called the point at infinity. Here the curve E_p (F_p) is cyclic and any point other than O is a generator of all points on curve. For example, G = (x, y) is a generator point such that the multiples kG of the generator point G (for $1 \le k \le n-1$), which is too huge to show here!

Alice-bob session: In order to communicate, both Alice and Bob selects random integers k_A and k_B, as their private-keys (kept secret). Next they compute $A = k_A G$ and $B = k_B G$ respectively, which are kept in the public-key server, for access to anyone wishing to communicate with them. Alice and Bob may both set up a common session key $k_B(k_A G) = k_A(k_B G) = S_{AB}$, using say ECDH key agreement scheme[7]. In order to send a message $P_w = wG = (w_1, w_2)$ the, Alice and Bob decides to use EC ElGamal crypto-scheme. Alice's message encrypted code is: $(A, x_m m_1, y_m m_2)$, where, A is Alice's public-key, $x_m m_1$ and $y_m m_2$ are the encrypted message points, respectively

Alice-bauxicat session: Now let's take a look at the interaction session between www.Bauxicat.com and Alice. Common session key between Alice and Bauxicat is $S_{AM} = k_M(k_A G) = k_A(k_M G) = (x_S, y_S)$. (Notice that there is no common session key between Alice, Bob and Bauxicat.) Alice license file contains the music file encrypted code: (A, C_{M1}, C_{M2}), where A is Alice's public-key and; $C_{M1} = x_M M_1 \bmod p$ and $C_{M2} = y_M M_2 \bmod p$, are the encrypted music file points, respectively.

The parameters for the elliptic curve featuring in Alice-Bob and Alice-Bauxicat communication are as follows:

p = 125356524088493297886732756415211775455817316865103799201 9849
a = 602060795339016484143949354474116343558714560760195967177227
b = 423523989258759639619145766689597173074955027398957554628 37
\\Order U = N*S with R Prime
U = 125356524088493297886732756415014869282522690604601115185239 2 \\ #E
N = 921739147709509543284799679522168156489137430916184670479 7 \\ #E
S = 136
G = (x, y) \\generator point of order U or R #E, depending on usage.
Where:
x = -308713420171208600284839010152791755784186567349039150579062
y = 494952139410905522093322548267656122025755798264866184081137

k_A = 4708050311399110513517527207693060456300217054473 12 \\A-priv-K
k_B = 5860050311399110513517527207693060456300217054473 4323 \\Bob priv
k_M = 6355050311399110513517527207693060456300217054473 67787 \\Baux priv
m = 1366455345179092321389620005147366685678341839524650183111 \\Music pt

Using the above parameters, we get:

$A = k_A G =$
(6188900424119399290869288778343622048753638117369891775078 29, 992634770636345737979891547952405518008570483614316391410858)

Now Alice's computer had sneakily loaded Bauxicat's license file private key m into memory which automatically computes music point to allow her access to music session:

$M = mG = (M_1, M_2) =$
(3065814374813709620247590402471773338860207891286833160059 50, 1202969469472906489213605300668912414928236030055814778423171)

which is however scrambled (encrypted) using BMRM public key $P_M = k_M G$ and Alice's public-key $A = k_A G$ into a music message point: (P_M, C_{M1}, C_{M2}). Where:

$C_{M1} = x_S M_1 \bmod p =$
3515166532529504884451912660975210439198860581359240307485 38

$C_{M2} = y_S M_2 \bmod p =$
7464293127084404866404996710238393956361140192732042733042 97

Next Alice's computer automatically uses BMRM public-key P_M to compute the session key which allows her access to music file, i.e.,:

$S_{AM} = k_A(k_M G) = k_M(k_A G) = (x_S, y_S)$
=(9516434360044726938559139967542958402699045221570348528486 42,1159393856248092091162667370129418457896536898166111537831211)

and then recovers (decrypts) the music point
$M_1 = x_M = (C_{M1}/x_S) \bmod p =$
3065814374813709620247590402471773338860207891286833160059 50

$M_2 = y_M = (C_{M2}/y_S) \bmod p =$
120296946947290648921360530066891241492823603005581477842317 1

and which maps the music point $M = (x_M, y_M)$ back into the original point M. The crucial parameter here is the x-coordinate, i.e.,:

$x_M =$ 3065814374813709620247590402471773338860207891286833160005950

which is the top secret BMRM magic "content key" that unlock lovekills.wma. Note that if Alice was aware of the license file secret-key m that her computer stealthily generated, she could easily compute the music point M herself and hence unlock lovekills.wma which she could now securely share with Bob.

NB: The number crunching done above is a game played by Big Boys, so we need number crunching software to crunch our computation undertaken above. The number crunching here was done using PARI a Free, open source, number cruncher[21].

Signature verification: Before Alice's computer can start to interact with BMRM software she wants to make sure that the music file is authentic before commencing music playing session. The set of values (R, s); which is the signature message for music point M deployed in Alice's computer was coded using the music-point private-key m = h(M) as in the previous example of EC-ElGamal signature algorithm performed earlier. To verify the authenticity of the music file she must do the following operation using ElGamal signature algorithm:

She must first verify that s is an integer in [1, n-1] and $R = (x_R, y_R) \in E(F_q)$; where:

$R = kG = (x_R, y_R)$,
= (387799827527934506622574480845692292854828456171775306847445,5562480564581014259609984504518868547517238934866799795 73643)

and
s = 54790562016533606530399971645141618833818076312835567036 59

then her computer automatically computes:
$V_1 = sR = s(kG)$
= (977649381406879411883749316393418288235757455473770256495338,9718931750932544 2877581365555121336023272006247861364756 3976)

and
$V_2 = mG = r(k_AG)$
= (977649381406879411883749316393418288235757455473770256495338,9718931750932544 2877581365555121336023272006247861364756 3976)

and verifies that: $V_1 = V_2$. So her computer goes ahead and commences the music playing session.

CONCLUSIONS AND FUTURE WORK

We have shown that elliptic curve ciphers require less computational power, memory and communication bandwidth giving it a clear edge over the traditional crypto-algorithms. To date elliptic curve cryptography is gaining wide acceptance, especially in wireless and hand-held devices, when compared to the conventional cryptosystems (DES, RSA, AES, etc.), which tend to be power hungry. However, while the performance advantages are impressive with ECC, the data security industry need to ensure that the security system, using elliptic curve algorithm has been studied extensively in the public forum and, also specified by major standards worldwide. But we think that elliptic curve cryptography is here today and is without question the next generation of public-key cryptography of choice. The ElGamal cryptosystem, as we have already seen, requires a high level of mathematical abstraction to implement. One significant practical problem if this system is to be useful is how can it be packaged in a user-friendly way so that developers can incorporate it into their applications with minimal knowledge of its inner workings-and that is subject of next work.

REFERENCES

1. Kefa, R., 2004. Data Security and cryptographic techniques-A review. PJIT., 3: 106-132.
2. Menezes, A, P. Van Oorschot and S. Vanstone, 1997. Handbook of Applied Cryptography. CRC Press.
3. Rivest, R., A. Shamir and L. Adleman, 1978. A method for obtaining digital signatures and public key cryptosystems. Comm. ACM., 21: 120-126.
4. Diffie, W. and M.E. Hellman, 1976. Multi-user cryptographic techniques. Proceedings of AFIPS National Computer Conference, pp: 109-112.
5. Rabin, M.O., 1979. Digital signature and public-key functions as intractable as factorization. MIT Laboratory of Computer Science, Technical Report, MIT/LCS/TR-212, Jan.

6. ElGamal, T., 1985. A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Info. Theory, 31: 469-472.
7. Rabah, K., 2005. Theory and implementation of elliptic curve cryptography. J. Applied Sci., 5: 604-633.
8. Miller, S., 1986. Use of elliptic curves in cryptography. In CRYPTO '85: 417-426.
9. Koblitz, N., 1987. Elliptic curve cryptosystems. Mathematics of Computation, 48: 203-209.
10. FIPS 113, National Institute of Standards and Technology (formerly National Bureau of Standards), 1985. FIPS PUB 113: Computer Data Authentication, May 30.
11. Koblitz, N., 1994. A course in number theory and cryptography. Springer Verlag.
12. Menezes, A.J., 1993. Elliptic Curve Public Key Cryptosystems. Kluwer Academic Publishers.
13. Odlyzko, A., 1984. Discrete Logarithms in Finite Fields and Their Cryptographic Significance. In: Advances in Cryptology Eurocrypt'84, Springer-Verlag, pp: 224-314.
14. Schoof, R., 1998. Elliptic curves over finite fields and computation of square roots mod p. Math. Comp., 44: 483-494.
15. ECB (Elliptic Curve Builder)-Ellipsa-http://www.ellipsa.net/index.html
16. Pohlig, S.C. and M.E. Hellman, 1978. An improved algorithm for computing logarithms over GP(p) and its cryptographic significance. IEEE Trans. Info. Theory, 24: 106-110.
17. Diffie, W., P. van Oorschot and M. Wiener, 1992. Authentication and authenticated key exchanges. Designs, Codes and Cryptography, 2: 107-125.
18. Basham, D. Johnson and T. Polk, 1999. Representation of Elliptic Curve Digital Signature Algorithm (ECDSA) Keys and Signatures in Internet X.509 Public-Key Infrastructure Certificates, Internet Draft, June 1999, http://www. ietf.org.
19. ANSI X9.62, 1999. The Elliptic Curve Digital Signature Algorithm (ECDSA). American Bankers Association.
20. Rabah, K., 2004. Steganography-The art of hiding data. ITJ., 3: 245-269,
21. PARI/GP-Computational Number Theory: http://pari.math.u-bordeaux.fr/