Elliptic Curve Cryptographic Algorithm

Prof. santoshi Ketan pote
Electronics and communication ,
SNDT women's University
Usha Mittal Institute of Technology
Mumbai, India
santoshipote@yahoo.co.in

Abstract

This paper deals with an implementation of Elliptic Curve Cryptography Algorithm. The implementation includes Diffie Hellman Key Exchange and the Digital Signature Algorithm. This paperr gives an overview of Elliptic Curve Cryptography algorithm. Cryptography (or cryptology) from Greek word *kryptos*, "hidden, secret"; and graph, "writing" is the practice and study of hiding information. Modern cryptography intersects the disciplines of mathematics, computer science, and engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce. Cryptology prior to the modern age was almost synonymous with encryption, the conversion of information from a readable state to nonsense. The sender retained the ability to decrypt the information and therefore avoid unwanted persons being able to read it. The secret key cryptography and public key cryptography are the two main types of cryptography.
RSA is the most prominent algorithm used in public key cryptography techniques for encryption and digital signatures. Over the years, the key lengths for RSA have been increasing. This puts considerable burden on RSA. Another public key cryptography technique is gaining popularity in the last few years. It is called as Elliptic Curve Cryptography (ECC).
The main difference between RSA and Elliptic Curve Cryptography is that unlike RSA, Elliptic Curve Cryptography offers the same level of security for smaller key sizes. Elliptic Curve Cryptography is highly mathematical in nature. While conventional public-key cryptosystems (RSA, Diffie - Hellman and DSA) operate directly on large integers, an Elliptic Curve Cryptography operates over points on an elliptic curve.

Keywords: cryptography, RSA, Elliptic Curve.

I. Introduction

Elliptic Curve Cryptography (ECC) is a public key cryptography. In public key cryptography each user or the device taking part in the communication generally have a pair of keys, a public key and a private key, and a set of operations associated with the keys to do the cryptographic operations. Only the particular user knows the private key whereas the public key is distributed to all users taking part in the communication. Some public key algorithms may require a set of predefined constants to be known by all the devices taking part in the communication. 'Domain parameters' in Elliptic Curve Cryptography are an example of such constants.
The mathematical operations of Elliptic Curve Cryptography are defined over the elliptic curve $y^2 = x^3 + ax + b$, where $4a^3 + 27b^2 \neq 0$. Each value of 'a' and 'b' gives a different elliptic curve. All points (x, y) which satisfy the above equation plus a point at infinity lie on the elliptic curve. One main advantage of Elliptic Curve Cryptography is its small key size. A 160-bit key in Elliptic Curve cryptography is considered to be as secure as 1024-bit key in RSA.

A. History of Elliptic Curve Cryptography

Elliptic curves were proposed for use as the basis for discrete logarithm-based cryptosystems almost 20 years ago, independently by Victor Miller of IBM and Neal Koblitz of the University of Washington. At that time, elliptic curves were already being used in various cryptographic contexts, such as integer factorization and primarily proving.

B. Discrete Logarithm Problem

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

II. Introduction to cryptography

This introduction covers all the terms, and definitions that are needed to understand Elliptic Curve Cryptography and Cryptography in general.

Traditionally, ciphers have used information contained in secret decoding keys to code and decode messages. The process of coding plaintext to create cipher text is called encryption and the process of decoding cipher text to produce the plaintext is called decryption. There are two types of encryption: symmetric key encryption and public (asymmetric) key encryption. Symmetric key and public key encryption are used, often in conjunction, to provide a variety of security functions for network and information security.

A. Asymmetric Key Encryption

Encryption algorithms that use the same key for encrypting and for decrypting information are called symmetric-key algorithms. The symmetric key is also called a secret key because it is kept as a shared secret between the sender and receiver of information. Otherwise, the confidentiality of the encrypted information is compromised. Symmetric key encryption is much faster than public key encryption, often by 100 to 1,000 times.

B. **Public Key Encryption**

Encryption algorithms that use different keys for encrypting and decrypting information are most often called public-key algorithms but are sometimes also called asymmetric key algorithms. Public key encryption requires the use of both a private key (a key that is known only to its owner) and a public key (a key that is available to and known to other entities on the network). A user's public key, for example, can be published in the directory so that it is accessible to other people in the organization. The two keys are different but complementary in function. Information that is encrypted with the public key can be decrypted only with the corresponding private key of the set. The algorithm used is RSA.

C. **Secret Key Exchange**

For symmetric key cryptography to work for online communications, the secret key must be securely shared with authorized communicating parties and protected from discovery and use by unauthorized parties. Public key cryptography can be used to provide a secure method for exchanging secret keys online. Two of the most common key exchange algorithms are the following:

- Diffie-Hellman Key Agreement algorithm
- RSA key exchange process

Both methods provide for highly secure key exchange between communicating parties. An intruder who intercepts network communications cannot easily guess or decode the secret key that is required to decrypt communications. The exact mechanisms and algorithms that are used for key exchange varies for each security technology. In general, the Diffie-Hellman Key Agreement algorithm provides better performance than the RSA key exchange algorithm.

III. RSA

The RSA algorithm was publicly described in 1978 by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT, the letters RSA are the initials of their surnames, listed in the same order as on the paper. The RSA algorithm involves three steps: key generation, encryption and decryption. The keys for the RSA algorithm are generated the following way:

1. Choose two distinct prime numbers p and q.
2. For security purposes, the integers p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.
3. Compute $n = pq$. n is used as the modulus for both the public and private keys
4. Compute $\varphi(n) = (p-1)(q-1)$, where φ is Euler's totient function.
5. Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e,\varphi(n)) = 1$, i.e. e and $\varphi(n)$ are coprime. e is released as the public key exponent.
6. Determine $d = e^{-1} \bmod \varphi(n)$; i.e. d is the multiplicative inverse of e mod $\varphi(n)$. This is often computed using the extended Euclidean algorithm.d is kept as the private key exponent.

The public key consists of the modulus n and the public (or encryption) exponent e. The private key consists of the private (or decryption) exponent d which must be kept secret.

A. Encryption and Decryption

Alice transmits her public key (n,e) to Bob and keeps the private key secret. Bob then wishes to send message M to Alice. He first turns M into an integer $0 < m < n$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext c corresponding to

$c = m^e \pmod n$.

Alice can recover m from c by using her private key exponent d via computing

$m = c^d \pmod n$.

Given m, she can recover the original message M.

by reversing the padding scheme.

IV. Diffie Hellman Key Exchange

Public key cryptography was first publicly proposed in 1975 by Stanford University researchers Whitfield Diffie and Martin Hellman to provide a secure solution for confidentially exchanging information online. Fig. 1 shows the basic Diffie-Hellman Key Agreement process.

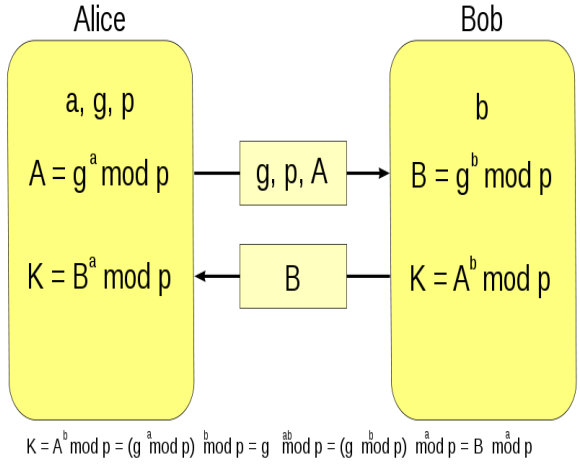

Fig. 1. Diffie-Hellman Key Agreement

Diffie-Hellman key agreement is not based on encryption and decryption, but instead relies on mathematical functions that enable two parties to generate a shared secret key for exchanging information confidentially online. Diffie-Hellman key exchange is widely used with varying technical details by Internet security technologies, such as IPSec and TLS, to provide secret key exchange for confidential online communications.

V. Elliptic Curve Cryptography Arithmetic

Elliptic Curve Cryptography involves mathematics of a different kind than the type used in other cryptographic algorithms.
The fig.2. Shows a hierarchical model of Elliptic Curve Cryptography. Elliptic Curve Cryptography is divided into three kinds of fields. Field over real numbers, field over prime numbers, and a binary Galois field. The main operations in Elliptic Curve Cryptography are Point Multiplication, Point Addition and Point Doubling. These operations can be performed over all kinds of fields, however this implementation deals only with the prime field, which is better suited for software implementation purposes.

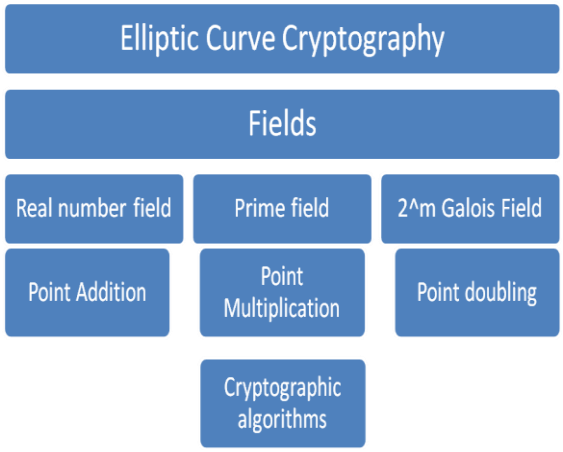

Fig. 2. Hierarchical Elliptic Curve Cryptography model

VI. Implementation of Elliptic Curve Cryptography Diffie Hellman Key Exchange (ECDH)

Public key cryptography was first publicly proposed in 1975 by Stanford University researchers Whitfield Diffie and Martin Hellman to provide a secure solution for confidentially exchanging information online. This paper looks at the implementation of the Diffie Hellman algorithm using Elliptic Curve Cryptography.

A.. Diffie-Hellman Key Agreement

Diffie-Hellman key agreement is not based on encryption and decryption, but instead relies on mathematical functions that enable two parties to generate a shared secret key for exchanging information confidentially online. Essentially, each party agrees on a public value *g* and a large prime number *p*. Next, one party chooses a secret value *x* and the other party chooses a secret value *y*. Both parties use their secret values to derive public values, *g* x mod *p* and *g* y mod *p*, and they exchange the public values. Each party then uses the other party's public value to calculate the shared secret key that is used by both parties for confidential communications. A third party cannot derive the shared secret key because they do not know either of the secret values, *x* or *y*.

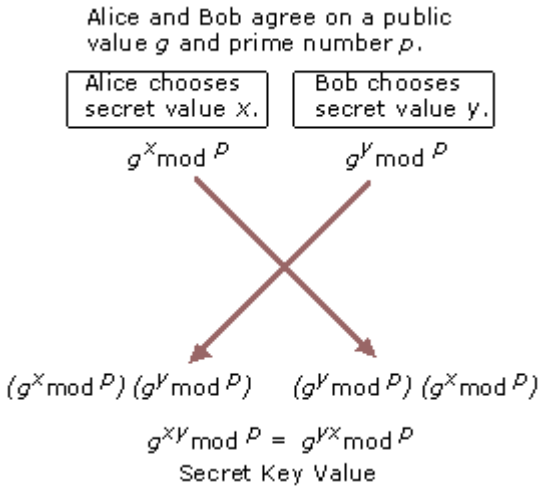

Fig.3. shows the basic Diffie-Hellman Key Agreement process.

For example, Alice chooses secret value *x* and sends the public value *g* x mod *p* to Bob. Bob chooses secret value *y* and sends the public value *g* y mod *p* to Alice. Alice uses the value *g* xy mod *p* as her secret key for confidential communications with Bob. Bob uses the value *g* yx mod *p* as his secret key. Because *g* xy mod *p* equals *g* yx mod *p*, Alice and Bob can use their secret keys with a symmetric key algorithm to conduct confidential online communications. The use of the mod function ensures that both parties can calculate the same secret key value, but an eavesdropper cannot. An eavesdropper can intercept the values of *g* and *p* but because of the extremely difficult mathematical problem created by the use of a large prime number in mod *p*, the eavesdropper cannot feasibly calculate either secret value *x* or secret value *y* . The secret key is known only to each party and is never visible on the network.

B.Elliptic Curve Domain parameters

Apart from the curve parameters a and b, there are other parameters that must be agreed by both parties involved in secured and trusted communication using Elliptic Curve Cryptography. These are domain parameters. The domain parameters for prime fields and binary fields are described below. The generation of domain parameters is out of scope of this paper. Generally the protocols implementing the Elliptic Curve Cryptography specify the domain parameters to be used.

C. Domain parameters for EC over field Fp

The domain parameters for Elliptic curve over Fp are p, a, b, G, n and h. p is the prime number defined for finite field Fp. a and b are the parameters defining the curve y2 mod p= x3 + ax + b mod p. G is the generator point (xG, yG), a point on the elliptic curve chosen for cryptographic operations. n is the order of the elliptic curve. The scalar for point multiplication is chosen as a number between 0 and n – 1. h is the cofactor where h = #E(Fp)/n. #E(Fp) is the number of points on an elliptic curve.

Hasse Theorem states that:

It is possible to define an addition rule to add points on *E*. The addition rule is specified as follows:

1. Rule to add the point at infinity to itself:

O+*O* = *O*:

2. Rule to add the point at infinity to any other point:

(x,y)+ *O*=*O*+(x,y)

D. Elliptic Curve Diffie Hellman (ECDH)

ECDH is a key agreement protocol that allows two parties to establish a shared secret key that can be used for private key algorithms. Both parties exchange some public information to each other. Using this public data and their own private data these parties calculates the shared secret. Any third party, who doesn't have access to the private details of each device, will not be able to calculate the shared secret from the available public information. An overview of ECDH process is defined below.

D.Elliptic Curve Diffie Hellman Algorithm

- Decide domain parameters.

Pa=na*P

Pb=nb*P

The end Alice computes KA=na*pb

- The end Bob computes KB= nb*pa
- Since na*pb=na*nb*P=nb*na*P =nb*pa .therefore KA=KB

Hence the shared secret key is K.

VII: Comparison table between RSA and ECC

	RSA and Diffie Hellman key size in bits	Elliptic curve key size in bits
	1024	160
	2048	224
	3072	256
	7680	384
	15360	512

VIII. Applications

An important factor for this emerging trend is the incorporation of ECDSA in several government and major research institution security standards, including IEEE P1363, ANSI X9.62, ISO 11770-3 and ANSI X9.63. Another factor is the strong promotion of the use of ECC through a Canadian-based Certicom Corporation. Certicom is a company that specializes in information security solutions in a mobile computing environment through providing software and services to its clients. Its success prompted many other companies to look more closely at the benefits and security of ECC. Now, ECC is becoming the mainstream cryptographic scheme in all mobile and wireless devices .

Smart cards are one of the most popular devices for the use of ECC. Many manufacturing companies are producing smart cards that make use of elliptic curve digital signature algorithms. PDAs are considered to be a very popular choice for implementing public key cryptosystems because they have more computing power compared to most of the other mobile devices, like cell phones or pagers. However, they still suffer from limited bandwidth and this makes them an ideal choice for using ECC.

IX. Conclusion and Future Scope

Elliptic Curve Cryptography is such an excellent choice for doing asymmetric cryptography in portable, necessarily constrained devices right now, mainly because of the level of security offered for smaller key sizes. A popular, recommended RSA key size for most applications is 2,048 bits. For equivalent security using Elliptic Curve Cryptography, you need a key size of 224 bits. The difference becomes more and more pronounced as security levels increase (and, as a corollary, as hardware gets faster, and the recommended key sizes must be increased). A 384 -bit Elliptic Curve Cryptography key matches a 7680-bit RSA key for security.

The smaller Elliptic Curve Cryptography keys mean the cryptographic operations that must be performed by the communicating devices can be squeezed into considerably smaller hardware, that software applications may complete cryptographic operations with fewer processor cycles, and operations can be performed that much faster, while still guaranteeing equivalent security. This means, in turn, less heat, less power consumption, less real estate consumed on the printed circuit board, and software applications that run more rapidly and make lower memory demands. Leading in turn to more portable devices which run longer, and produce less heat.

In short, if you're trying to make your devices smaller—and if you need to do asymmetric cryptography, you need Elliptic Curve Cryptography. If you're trying to make them run longer on the same battery, and produce less heat, and you need asymmetric cryptography, you need Elliptic Curve Cryptography. And if you want an asymmetric cryptosystem that scales for the future, you want Elliptic Curve Cryptography. If you just want the most elegant, most efficient asymmetric cryptosystem going, you want Elliptic Curve Cryptography. If you just want the most elegant, most efficient asymmetric cryptosystem going, you want Elliptic Curve Cryptography.

A. Future Scope

In the future the Digital Signature Algorithm will be implemented using elliptic curves. The ECDSA Algorithm will be implemented using Matlab over a Prime field.

B. References

[1] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, *Guide to Elliptic curve Cryptography*, 1996.

[2] Certicom, Standards for Efficient Cryptography, *SEC 1: Elliptic Curve*

[3] George Barwood. Elliptic curve cryptography faq v1.12. 1997.Bruce Schneier. Applied cryptography (2nd ed.): protocols, algorithms, and source code in C. John Wiley & Sons, Inc., New York, NY, USA, 1995

[4] N. Koblitz. CM-curves with good cryptographic properties. In *Advances in Cryptology: Crypto '91*,volume 576 of *Lecture Notes in Computer Science*, pages 279–287, Springer-Verlag, 1992.

[5] "RSA." *Wikipedia*. Wikipedia, n.d. Web. 09 Feb 2011. Stallings, William. *Cryptography and Network Security*. Fourth. Pearson, 2009. Print.