Available online at www.elixirpublishers.com (Elixir International Journal)

Computer Science and Engineering

Elixir Comp. Sci. & Engg. 38 (2011) 4409-4413

ISSN: 2229-712X

The security of elliptic curve cryptography over RSA cryptography

G. Jai Arul Jose and C. Sajeev
Sathyabama University, Chennai.

ARTICLE INFO

Article history:
Received: 22 August 2011;
Received in revised form:
26 August 2011;
Accepted: 31 August 2011;

Keywords
Cryptography,
ECC,
RSA,
Public Key.

ABSTRACT

Elliptic curves are algebraic curves that have been studied by many mathematicians since the seventeenth century. In 1985, Neal Koblitz and Victor Miller independently proposed public key cryptosystems using the group of points on an elliptic curve. The elliptic curve cryptosystem (ECC) was thus created. Since then, numerous researchers and developers have spent years researching the strength of ECC and improving techniques for its implementation. Secure applications in smart cards present implementation challenges particular to the platform's memory, bandwidth, and computation constraints. ECC's unique properties make it especially well suited to smart card applications. This paper describes the Elliptic Curve Cryptography algorithm and its security over RSA algorithm.

Introduction

Information security is essential for today's world since, for profitable and legal trading, confidentiality, integrity and non-repudiability of the associated information are necessary. This can be done using cryptographic systems. Integrated cryptographic systems satisfy all the above-mentioned requirements. Desired properties of a secure communication system may include any or all of the following:

Confidentiality Only an authorized recipient should be able to extract the contents of the encoded data, in part or whole.

Connection Confidentiality: The protection of all user data on a connection.

Connectionless Confidentiality: The protection of all user data in a single data block

Selective-Field Confidentiality: The confidentiality of selected fields within the user data on a connection or in a single data block.

Traffic Flow Confidentiality: The protection of the information that might be derived from observation of traffic flows.

Integrity The recipient should be able to establish if the message has been altered during transmission.

Connection Integrity with Recovery: Provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data within an entire data sequence, with recovery attempted.

Connection Integrity without Recovery: As above, but provides only detection without recovery.

Selective-Field Connection Integrity: Provides for the integrity of selected fields within the user data of a data block transferred over a connection and takes the form of determination of whether the selected fields have been modified, inserted, deleted, or replayed.

Connectionless Integrity: Provides for the integrity of a single connectionless data block and may take the form of detection of data modification. Additionally, a limited form of replay detection may be provided.

Selective-Field Connectionless Integrity: Provides for the integrity of selected fields within a single connectionless data block; takes the form of determination of whether the selected fields have been modified.

Authentication The recipient should be able to identify the sender, and verify that the purported sender actually sent the message.

Peer Entity Authentication: Used in association with a logical connection to provide confidence in the identity of the entities connected.

Data Origin Authentication: In a connectionless transfer, provides assurance that the source of received data is as claimed.

Non-Repudiation The sender should not be able to deny sending the message, if he actually did send it.

Nonrepudiation, Origin: Proof that the message was sent by the specified party.

Nonrepudiation, Destination: Proof that the message was received by the specified party.

Anti-replay The message should not be allowed to be sent to multiple recipients, without the sender's knowledge.

Proof of Delivery The sender should be able to prove that the recipient received the message.

A brief history of cryptography

Cryptography has been in use for centuries now, and the earliest ciphers were either used transposition or substitution, and messages were encoded and decoded by hand.

However, these schemes satisfied only the basic requirement of confidentiality.

In more recent times, with the invention of processing machines, more robust algorithms were required, as the simple ciphers were easy to decode using these machines, and moreover they did not have any of the afore mentioned properties.

Secure data communication became a necessity in the 20t h century and a lot of research was done in this field by government agencies, during and following the world-wars.

The most famous machine of this time, Enigma was an electromechanical device which was used by the German Army.

Tele: +91 9443275341
E-mail addresses: jaiaruljose@yahoo.co.in

Symmetric Algorithms

The first secret key-based cryptographic algorithms worked on the symmetric algorithms. They assumed that both communicating parties shared some secret information, which was unique to them, much like the older One Time Pads. Using this secret information also called a key, the sender encrypted1 the data, and the recipient was able to decrypt. Suppose Alice wants to send a message m to Bob, and assume that they both have already shared a key k. Alice encrypts m using the shared key k to get the cipher text.

$$C_{(k, m)} = E_k(m) \quad \text{----------} \quad (1)$$

Bob can then decrypt this message using his copy of the key k, and extract the original message m.

$$D_k(C_{(k,m)}) = D_k(E_k(m)) = m \quad \text{----------} \quad (2)$$

This technique though simple and easy to implement, has obvious drawbacks, some of which are listed here:

- A shared secret key must be agreed upon by both parties.
- If a user has n communicating partners, then n secret keys must be maintained, one for each partner.
- Authenticity of origin or receipt cannot be proved because the secret key is shared.

Management of the symmetric keys becomes problematic.

Asymmetric Key Cryptography

Asymmetric Key Cryptography otherwise known as Public Key Cryptography. The concept of Public Key cryptography (PKC) was first introduced by Diffie and Hellman in 1976, in their seminal paper, New Directions in Cryptography. This paper also addressed the issue of key exchange, based on the intractability of the discrete logarithm problem. In a public key cryptosystem, each user has a pair of keys, one published publicly, known as the public key, and the other known as a private key, is stored in a secure location. Public key cryptosystems rely on the existence of a trapdoor function, which makes decoding possible given the knowledge of the private key corresponding to the public key for encryption. Considering a case analogous to the one described in the case of symmetric keys, whereby Alice wishes to send a message m to Bob, the following steps will accomplish the task:

1. Alice passes the message m and Bob's public key PUB to an appropriate encryption algorithm to construct the encrypted message.

$$C(PUB, m) = EPUB(m) \quad \text{----------} \quad (3)$$

2. Alice transmits the encoded message to Bob.
3. Bob decrypts the encrypted message received by him, using his private key PRB and the appropriate decryption algorithm.

$$DPRB(C(PUB, m)) = DPRB(EPUB(m)) = m \quad \text{----------} \quad (4)$$

Bob is assured that the data he received is not tampered with or leaked, as only his private key can decrypt the data. Similarly Bob can send data to Alice using her public key A. The PKC scheme also satisfies the Non-Repudiation and Authenticity by using innovative techniques such as Digital Signatures.

Smart Cards

Smart cards are small, portable, tamper resistant devices providing users with convenient storage and processing capability. Because of their unique form factor, smart cards are proposed for use in a wide variety of applications such as electronic commerce, identification, and health care. For many of these proposed applications, cryptographic services offered by digital signatures would be required. To be practical for widespread use, however, smart cards also need to be inexpensive. The smart card is amenable to cryptographic implementations for several reasons. The card contains many security features that enable the protection of sensitive cryptographic data and provide for a secure processing environment.

Smart card, chip card, or integrated circuit(s) card (ICC), is defined as any pocket-sized card with embedded integrated circuits. Although there is a diverse range of applications, there are two broad categories of ICCs. Memory cards contain only non-volatile memory storage components, and perhaps some specific security logic. Microprocessor cards contain memory and microprocessor components. The standard perception of a smart card is a microprocessor card of credit-card dimensions (or smaller, e.g. the GSM SIM card) with various tamper-resistant properties (e.g. a secure cryptoprocessor, secure file system, human-readable features) and is capable of providing security services (e.g. confidentiality of information in the memory). Not all chip cards contain a microprocessor (eg. The memory cards), therefore not all chip cards are necessarily also smart cards.

Application

Smartcards were invented and patented in early 1970's, but the first mass use of smartcards was made in 1983, in French pay-phones. A major boom in Smartcard use came in 1990's, with their introduction as SIM cards in mobile phones. They are commonly in use now. Smartcard technology is an industry standard defined and controlled by the Joint Technical Committee 1(JCT1) of the International Standards Organization (ISO) and the International Electronic Committee (IEC). The series of international standards ISO/IEC 7816, introduced in 1987 with the latest update in 2003, defines various aspects of a smartcard, including physical characteristics, physical contacts, electronic signals and transmission protocols, commands, security architecture etc. Smartcards don't contain a battery, and become active only when connected with a card reader. When connected, after a reset sequence, the card remains passive, waiting to receive a command request from a client (host) application.

Smartcards can be contactless (based on Radio Frequency ID tags), or can have a standard 8-pin contact. Today smartcards are used for various applications all over the world including Banking, Medical records, GSM SIM cards, Identification and cryptographic services. They have storage and processing capability, and are convenient to carry around, and as the processing power and memory capacity of smartcards improves, their range of applications is expanding as well.

Java Card technology adapts the Java platform for use on smart cards and other devices whose environments are highly specialized, and whose memory and processing constraints are typically more severe than those of J2ME devices. On a Java Card platform multiple applications from di erent vendors can co-exist securely.

Java Cards are capable of running Java byte codes, and up to 3 applets at once. A major advantage of running downloadable applets is that in case of a security breach, the user only needs to download and write a new applet onto his/her Java Card, instead of destroying the Card and waiting for a new one to be shipped to him/her. A typical Java Card device has an 8- or 16-bit CPU running at 5 MHz, with 2K of RAM and more than 32K of non-volatile memory (EEPROM or Flash). High performance smartcards come with a separate processor and cryptographic chip and memory for encryption, and some come with a 32-bit CPU.

The RSA cryptographic algorithm

RSA stands for Rivest, Adleman and Shamir, who devised this algorithm in 1977 at MIT. RSA is the most widely used public-key encryption scheme today. The US patent on the RSA algorithm expired in 2000, but as the algorithm was already published prior to patent application, it precluded patents elsewhere. The security of the RSA cryptosystem is based on two mathematical problems: the problem of factoring very large numbers, and the RSA problem. Both of these problems are hard, i.e., no efficient algorithm exists for solving them.

The RSA problem is defined as the task of taking eth roots modulo a composite n: recovering a value m such that $m^e = c \pmod n$, where (e, n) is the public key and c is the ciphertext. Currently the most promising approach to solving the RSA problem is to factor the modulus n. With the ability to recover prime factors, an attacker can compute the secret exponent d from a public key (e, n), then decrypt c using the standard procedure. To accomplish this, an attacker factors n into p and q, and computes $(p-1)(q-1)$ which allows the determination of d from e. No polynomial-time method for factoring large integers on a classical computer has yet been found, but it has not been proven that none exists.

Key Generation

Suppose Alice and Bob are communicating over an insecure transmission medium, and Alice wants Bob to send her a private (or secure) message.

Using RSA, Alice will take the following steps to generate a public key and a private key:

1. Choose two large numbers prime numbers p and q such that $p \neq q$, randomly and independent of each other.
2. Compute $n = pq$ --------- (5)
3. Compute the Euler Totient Function $\varphi(n)=(p-1)(q-1)$ -------- (6)
4. Choose an integer e such that $1 < e < \varphi(n)$ which is co prime to $\varphi(n)$.
5. Compute d such that

$$de \equiv 1 \pmod{\varphi(n)} \quad \text{---------- (7)}$$

The public key consists of:

- n, the modulus, and
- e, the public exponent (sometimes encryption exponent)

The private key consists of:

- n, the modulus, and
- d, the private exponent (sometimes decryption exponent), which must be kept secret.

Alice transmits the public key to Bob, and keeps the private key secret. p and q are sensitive since they are the factors of n, and allow computation of d given e.

Encrypting Messages

Suppose Bob wishes to send a message M to Alice. He turns M into a number $m < n$, using some previously agreed-upon reversible protocol known as a padding scheme. Bob now has m, and knows n and e, which Alice has announced. He then computes the ciphertext c corresponding to m:

$c = m^e \pmod n$ -------- (8)

Bob Transmits C to Alice

Decrypting Messages

Alice receives c from Bob, and knows her private key d. She can recover m from c by the following procedure:

$m = c^d \pmod n$ -------- (9)

Given m, she can recover the original message M. The decryption procedure works because

$c^d \equiv (m^e)^d \equiv m^{ed} \pmod n$ -------- (10)

Now, since

$ed = 1 \pmod{p-1}$ -------- (11)

and

$$ed = 1 \pmod{q-1}, \quad \text{------- (12)}$$

Fermat's little theorem yields

$$m^{ed} \equiv m \pmod p \quad \text{------- (13)}$$

and

$$m^{ed} \equiv m \pmod q. \quad \text{------- (14)}$$

Since p and q are distinct prime numbers, applying the Chinese remainder theorem to these two congruences yields

$$m^{ed} \equiv m \pmod{pq}. \quad \text{------- (15)}$$

Thus,

$$c^d \equiv m \pmod n. \quad \text{-------- (16)}$$

Issues with RSA

As of 2005, the largest number factored using general purpose methods is 663-bits long, using state of the art distributed methods. Experts feel that 1024-bit keys may become breakable in the near future (though disputed). 256-bit length keys are breakable in a few hours using a personal computer. The current recommended key-length is 2048-bits. Though this length may be insignificant for most personal computers in use, it causes low processing power portable devices like smartcards to become inefficient. There are constraints on processor word length, available memory and clock speeds in these devices.

As the need for portable and secure identification slowly becomes a necessity, and as RSA key sizes will increase in proportion to the processor power available, there arises a need to devise a scheme which provides the same level of cryptographic security with smaller key lengths. ECC is one such scheme, described in the following section.

Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography is an approach to public-key cryptography, based on elliptic curves over finite fields. The technique was first proposed individually by Neal Koblitz and Victor Miller in 1985. The ECC is based on the Elliptic Curve Discrete Logarithm problem, which is a known NP-Hard problem. An elliptic curve is defined by the equation,

$$y^2 + xy = x^3 + ax + b \quad \text{------ (17)}$$

A brief introduction to the mathematics required for elliptic curves, and their usage is given in the following section.

Operations on Elliptic Curve

The crucial property of an elliptic curve is that we can define a rule for adding two points which are on the curve, to obtain a third point which is also on the curve. This addition rule satisfies the normal properties of addition. The points and the addition law form a finite Abelian group. For addition to be well defined for any two points, we need to include an extra zero point 0, which does not satisfy the elliptic curve equation. 0 is taken to be a point of the curve. The order of the curve is the number of distinct points on the curve, including the zero point. Having defined addition of two points, we can also define multiplication kP where k is a positive integer and P is a point as the sum of k copies of P.

$$\therefore 2P = P + P$$

Cryptography

Alice, Bob, Cathy, David . . . agree on a (non-secret) elliptic curve and a (non-secret) fixed curve point F. Alice chooses a secret random integer A_k which is her secret key, and publishes the curve point $A_P = A_kF$ as her public key. Bob, Cathy and David do the same. Now suppose Alice wishes to send a message to Bob. One method is for Alice to simply compute

A_kB_P and use the result as the secret key for a conventional symmetric block cipher (say DES). Bob can compute the same number by calculating B_kA_P , since

$B_kA_P = B_k \cdot (A_kF) = A_k \cdot (B_kF) = A_kB_P$ --------- (18)

The security of the scheme is based on the assumption that it is difficult to compute k given F and kF.

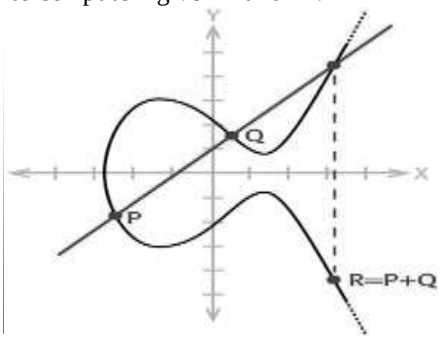

Fig. 1: Elliptic curve showing the operation P + Q = R

Choosing the Fixed Curve

A finite field is first chosen. If the field is GF(p) where p is a large prime, the *xy* term is omitted, leaving us with (see Equation 17)

$$y^2 = x^3 + ax^2 + b, \text{ where } 4a^3 + 27b^2 \neq 0 \text{ ------- (19)}$$

If the field is GF(2^m), then we include the *xy* term to get

$$y^2 + xy = x^3 + ax^2 + b, \text{ where } b \neq 0 \text{ ------- (20)}$$

Fields GF(p^m) with both p > 2 and m > 1 are not considered here.

Choosing the Fixed Point

For any point P on a elliptic curve in the GF(p^m),

$$\lim_{k \to \infty} kP \to 0$$

For some a and b, b > a, we will have aP = bP. This implies cP = 0 where c = b − a. The least c for which this is true is called the order of the point, and c must divide the order of the curve. For good security, the curve and fixed point are chosen so that the order of the fixed point F is a large prime number. This is determined from the order of the curve, which is done from Schoof's Algorithm. For good security, the order of the fixed point should also satisfy the MOV condition to prevent certain possible attacks. As far as is known, with the above provisions, if the order of the fixed point F is an n-bit prime, then computing k from kF and F takes roughly $2^{\Pi/2}$ operations. This is what makes the use of elliptic curves attractive – it means that public keys and signatures can be much smaller than with RSA for the same predicted security.

Advantage of ECC over RSA

Security

The main advantage ECC has over RSA is that the basic operation in ECC is point addition, which is known to be computationally very expensive. This is one of the reasons why it is very unlikely that a general sub-exponential attack on ECC will be discovered in the near future, though ECC has a few attacks on a few particular classes of curves. These curves can be readily distinguished and can be avoided. On the other hand, RSA already has a known sub-exponential attack which works in general. Thus, to maintain the same degree of security, in view of rising computing power, the number of bits required in the RSA generated key pair will rise much faster than in the ECC generated key pair, as seen in table 1. Menezes and Jurisic, in their paper, said that to achieve reasonable security, a 1024-bit modulus would have to be used in a RSA system, while 160-bit modulus should be sufficient for ECC.

Table 1: Comparison of strength of RSA and ECC

Time to break (in MIPS-years)	RSA key-size (in bits)	ECC key-size (in bits)
10^4	512	106
10^8	768	132
10^{11}	1024	160
10^{20}	2048	210
10^{78}	21000	600

Most attacks on ECC are based on attacks on similar discrete logarithm problems, but these work out to be much slower due to the added complexity of point addition. Also, mathods to avoid each of the attacks have already been designed.

Space Requirements

Due to increasing computation required for higher bit encryption, more transistors are required onboard the smart card to perform the operation. This leads to an increase in area used for processor. Using ECC, the number of transistors can be cut back on since the numbers involved are much smaller than an RSA system with as similar-level security. Also, the bandwidth requirement for both of the systems is the same when the messages to be signed are long, but ECC is faster when the messages are short. This is more relevant, since PKC is used to transmit mostly short messages, e.g. session ids.

Efficiency

Both methods can be made faster – in RSA, by using smaller public exponent, though this holds a greater security risk and in ECC, some results of the calculation can be stored beforehand.

Certicom, a Canadian company, has been studying and promoting the ECC system since the early '80s. Some of their results of fast implementations of ECC compared to RSA are given in table 2.

Table 2: Comparison of RSA and ECC

Function	ECC 163-bit (in ms)	RSA 1024-bit (in ms)
Key Generation	3.8	4708.3
Sign	2.1(ECNRA) 3.0(ECDSA)	228.4
Verify	9.9(ECNRA) 10.7(ECDSA)	12.7

Conclusion

Elliptic Curve Cryptosystems provide the highest strength per bit of any cryptosystem known today. ECC is faster, and occupies less memory space than an equivalent RSA system. This means that it is suitable for constrained environments, especially in smartcards, where fast operations are necessary. The difference in the key-sizes between ECC and RSA will grow exponentially to maintain the same relative strength as compared to the average computing power available. With a 160-bit modulus, an elliptic curve system offers the same level of cryptographic security as DSA or RSA with 1024-bit moduli. The smaller key sizes result in smaller system parameters, smaller public-key certificates, bandwidth savings, faster implementations, lower power requirements, and smaller hardware processors.

RSA has been well-researched and has been the topic of many seminal theses. In fact, the cryptographic use for elliptic curves was only discovered in the process of finding out new attacks on the RSA system.

Reference:

[1] Diffie, W., and Hellman, M. "Multiuser Cryptographic Techniques." IEEE Transactions on Information Theory, November 1976.

[2] Rivest, R.; Shamir, A.; and Adleman, L. "A Method for Obtaining Digital Signatures and Public Key Cryptosystems." Communications of the ACM, February 1978.

[3] http://www.rsasecurity.com/rsalabs

[4] www.certicom.com

[5] Seroussi, G.; and Smart, N. Elliptic Curves in Cryptography. Cambridge: Cambridge University Press, 1999.

[6] Enge, A. Elliptic Curves and Their Applications to Cryptography. Norwell, MA: Kluwer Academic Publishers, 1999.

[7] Fernandes, A. "Elliptic Curve Cryptography." Dr. Dobb's Journal, December 1999.

[8] The Elliptic Curve Cryptosystem for Smartcards, the seventh in a series of ecc white papers, a certicom white paper published: may 1998.

[9] Aleksandar Jurisic and Alfred J. Menezes. Elliptic curves and cryptography. Dr. Dobb's Journal, 1997. Zoreda, Jose Luis and Oton, Jose Manuel, Smart Cards, Artech House,

[10] Monk, J. Thomas and Dreifus, Henry N., Smart Cards: A Guide to Building and Managing Smart Card Applications, John Wiley & Sons, 1997.

[11] Miller, V., "Uses of elliptic curves in cryptography," Advances in Cryptology, CRYPTO '85 - Lecture Notes in Computer Science, Volume 218, Springer-Verlag, pages 417-426, 1986.

[12] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. IEEE Transactions on Information Theory, IT-22(6):644–654, 1976.