DEVELPOMENT OF SECURE AND EFFICIENT HYBRID CRYPTOSYSTEM

A Project Report

Submitted in partial fulfillment of the requirement for the award of the degree of

MASTER OF ENGINEERING
(COMPUTER TECHNOLOGY & APPLICATIONS)

By

NITIN GOEL
College Roll No. 04/CTA/04
Delhi University Roll No: 8509

Under the guidance of
Mr. Rajeev Kumar.

Department Of Computer Engineering
Delhi College of Engineering, New Delhi-110042
(University of Delhi)

CERTIFICATE

This is certified that this project work entitled **"DEVELOPMENT OF SECURE AND EFFICIENT HYBRID CRYPTOSYSTEM"** is the bonafide work carried out by Mr. Nitin Goel (04/CTA/04) who carried out the work under my supervision, and submitted in partial fulfillment of the requirements for the award of the degree, Master of Engineering (Computer Technology & Applications), during the year 2004 - 2006.

Mr. Rajeev Kumar
Lecturer (Dept. of CSE)
Department of Computer Engineering
Delhi College of Engineering
Bawana Road, Delhi

ABSTRACT

Cryptography is the science of using mathematics to encrypt and decrypt data. Cryptography enables you to store sensitive information or transmit it across insecure networks (like the Internet) so that it cannot be read by anyone except the intended recipient.

Theoretically speaking, no crypto system is completely unbreakable. However, as the complexity of the crypto algorithm increases it becomes practically impossible to the crypto analyst to break. Aim is to make secure cryptosystem .Traditional private key algorithms like DES increases the length of the cipher key to resist the brute force attacks. Though increasing the key length is a simple solution. As the length of the key increases the time to encrypt and Decrypt the input will drastically increase. More over, there is a problem in sending these lengthy keys; to avoid this two crypto system have been implemented:

1) Symmetric key Cryptosystem
2) Public key cryptosystem.

Public Key cryptosystem has been made using elliptic curve as they provide more security than other techniques. And two symmetric key cryptosystem has been implemented one is using elliptic curve and other is AES.

ACKNOWLEDGEMENTS

I wish to take this opportunity to express our deep gratitude to all the people who have extended their cooperation in various ways during our project work. It is our pleasure to acknowledge the help of all these individuals.

I feel it our responsibility to thank Mr. Rajeev Kumar Lecturer (CSE), under whose valuable guidance that the project came out successfully after each stage.

I whole-heartedly thank Dr. Goldie Gabrani, Head of the Department for her constant encouragement and support. I consider ourselves extremely fortunate for having been associated with her.

I am equally grateful and indebted to our beloved Principal Dr.P.B.Sharma for providing us all the pre-requisite facilities.

I also take this opportunity to express our deep sense of gratitude and sincere thanks to the Staff of our Department, for their unstinted co-operation.

Nitin Goel
M.E. (Computer Technology & Applications)
College Roll No. 04/CTA/04
Delhi University Roll No. 8509

Table of Contents

LIST OF FIGURES

LIST OF TABLES

Abbreviations

AES	**Advanced Encryption Standard**
DEA	**Data Encryption Algorithm**
DES	**Data Encryption Standard**
ECC	**Elliptic Curve Cryptosystem**
ECDLP	**Elliptic Curve Discrete Logarithm Problem**
ECDSA	**Elliptic Curve Digital Signature Algorithm**
FIPS	**Federal Information Processing Standard**
GCD	**Greatest Common Divisor**
iKP	**Internet Keyed Payments Protocol**
MAC	**a message authentication code.**
NIST	**National Institute Of Standards And Technology.**
RSA	**Rivest-Shamir-Adleman public key encryption scheme**
WAP	**Wireless Application Protocol**

Chapter-1

INTRODUCTION

1.1 A Brief Introduction to Cryptography:

Cryptography is the science of using mathematics to encrypt and decrypt data. Cryptography enables you to store sensitive information or transmit it across insecure networks (like the Internet) so that it cannot be read by anyone except the intended recipient.

Cryptography might be summed up as the study of techniques and applications that depend on the existence of difficult problems. *Cryptology* (from the Greek *kryptós lógos*, meaning ``hidden word'') is the discipline of cryptography and cryptanalysis combined. To most people, cryptography is concerned with keeping communications private. Indeed, the protection of sensitive communications has been the emphasis of cryptography throughout much of its history. However, this is only one part of today's cryptography.

As we move into an information society, the technological means for global surveillance of millions of individual people are becoming available to major governments. Cryptography has become one of the main tools for privacy, trust, access control, electronic payments, corporate security, and countless other fields. Cryptography is no longer a military thing that should not be messed with.

Encryption is the transformation of data into a form that is as close to impossible as possible to read without the appropriate knowledge (a key). Its purpose is to ensure privacy by keeping information hidden from anyone for whom it is not intended, even those who have access to the encrypted data.

Decryption is the reverse of encryption; it is the transformation of encrypted data back into an intelligible form.

Encryption and decryption generally require the use of some secret information, referred to as a *key*. For some encryption mechanisms, the same key is used for both encryption and decryption; for other mechanisms, the keys used for encryption and decryptions are different.

1.2 Importance of cryptography in Modern World:

Cryptography allows people to carry over the confidence found in the physical world to the electronic world, thus allowing people to do business electronically without worries of deceit and deception. Every day hundreds of thousands of people interact electronically, whether it is through *e*-mail, *e*-commerce (business conducted over the Internet), ATM machines, or cellular phones. The perpetual increase of information transmitted electronically has lead to an increased reliance on cryptography.

Cryptography on the Internet: The Internet, comprised of millions of interconnected computers, allows nearly instantaneous communication and transfer of information, around the world. People use *e*-mail to correspond with one another. The World Wide Web is used for online business, data distribution, marketing, research, learning, and a myriad of other activities.

Cryptography makes secure web sites and electronic safe transmissions possible. For a web site to be secure all of the data transmitted between the computers where the data is kept and where it is received must be encrypted. This allows people to do online banking, online trading, and make online purchases with their credit cards, without worrying that any of their account information is being compromised. Cryptography is very important to the continued growth of the Internet and electronic commerce.

E-mail: It is transmitted in plain text over unknown pathways and resides for various periods of time on computer files over which you have no control. Whether you're planning a political campaign, discussing your finances, having an affair, completing a business deal, or engaging in some totally innocuous activity, your messages have less privacy than if you sent all of your written correspondence on postcards.

The nature of the Internet and the electronic medium allows effective scanning of message contents using sophisticated filtering software. Electronic

mail is gradually replacing conventional paper mail and messages can be easily and automatically intercepted and scanned for interesting keywords.

Another problem with e-mail is that it is very easy to forge the identity of the sender. The solution to these problems is to use cryptography. However, there are restrictions on the export and use of strong cryptography, particularly in the USA, but now gaining momentum in other countries. Furthermore, some governments, and again the USA is the most prominent, want decryption keys lodged with escrow agents, so that law enforcement agencies can, with appropriate authorization, intercept and decrypt private messages. It is often claimed that this facility is no different from powers that the government has always possessed to wiretap telephones. There is however, a vital difference. Citizens are now being asked to take action to make them available for surveillance.

The major applications for encryption may then be summarized as:

To protect privacy and confidentiality.

To transmit secure information (e.g. credit card details)

To provide authentication of the sender of a message.

To provide authentication of the time a message was sent.

***E*-commerce:** it is increasing at a very rapid rate. By the turn of the century, commercial transactions on the Internet are expected to total hundreds of billions of dollars a year. This level of activity could not be supported without cryptographic security. It has been said that one is safer using a credit card over the Internet than within a store or restaurant. It requires much more work to seize credit card numbers over computer networks than it does to simply walk by a table in a restaurant and lay hold of a credit card receipt. These levels of security, though not yet widely used, give the means to strengthen the foundation with which *e*-commerce can grow.

People use *e*-mail to conduct personal and business matters on a daily basis. *E*-mail has no physical form and may exist electronically in more than one

place at a time. This poses a potential problem as it increases the opportunity for an eavesdropper to get a hold of the transmission. Encryption protects *e*-mail by

rendering it very difficult to read by any unintended party. Digital signatures can also be used to authenticate the origin and the content of an *e*-mail message.

Access Control: Cryptography is also used to regulate access to satellite and cable TV. Cable TV is set up so people can watch only the channels they pay for. Since there is a direct line from the Cable Company to each individual subscriber's home, the Cable Company will only send those channels that are paid for. Many companies offer pay-per-view channels to their subscribers. Pay-per-view cable allows cable subscribers to ``rent'' a movie directly through the cable box. What the cable box does is decode the incoming movie, but not until the movie has been ``rented.'' If a person wants to watch a pay-per-view movie, he/she calls the Cable Company and requests it. In return, the Cable Company sends out a signal to the subscriber's cable box, which unscrambles (decrypts) the requested movie.

Satellite TV works slightly differently since the satellite TV companies do not have a direct connection to each individual subscriber's home. This means that anyone with a satellite dish can pick up the signals. To alleviate the problem of people getting free TV, they use cryptography. The trick is to allow only those who have paid for their service to unscramble the transmission; this is done with receivers (``unscrambles''). Each subscriber is given a receiver; the satellite transmits signals that can only be unscrambled by such a receiver (ideally). Pay-per-view works in essentially the same way as it does for regular cable TV.

As seen, cryptography is widely used. Not only is it used over the Internet, but also it is used in phones, televisions, and a variety of other common household items. Without cryptography, hackers could get into our *e*-mail, listen in on our phone conversations, tap into our cable companies and acquire free cable service, or break into our bank/brokerage accounts.

Chapter-2

BASIC TECHNIQUES AND ALGORITHMS

A *cryptographic algorithm*, or *cipher*, is a mathematical function used in the Encryption and decryption process. A cryptographic algorithm works in Combination with a *key,* a word, numbers or phrase — to encrypt the plain text.

The same plain text encrypts to different cipher text with different keys. The security of encrypted data is entirely dependent on two things: the strength of the cryptographic algorithm and the secrecy of the key. A cryptographic algorithm, plus all possible keys and all the protocols that make it work comprise a *cryptosystem*.

The method of encryption and decryption is called a **cipher**. Some cryptographic methods rely on the secrecy of the algorithms; such algorithms are only of historical interest and are not adequate for real-world needs. All modern algorithms use a **key** to control encryption and decryption; a message can be decrypted only if the key matches the encryption key.

2.1 Symmetric Key vs. Asymmetric Key Ciphers:

There are two classes of key-based encryption algorithms, **symmetric** (or **secret-key**) and **asymmetric** (or **public-key**) algorithms. The difference is that symmetric algorithms use the same key for encryption and decryption (or the decryption key is easily derived from the encryption key), whereas asymmetric algorithms use a different key for encryption and decryption, and the decryption key cannot be derived from the encryption key.

Symmetric algorithms can be divided into **stream ciphers** and **block ciphers**. Stream ciphers can encrypt a single bit of plain text at a time, whereas block ciphers take a number of bits (typically 64 bits in modern ciphers), and encrypt them as a single unit.

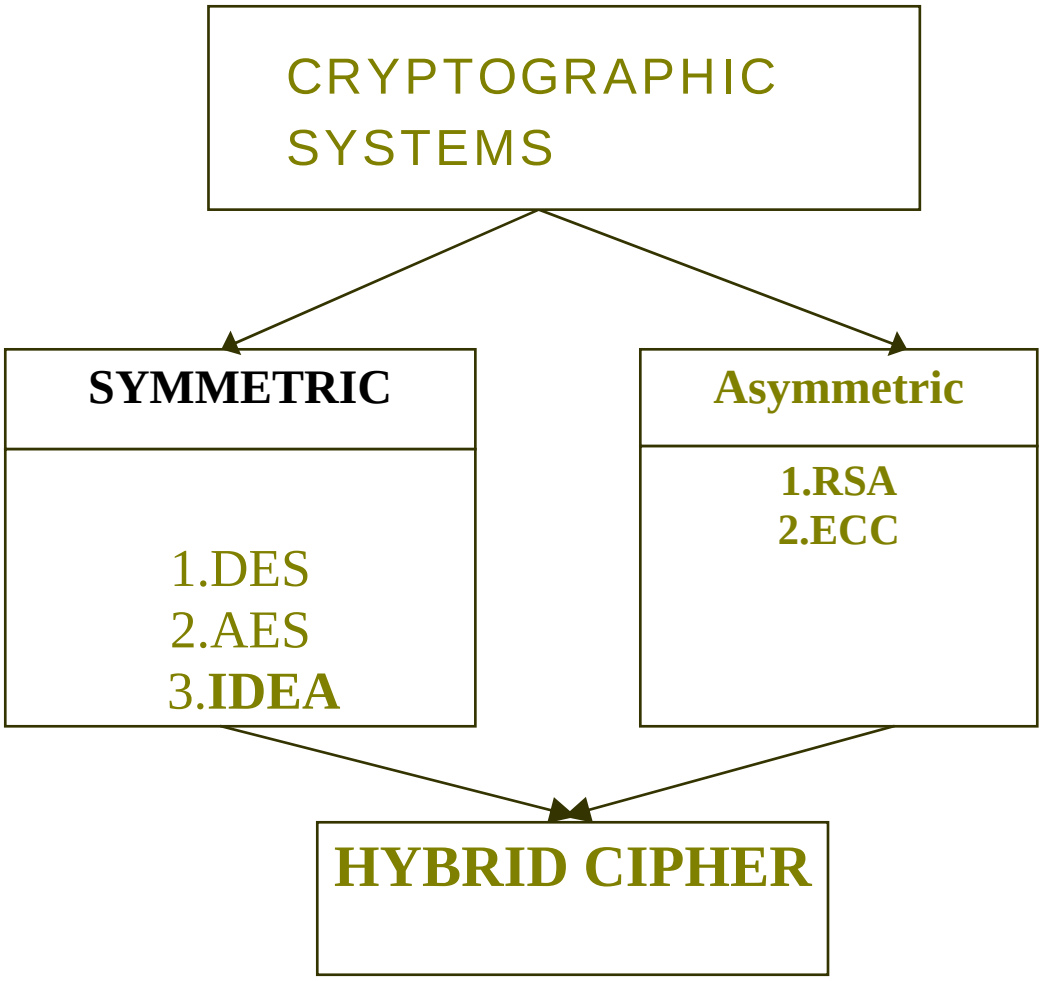


Fig 2.1 Symmetric Vs Asymmetric

Asymmetric ciphers (also called **public-key algorithms** or generally **public-key cryptography**) permit the encryption key to be public (it can even be published in a newspaper), allowing anyone to encrypt with the key, whereas only the proper recipient (who knows the decryption key) can decrypt the message. The encryption key is also called the **public key** and the decryption keys the **private key** or **secret key**.

Modern cryptographic algorithms are no longer pencil-and-paper ciphers. Strong cryptographic algorithms are designed to be executed by computers or specialized hardware devices. In most applications, cryptography is done in computer software.

Generally, symmetric algorithms are much faster to execute on a computer than asymmetric ones. In practice they are often used together, so that a public-key algorithm is used to encrypt a randomly generated encryption key, and the random key is used to encrypt the actual message using a symmetric algorithm. This is sometimes called hybrid encryption.

The most studied and probably the most widely spread symmetric cipher is DES; the upcoming AES might replace it as the most widely used encryption algorithm. RSA is probably the best-known asymmetric encryption algorithm.

2.2 Strength of Cryptographic Algorithms:

Good cryptographic systems should always be designed so that they are as difficult to break as possible. It is possible to build systems that cannot be broken in practice (though this cannot usually be proved). This does not significantly increase system implementation effort; however, some care and expertise is required. There is no excuse for a system designer to leave the system breakable. Any mechanisms that can be used to circumvent security must be made explicit, documented, and brought into the attention of the end users.

In theory, any cryptographic method with a key can be broken by trying all possible keys in sequence. If using **brute force** to try all keys is the only option, the required computing power increases exponentially with the length of the key. A *32*-bit key takes 2^{32} (about 10^{9}) steps. This is something anyone can do on his/her home computer. A system with *40* bit keys takes 2^{40} steps – this kind of computation requires something like a week (depending on the efficiency of the algorithm) on a modern home computer. A system with *56* bit keys (such as DES) takes a substantial effort (with a large number of home computers using distributed effort, it has been shown to take just a few months), but is easily breakable with special hardware. The cost of the special hardware is substantial but easily within reach of organized criminals, major companies, and

lesser governments in few years. Keys with *80* bits appear good for a few years, and keys with *128* bits will probably remain unbreakable by brute force for the foreseeable future. Even larger keys are sometimes used.

However, key length is not the only relevant issue. Many ciphers can be broken without trying all possible keys. In general, it is very difficult to design ciphers that could not be broken more effectively using other methods. Designing your own ciphers may be fun, but it is not recommended for real applications unless you are a true expert and know exactly what you are doing.

To give some idea of the complexity for the RSA cryptosystem, a *256*-bit modulus is easily factored at home, and *university research groups can break 512 bit keys* within a few months. Keys with *768* bits are probably not secure in the long term. Keys with *1024* bits and more should be safe for now unless major cryptographical advances are made against RSA; keys of *2048* bits are considered by many to be secure for decades.

It should be emphasized that the strength of a cryptographic system is usually equal to its weakest link. No aspect of the system design should be overlooked, from the choice algorithms to the key distribution and usage policies.

ALGORITHM	KEY SIZE	No. OF ROUNDS	MATHEMATICAL OPERATIONS
DES	56 BITS	16	XOR,S-boxes
TRIPLE DES	112 to 168	48	XOR,S-boxes
IDEA	128	8	XOR,Addition,Multiplication
CAST-128	40 TO 128	16	XOR, Sboxes, Rotation, Subtraction, Addition.
TWO FISH	128 TO 256	16	XOR, Sboxes, Rotation.

Table 1: Comparison of Crypt algorithms

2.3 Cryptanalysis and Attacks on Cryptosystems:

Cryptanalysis is the art of deciphering encrypted communications without knowing the proper keys. There are many cryptanalytic techniques. Some of the more important ones for a system implementer are described below.

Cipher text-only attack: This is the situation where the attacker does not know anything about the contents of the message, and must work from cipher text only. In practice it is quite often possible to make guesses about the plaintext, as many types of messages have fixed format headers.

Modern cryptosystems are not weak against only-only attacks, although sometimes they are considered with the added assumption that the message contains some statistical bias.

Known-plaintext attack: The attacker knows or can guess the plaintext for some parts of the only. The task is to decrypt the rest of the only blocks using this information. This may be done by determining the key used to encrypt the data, or via some shortcut.

One of the best-known modern known-plaintext attacks is linear cryptanalysis against block ciphers.

Chosen-plaintext attack: The attacker is able to have any text he likes encrypted with the unknown key. The task is to determine the key used for encryption.

Chapter-3

DATA ENCRYPTION AND ITS APPLICATIONS

Applications of Cryptography include the most important protocols and systems made possible by cryptography. In particular they discuss the issues involved in establishing a cryptographic infrastructure, and it gives a brief overview of some of the electronic commerce techniques available today. It is :

Electronic Commerce

3.1 Electronic Commerce:

Cryptography is extremely useful to electronic commerce in the areas of payment systems, and transactions over open networks. While several protocols and payment systems exist, the most widely used protocol for Internet transactions is SSL.

3.1.1 Electronic money:

Electronic money (also called electronic cash or digital cash) is a term that is still fairly vague and undefined. It refers to transactions carried out electronically with a net result of funds transferred from one party to another. Electronic money may be either debit or credit. Digital cash per se is basically another currency, and digital cash transactions can be visualized as a foreign exchange market. This is because we need to convert an amount of money to digital cash before we can spend it. The conversion process is analogous to purchasing foreign currency.

Digital cash in its precise definition may be anonymous or identified. Anonymous schemes do not reveal the identity of the customer and are based on blind signature schemes. Identified spending schemes always reveal the identity of the customer and are based on more general forms of signature schemes. Anonymous schemes are the electronic analog of cash, while identified schemes are the electronic analog of a debit or credit card. There are other approaches; payments can be anonymous with respect to the merchant but not the bank, or anonymous to everyone, but traceable (a sequence of purchases can be related, but not linked directly to the spender’s identity).

Since digital cash is merely an electronic representation of funds, it is possible to easily duplicate and spend a certain amount of money more than once. Therefore, digital cash schemes have been structured so that it is not

possible to spend the same money more than once without getting caught immediately or within a short period of time. Another approach is to have the digital cash stored in a secure device, which prevents the user from double spending.

Electronic money also encompasses payment systems that are analogous to traditional credit cards and checks. Here, cryptography protects conventional transaction data such as an account number and amount; a digital signature can replace a handwritten signature or a credit-card authorization, and public-key encryption can provide confidentiality.

There are a variety of systems for this type of electronic money, ranging from those that are strict analogs of conventional paper transactions with a typical value of several dollars or more, to those (not digital cash per se) that offer a form of ``micro payments'' where the transaction value may be a few pennies or less. The main difference is that for extremely low-value transactions even the limited overhead of public-key encryption and digital signatures is too much, not to mention the cost of ``clearing'' the transaction with bank. As a result, ``batching'' of transactions is required, with the public key operations done only occasionally.

3.1.2 The iKP:

The Internet Keyed Payments Protocol (iKP) is architecture for secure payments involving three or more parties. Developed at IBM's T.J. Watson Research Center and Zurich Research Laboratory, the protocol defines transactions of a ``credit card'' nature, where a buyer and seller interact with a third party ``acquirer,'' such as a credit-card system or a bank, to authorize transactions. The protocol is based on public-key cryptography.

IKP is no longer widely in use; however it is the current foundation for SET.

3.1.3 SET:

Visa and MasterCard have jointly developed the Secure Electronic Transaction (SET) protocol as a method for secure, cost effective bankcard transactions over open networks. SET includes protocols for purchasing goods and services electronically, requesting authorization of payment, and requesting ``credentials'' (that is, certificates) binding public keys to identities, among other services. Once SET is fully adopted, the necessary confidence in secure electronic transactions will be in place, allowing merchants and customers to partake in electronic commerce.

SET supports DES for bulk data encryption and RSA for signatures and public-key encryption of data encryption keys and bankcard numbers. The RSA public-key encryption employs Optimal Asymmetric Encryption Padding. SET is being published as open specifications for the industry, which may be used by software vendors to develop applications.

Chapter-4

EXISTING SYSTEMS FOR DATA ENCRYPTION

4.1 The RSA Algorithm:

The RSA cryptosystem is a public-key cryptosystem that offers both encryption and digital signatures (authentication). Ronald Rivest, Adi Shamir, and Leonard Adleman developed the RSA system in 1977 [RSA78]; RSA stands for the first letter in each of its inventors' last names.

The RSA algorithm works as follows: take two large primes, p and q, and compute their product $n = pq$; n is called the modulus. Choose a number, e, less than n and relatively prime to (p-1) (q-1), which means e and (p-1)(q-1) have no common factors except 1. Find another number d such that (ed – 1) is divisible by (p-1)(q-1). The values e and d are called the public and private exponents, respectively. The public key is the pair (n, e); the private key is (n, d). The factors p and q may be destroyed or kept with the private key.

It is currently difficult to obtain the private key d from the public key (n, e). However if one could factor n into p and q, then one could obtain the private key d. Thus the security of the RSA system is based on the assumption that factoring is difficult. The discovery of an easy method of factoring would ``break'' RSA.

Here is how the RSA system can be used for encryption and digital signatures (in practice, the actual use is slightly different):

Encryption: Suppose Alice wants to send a message m to Bob. Alice creates the cipher text c by exponentiating: $c = m^e \bmod n$, where e and n are Bob's public key. She sends c to Bob. To decrypt, Bob also exponentiates: $m = c^d \bmod n$; the relationship between e and d ensures that Bob correctly recovers m. Since only Bob knows d, only Bob can decrypt this message.

In practice, the RSA system is often used together with a secret-key crypto system, such as DES, to encrypt a message by means of an RSA digital envelope. Suppose Alice wishes to send an encrypted message to Bob. She first encrypts the message with DES, using a randomly chosen DES key. Then she looks up Bob's public key and uses it to encrypt the DES key. The DES-encrypted message and the RSA-encrypted DES key together form the RSA digital envelope and are sent to Bob. Upon receiving the digital envelope, Bob decrypts the DES key with his private key, and then uses the DES key to decrypt

the message itself. This combines the high speed of DES with the key management convenience of the RSA system.

An ``RSA operation,'' whether encrypting, decrypting, signing, or verifying is essentially a modular exponentiation. This computation is performed by a series of modular multiplications.

In practical applications, it is common to choose a small public exponent for the public key. In fact, entire groups of users can use the same public exponent, each with a different modulus. (There are some restrictions on the prime factors of the modulus when the public exponent is fixed.) This makes encryption faster than decryption and verification faster than signing.

The best size for a modulus depends on one's security needs. The larger the modulus, the greater the security, but also the slower the RSA algorithm operations. One should choose a modulus length upon consideration, first, of the value of the protected data and how long it needs to be protected, and, second, of how powerful one's potential threats might be.

A good analysis of the security obtained by a given modulus length is given by Rivest, in the context of discrete logarithms modulo a prime, but it applies to the RSA algorithm as well.

As for the slowdown caused by increasing the key size, doubling the modulus length will, on average, increase the time required for public key operations (encryption and signature verification) by a factor of four, and increase the time taken by private key operations (decryption and signing) by a factor of eight. The reason public key operations are affected less than private key operations is that the public exponent can remain fixed while the modulus is increased, whereas the length of the private exponent increases proportionally. Key generation time would increase by a factor of 16 upon doubling the modulus, but this is a relatively infrequent operation for most users.

4.2 The DES Algorithm:

DES, an acronym for the Data Encryption Standard, is the name of the Federal Information Processing Standard (FIPS) 46-3, which describes the data encryption algorithm (DEA).

The DEA has a 64-bit block size and uses a 56-bit key during execution (8 parity bits are stripped off from the full 64-bit key). The DEA is a symmetric crypto system, specifically a 16-round Feistel cipher and was originally designed for implementation in hardware. When used for communication, both sender and receiver must know the same secret key, which can be used to encrypt and decrypt the message, or to generate and verify a message authentication code (MAC). The DEA can also be used for single-user encryption, such as to store files on a hard disk in encrypted form. In a multi-user environment, secure key distribution may be difficult; public-key cryptography provides an ideal solution to this problem.

It should be noted that the key sizes for the RSA system (and other public-key techniques) are much larger than those for block ciphers like DES, but the security of an RSA key cannot be compared to the security of a key in another system purely in terms of length.

Chapter-5

EXISTING SYSTEMS – PROS AND CONS

Theoretically speaking, no crypto system is completely unbreakable. However, as the complexity of the crypto algorithm increases it becomes practically impossible to the crypto analyst to break it using even the most modern and powerful computing hardware. As the power of the hardware increases with time, and more computing power becomes available to the crypto analyst, the robustness of the crypto cipher should also be increased.

Traditional private key algorithms like DES increases the length of the cipher key to resist the brute force attacks. For example, some 10 years ago, when a typical high-end computer system used to run at the speed in the order of 100 MHz or so, a DES cipher of 64 Bit was pretty secure. But using to day's high-end systems, which operate typically around 1.5 GHz speed, the old 64 Bit DES cipher is no longer secure. We need a 128 bit or greater cipher to withstand the brute force attacks using the modern high-end systems.

Though increasing the key length is a simple solution to make the algorithm practically impossible to break, it has its own price to pay for. As the length of the key increases the time to encrypt and Decrypt the input will drastically increase. More over, additional secure methods are needed to store and transport such lengthy keys. All these side effects effectively reduce the overall security the cipher offers and also hampers the performance of the crypto system.

Public key crypto systems like the RSA can solve the some of the traditional problems associated with Private key cryptographic systems like DES. Irrespective of the length of the key, these systems offer a reasonably good degree of security. However, these algorithms are formidably complex and tediously slow when compared to symmetric key algorithms. Thus they are not very suitable for developing Block Cipher based crypto systems.

Particularly for developing a static data encryption system like the one that handles databases, images, audio and video files, binary data using a public key based algorithm is impractical as they are very slow. Symmetric key crypto systems like the DES with some degree of sophistication are very well suited for such applications.

Many such applications use a hybrid of symmetric and asymmetric ciphers. For example, a system can use the RSA techniques to generate the key, which in turn will be used by a DES cipher to encrypt the data. Though this solution seems to be a sound one to avoid the many performance bottlenecks associated with the traditional Asymmetric key crypto systems, it can't be an effective and universal alternative for all cases. There are many tradeoffs in this approach.

To over come these tradeoffs, crypto experts devised several other techniques. The proposed hybrid cipher is on such system that aims at using the basic DES routines for core encryption. However, the key that the system uses will be generated using a complex set of discrete mathematical functions.

Chapter-6

WHY ELLIPTIC CURVE PUBLIC KEY CRYPTOSYSTEM

Elliptic curve for making public key cryptosystem has been used because of following reasons [7]:

1). For the same level of security per best current known attacks, elliptic curve –based systems can be implemented with much smaller parameters leading to significant performance advantages. Such performance improvements are particularly important in the wireless arena where computing power, memory, and battery life of devices are more constrained.

2).Key sizes for equivalent security levels: Table 2 shows key comparison of AES (rinjdael), ECC and (DH/DSA/RSA) for the same security levels:

Symmetric	ECC	DH/DSA/RSA
80	163	1024
128	283	3072
192	409	7680
256	571	15,360

Table 2.Key sizes for equivalent security levels (in bits)

Its clear that ECC provide security more or equivalent to symmetric AES.Table 1 can be found in a number of standard documents (see references)

3). In Table 3 rows 1 and 2 do not claim to be optimized, but show two different platforms and are directly comparable to RSA numbers for the same platforms. Rows 3 and 4 in Table 3 take advantage of the special form of the generalized Mersenne primes for the NIST curves.

	Processor	MHz	163-bit	192-bit	256-bit	384-bit	521-bit
1)	Ultra SPARC II	450	6.1	8.7	–	–	–
2)	StrongARM	200	22.9	37.7	–	–	–
3)	Pentium II	400	–	18.3	42.4	136.4	310.4
4)	Pentium II	400	–	2.1	5.1	16.4	27.8

Table3. Sample elliptic curve exponentiation timings over prime fields (in ms)

	Processor	MHz	1024-RSA_d	1024-RSA_e	2048-RSA_d	2048-RSA_e
1)	Ultra SPARC II	450	32.1	1.7	205.5	6.1
2)	StrongARM	200	188.7	10.8	1273.8	39.1
3)	ARM7TDMI	1	12,070	1180	–	–

Table4. Sample RSA encrypt/decrypt timings (in milliseconds)

In Table 4 RSA_d is the private key operation, whereas RSA_e is the public key operation.

At the 163-bit ECC/1024-bit RSA security level, an elliptic curve exponentiation for general curves over arbitrary prime fields is roughly 5 to 15 times as fast as an RSA private key operation, depending on the platform and optimizations. At the 256-bit ECC/3072-bit RSA security level the ratio has already increased to between 20 and 60, depending on optimizations. To secure a 256-bit AES key, ECC-521 can be expected to be on average 400 times faster than 15,360-bit RSA.

Chapter-7

INTRODUCTION TO ELLIPTIC CURVE AND ITS CRYPTOGRAPHY

7.1 Introduction and History

The mathematical idea fundamental to public-key cryptography is that of a hard problem and from such problems, mechanisms for public-key key exchange might be constructed. If an additional technical requirement (a trapdoor) can be designed then the hard problem might possibly be used to construct a public-key encryption or a digital signature algorithm.

While the 20-year history of public-key cryptography has seen a diverse range of proposals for candidate hard problems only two have truly stood the test of time. These problems are known as the discrete logarithm problem over a finite field and integer factorization.

During the mid-1980's various researchers observed that another source for hard problems might be discovered by looking at elliptic curves. Elliptic curves are rich mathematical structures which have shown themselves to be remarkably useful in a range of applications including primarily testing and integer factorization. One potential use of elliptic curves is in the definition of public-key cryptosystems that are close analogs of existing schemes. In this way, variants of existing schemes can be devised that rely for their security on a different underlying hard problem.

7.2 Elliptic Curve Cryptosystems

Elliptic curve cryptosystems are analogs of existing RSA cryptosystem and it is possible to define analogs of public-key cryptosystems that are based on the discrete logarithm problem (such as ElGamal encryption and the DSA for instance).

It is interesting to note that the problems of integer factorization and of discrete logarithms over a prime field appear to be of roughly the same difficulty. Techniques used to solve one problem can be adapted to tackle the other

The situation is different with variants of discrete logarithm cryptosystems. The security of the elliptic curve variants of discrete logarithm cryptosystems depends on a restatement of the conventional discrete logarithm problem for elliptic curves. This restatement is such that current algorithms that solve the

conventional discrete logarithm problem in what is termed sub-exponential time are of little value in attacking the analogous elliptic curve problem. Instead the only available algorithms for solving these elliptic curve problems are more general techniques that run in what is termed exponential time.

The distinction between exponential and sub-exponential time for solving some problem is a vitally important one. In essence it means that methods of finding a solution to one problem are becoming infeasible much faster than those for solving the other problem.

Elliptic curve cryptosystem are much more secure then its analogous.Thats why now days it is using more as comparison to RSA.[5]

7.3 Introduction of Elliptic Curve Arithmetic

7.3.1 Abelian Groups

An abelian group G, sometimes denoted by {G,*} is a set of elements with a binary operation, denoted by *,that associates to each ordered pair(a,b) of elements (a*b) in G such that the following axioms are obeyed.

(A1) Closure:	If a and b belong to G, then a * b is also in G.
(A2) Associate:	a * (b * c) = (a * b) * c for all a, b, c in G
(A3) Identity element:	There is an element in G such that a*e=e*a=a for all a in G.
(A4) Inverse element:	For each a in G there is an element a' in G such that a * a' = a' * a = e
(A5) Commutative:	a * b = b * a for all a, b in

A number of public-key ciphers are based on the use of an abelian group. For example, Diffie-Hellman key exchange involves multiplying pairs of nonzero integers modulo a prime number q. Keys are generated by exponentiation over the gop, with exponentiation defined as repeated multiplication. For example, a mod q = a X a X ….. X a) mod q. To attack Diffie-Hellman, the attacker must determine k given a; this is the discrete log problem.

For elliptic curve cryptography, an operation over elliptic curves, called addition, is used. Multiplication is defined by repeated addition. For example, a * k = (a + a + …. + a), where the addition is performed over an elliptic curve. Cryptanalysis involves determining k given a and (a * k).

An elliptic curve is defined by an equation in two variables, with coefficients. For cryptography, the variables and coefficients are restricted to elements in a finite field, which results in the definition of a finite abelian group.

7.3.2 Elliptic Curve Groups over Real Numbers

Elliptic curve are not ellipses .They are so named because they are described by cubic equation, similar to those used for calculating the circumference of an ellipse. In general, cubic equation for elliptic curves takes the form

$$y^{2+} axy + by^{=}x3 + cx2 + dx + e$$

Where a, b, c, d and e are real numbers and x and y lake on values in the real numbers.

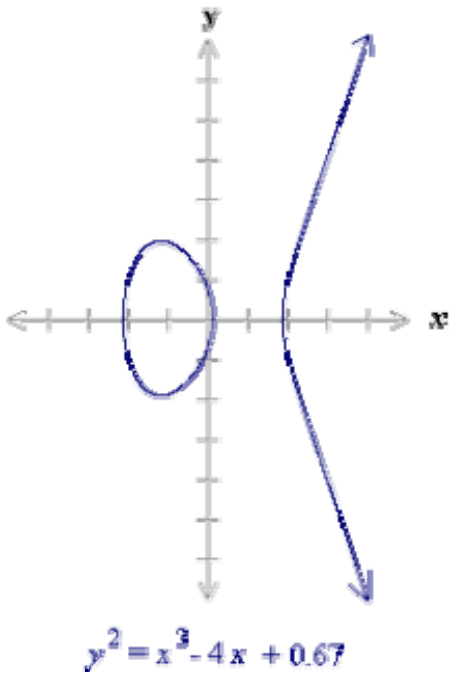
For our purpose, it is sufficient to limit ourselves to equation of the form

$$y^2 = x^3 + ax + b,$$

Where x, y, a and b are real numbers.

Each choice of the numbers a and b yields a different elliptic curve. For example, a = -4 and b = 0.67 gives the elliptic curve with equation $y^2 = x^3 - 4x + 0.67$; the graph of this curve is shown below:

If $x^3 + ax + b$ contains no repeated factors, or equivalently if $4a^3 + 27b^2$ is not 0, then the elliptic curve $y^2 = x^3 + ax + b$ can be used to form a group. An elliptic curve group over real numbers consists of the points on the corresponding elliptic curve, together with a special point O called the point at infinity.



7.3.3 Geometric Description of Addition

It can be shown that a group can be defined based on the set E (a, b) provided that x + ax + b has no repeated factors. This is equivalent to the condition

4a3 + 27b2 ≠ 0

An operation called addition and denote by +, for the set E (a, b), where a and b satisfy above Equation. In geometric terms, the rules for addition can be stated as follows: If three points on an elliptic curve lie on a straight line, their sum is O. From this definition, the rules of addition over an elliptic curve:

1).O serves as the additive identity. Thus O = − O ; for any point P on the elliptic curve, P + O = P. In what follows, we assume P ≠ O and Q ≠ O.

2). The negative of a point P is the point with the same x coordinate but the negative of the y coordinate [i. e, if P = (x, y), then – P = (x, -y)]. Note that these two points can be joined by a vertical line. Note that P + (− P) = P – P = O.

3). To add two points P and Q with different x coordinate, draw a straight line between them and find the third point of intersection R. It is easily seen that there is a unique point R that is the point of intersection (unless the line is tangent to the curve at either P or Q, in which case we take R = P or R + Q,

respectively). To form a group structure, we need to define addition on these three points as follows: P + Q = - R. That is, we define P + Q to be the mirror image (with respect to the x axis) of the third point of intersection.

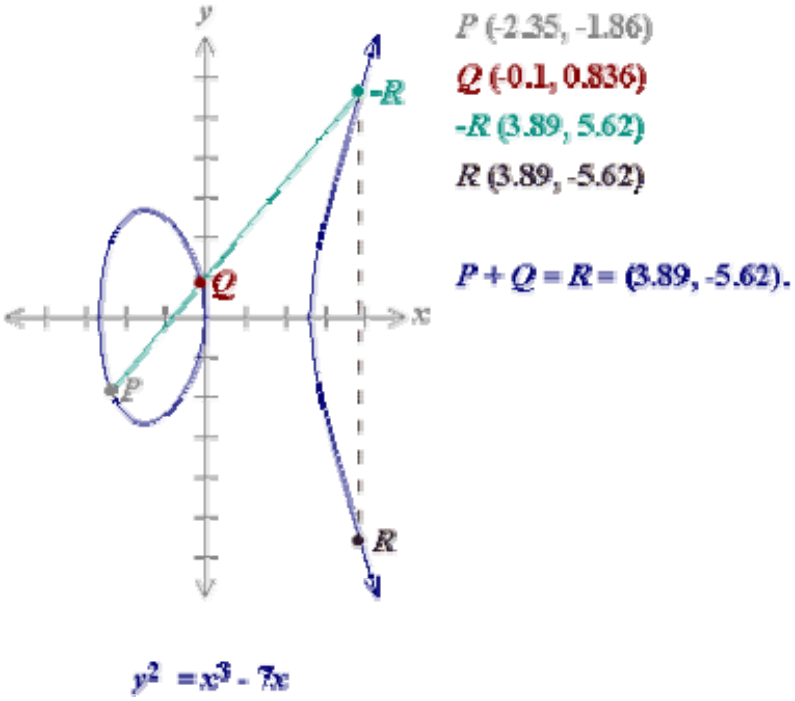
4). The geometric interpretation of the preceding item can also applies to two points P and – P, with the same x coordinate. The points are joined by a vertical line, which can be viewed as also intersecting the curve at the infinity point. We therefore have P + (− P) = O, consistent with item (2).

5). Then Q + Q = 2Q = − S.

7.3.3.1 Examples

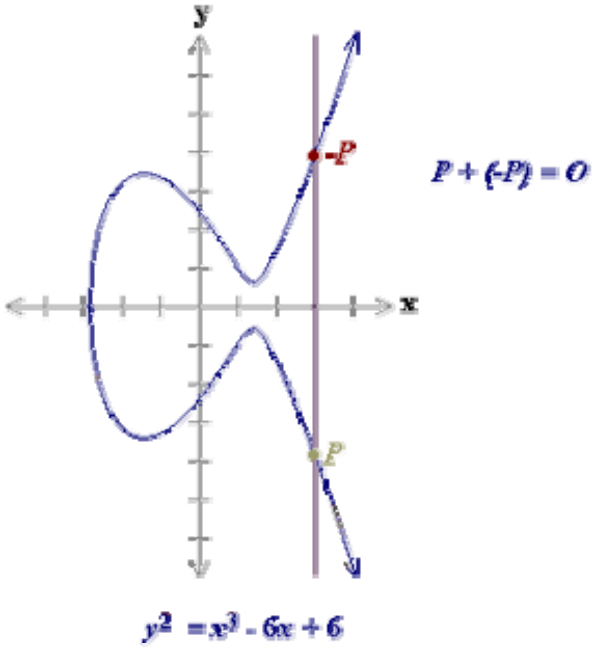
Adding distinct points P and Q

Suppose that P and Q are two distinct points on an elliptic curve, and the P is not -Q. To add the points P and Q, a line is drawn through the two points. This line will intersect the elliptic curve in exactly one more point, call -R. The point -R is reflected in the x-axis to the point R. The law for addition in an elliptic curve group is P + Q = R For example:



7.3.3.2 Adding the points P and -P

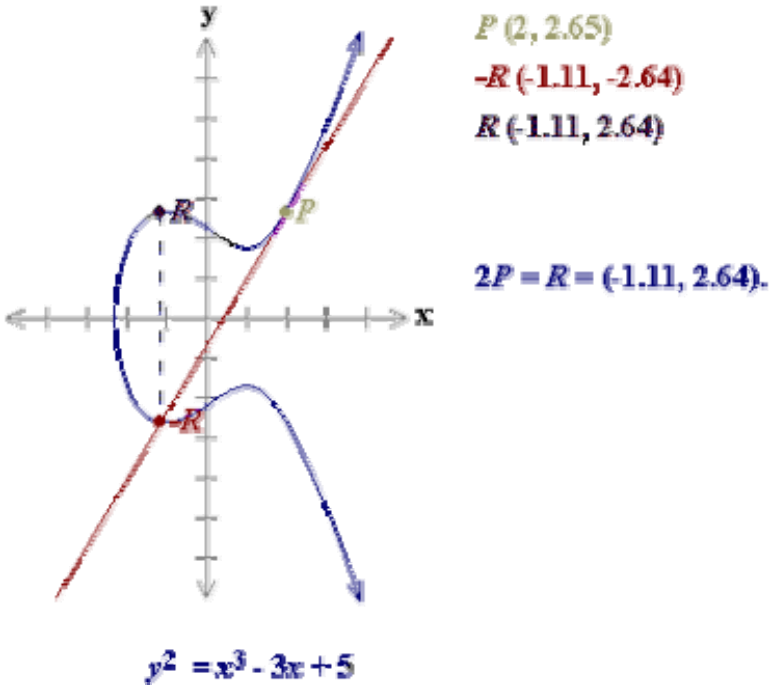
The line through P and -P is a vertical line which does not intersect the elliptic curve at a third point; thus the points P and -P cannot be added as previously. It is for this reason that the elliptic curve group includes the point at infinity O. By definition, P + (-P) = O. As a result of this equation, P + O = P in the elliptic curve group . O is called the additive identity of the elliptic curve group; all elliptic curves have an additive identity.



7.3.3.3 Doubling the point P

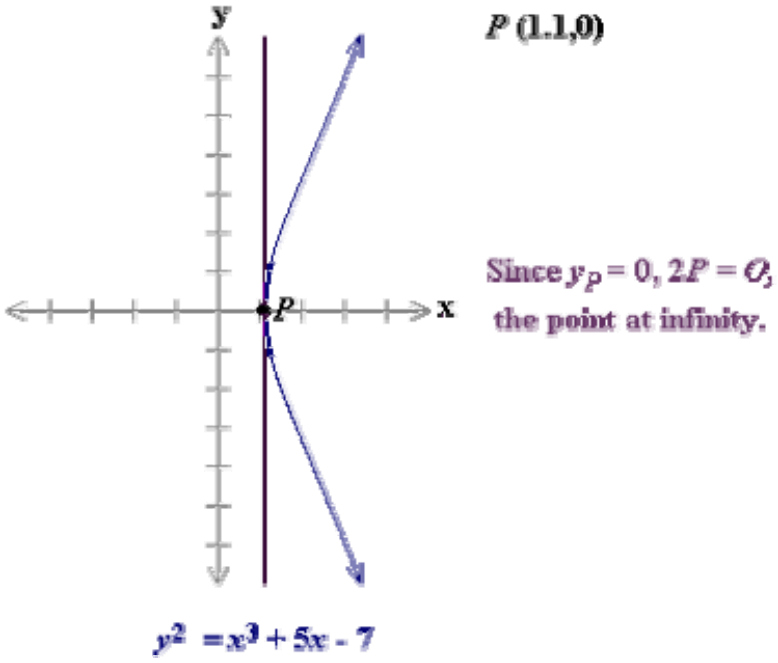
To add a point P to itself, a tangent line to the curve is drawn at the point P. If yP is not 0, then the tangent line intersects the elliptic curve at exactly one other point, -R. -R is reflected in the x-axis to R. This operation is called doubling the point P; the law for doubling a point on an elliptic curve group is defined by:

P+P=2P=R.



Doubling the point P if yP = 0

If a point P is such that yP = 0, then the tangent line to the elliptic curve at P is vertical and does not intersect the elliptic curve at any other point. By definition, 2P = O for such a point P. If one wanted to find 3P in this situation, one can add 2P + P. This becomes P + O = P Thus 3P = P. 3P = P, 4P = O, 5P = P, 6P = O, 7P = P, etc.



7.3.4 Elliptic Curve Addition: An Algebraic Approach

Although the previous geometric descriptions of elliptic curves provide an excellent method of illustrating elliptic curve arithmetic, it is not a practical way to implement arithmetic computations. Algebraic formulae are constructed to efficiently compute the geometric arithmetic. [5]

7.3.4.1 Adding distinct points P and Q

When P = (x_P,y_P) and Q = (x_Q,y_Q) are not negative of each other, P + Q = R where s = $(y_P - y_Q) / (x_P - x_Q)$ $x_R = s^2 - x_P - x_Q$ and $y_R = -y_P + s(x_P - x_R)$ Note that s is the slope of the line through P and Q.

7.3.4.2 Doubling the point P

When y_P is not 0, 2P = R where s = $(3x_P^2 + a) / (2y_P)$ $x_R = s^2 - 2x_P$ and $y_R = -y_P + s(x_P - x_R)$. Recall that a is one of the parameters chosen with the elliptic curve and that s is the tangent on the point P.

Some facts about elliptic arithmetic over real numbers:

1). The elliptic curve equation $y^2 = x^3 - 7x - 6$ over real numbers define a group as

$4a^3 + 27b^2 = 4(-7)^3 + 27(-6)^2 = -400$.The equation $y^2 = x^3 - 7x - 6$ does define an elliptic curve group because $4a^3+27b^2$is not 0.

2). The additive identity of regular integers is 0, since x + 0 = x for all integers.

3). The point (4,7) is lie on the elliptic curve $y^2 = x^3 - 5x + 5$ over real numbers since the equation holds true for x = 4 and y = 7:

$(7)^2 = (4)^3 - 5(4) + 5$ 49 = 64 - 20 + 5 49 = 49

4). The negative is the points on the elliptic curve over real numbers are reflected through the x-axis. Thus P (-4,-6) = -P (-4, 6), Q (17, 0) =-Q (17, 0),
R (3, 9) = -R (3,-9).

5). In the elliptic curve group defined by $y^2 = x^3 - 17x + 16$ over real numbers,

(i) the P + Q if P = (0,-4) and Q = (1,0) is : $s = (y_P - y_Q) / (x_P - x_Q) = (-4 - 0) / (0 - 1) = 4$ $x_R = s^2 - x_P - x_Q = 16 - 0 - 1 = 15$ and $y_R = -y_P + s(x_P - x_R) = 4 + 4(0 - 15) = -56$ Thus P + Q = (15, -56)

(ii) 2P if P = (4, 3.464) **:** $s = (3x_P^2 + a) / (2y_P) = (3*(4)^2 + (-17)) / 2*(3.464) = 31 / 6.928 = 4.475$ $x_R = s^2 - 2x_P = (4.475)^2 - 2(4) = 20.022 - 8 = 12.022$ and $y_R = -y_P + s(x_P - x_R) = -3.464 + 4.475(4 - 12.022) = - 3.464 - 35.898 = -39.362$ Thus 2P = (12.022, -39.362) .

7.3.5 Elliptic Curves over Z

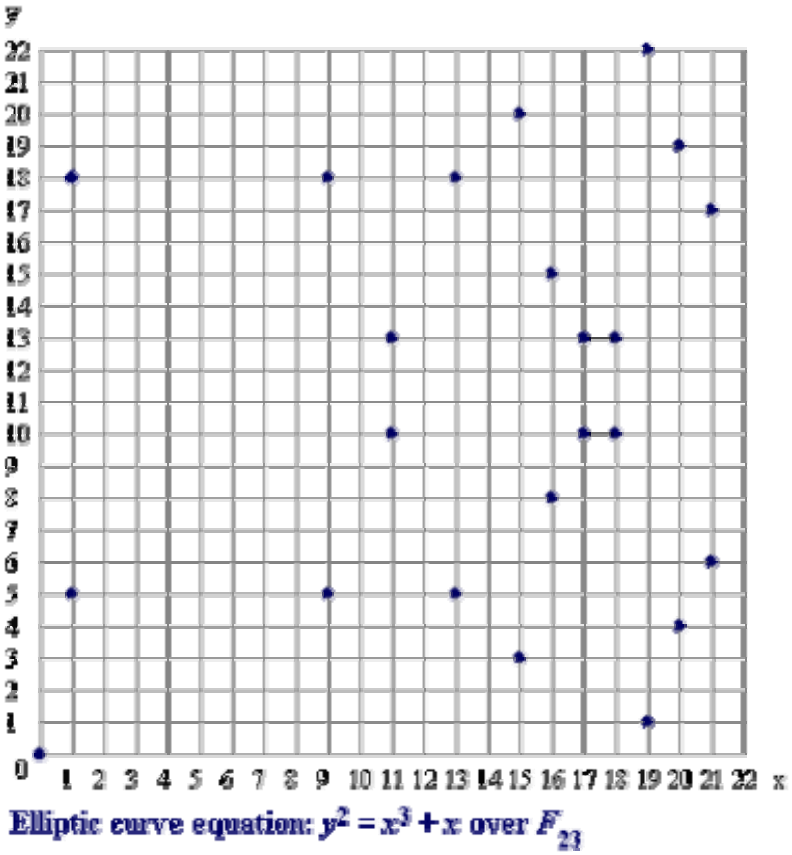
Calculations over the real numbers are slow and inaccurate due to round-off error. Cryptographic applications require fast and precise arithmetic; thus elliptic curve groups over the finite fields of F_p and F_{2m} are used in practice. Recall that the field F_p uses the numbers from 0 to p - 1, and computations end by taking the remainder on division by p. For example, in F_{23} the field is composed of integers from 0 to 22, and any operation within this field will result in integer also between 0 and 22.

An elliptic curve with the underlying field of F_p can formed by choosing the variables a and b within the field of F_p. The elliptic curve includes all points (x,y) which satisfy the elliptic curve equation modulo p (where x and y are numbers in F_p). If $x^3 + ax + b$ contains no repeating factors (or, equivalently, if $4a^3 + 27b^2$ mod p is not 0), then the elliptic curve can be used to form a group. An elliptic

curve group over F_p consists of the points on the corresponding elliptic curve, together with a special point O called the point at infinity. There are finitely many points on such an elliptic curve. Note the seemingly random spread of points for the elliptic curve over F_p.

Example of an Elliptic Curve Group over F_p

As a very small example, consider an elliptic curve over the field F_23. With a = 1 and b = 0, the elliptic curve equation is $y^2 = x^3 + x$. The point (9,5) satisfies this equation since: y^2 mod p = x^3 + x mod p 25 mod 23 = 729 + 9 mod 23 25 mod 23 = 738 mod 23 2 = 2 The 23 points which satisfy this equation are: (0,0) (1,5) (1,18) (9,5) (9,18) (11,10) (11,13) (13,5) (13,18) (15,3) (15,20) (16,8) (16,15) (17,10) (17,13) (18,10) (18,13) (19,1) (19,22) (20,4) (20,19) (21,6) (21,17) These points may be graphed as



Elliptic curve equation: $y^2 = x^3 + x$ over F_{23}

Note that there is two points for every x value. Even though the graph seems random, there is still symmetry about y = 11.5. Recall that elliptic curves over real numbers, there exists a negative point for each point which is reflected through the x-axis. Over the field of F_{23}, the negative components in the y-values are taken modulo 23, resulting in a positive number as a difference from 23. Here -P = (x_P, (-y_P Mod 23))

Note that these rules are exactly the same as those for elliptic curve groups over real numbers, with the exception that computations are performed modulo p.

7.3.6 Arithmetic in an Elliptic Curve Group over F_p

There are several major differences between elliptic curve groups over F_p and over real numbers. Elliptic curve groups over F_p have a finite number of points, which is a desirable property for cryptographic purposes. Since these curves consist of a few discrete points, it is not clear how to "connect the dots" to make their graph look like a curve. It is not clear how geometric relationships can be applied. As a result, the geometry used in elliptic curve groups over real numbers cannot be used for elliptic curve groups over F_p. However, the algebraic rules for the arithmetic can be adapted for elliptic curves over F_p. Unlike elliptic curves over real numbers, computations over the field of F_p involve no round off error - an essential property required for a cryptosystem.

Adding distinct points P and Q

The negative of the point P = (x_P, y_P) is the point -P = (x_P, -y_P mod p). If P and Q are distinct points such that P is not -Q, then P + Q = R where s = (y_P - y_Q) / (x_P - x_Q) mod p $xR = s^2 - x_P - x_Q$ mod p and $y_R = -y_P + s(x_P - x_R)$ mod p

Note that s is the slope of the line through P and Q.

Doubling the point P

Provided that y_P is not 0, 2P = R where s = ($3x_P^2$ + a) / ($2y_P$) mod p $x_R = s^2 - 2x_P$ mod p and $y_R = -y_P + s(x_P - x_R)$ mod p Recall that a is one of the

parameters chosen with the elliptic curve and that s is the slope of the line through P and Q.

Some of the facts of Elliptic Curve Groups over F_p

1). The elliptic curve equation $y^2 = x^3 + 10x + 5$ doesn't define a group over F_{17} since: = 4(10)3 + 27(5)2 mod 17 = 4675 mod 17 = 0 .

2). The points P(2,0) lie on the elliptic curve $y^2 = x^3 + x + 7$ over F_{17} as both side of the equation agree: (0)2 mod 17 = (2)3 + 2 + 7 mod 17 0 mod 17 = 17 mod 17 0 = 0. However, the point Q (6, 3) is not on the elliptic curve since the equation is false: (3)2 mod 17 = (6)3 + 6 + 7 mod 17 9 mod 17 = 229 mod 17 9 = 8, does not agree.

3). The negative of a point P = (xP, yP) on the elliptic curve points over F_{17} is the point -P = (xP, -yP mod p). Thus P(5,8)=-P(5,9),Q(3,0)= -Q(3,0).

4). In the elliptic curve group defined by $y^2 = x^3 + x + 7$ over F_{17},

i) The P + Q if P = (2,0) and Q = (1,3) is : s = (yP - yQ) / (xP - xQ) mod p = (-3) / 1 mod 17 = -3 mod 17 = 14 xR = s2 - xP - xQ mod p = 196 - 2 - 1 mod 17 = 193 mod 17 = 6 yR = -yP + s(xP - xR) mod p = 0 + 14*(2 - 6) mod 17 = -56 mod 17 = 12 Thus P + Q = (6,12)

ii) The 2P if P= (1, 3) is s = (3xP2 + a) / (2yP) mod p = (3 + 1) * 6-1 mod 17 = 4 * 3 mod 17 = 12 xR = s2 - 2xP mod p = 144 - 2 mod 17 = 142 mod 17 = 6 yR = -yP + s(xP - xR) mod p = -3 + 12 * (1 - 6) mod 17 = -63 mod 17 = 5 .Thus 2P = (6,5)

]

7.3.7 Elliptic Curve Groups over F_2m

Elements of the field F_2m are m-bit strings. The rules for arithmetic in F_2m can be defined by either polynomial_representation or by optimal normal basis representation. Since F_2m operates on bit strings, computers can perform arithmetic in this field very efficiently. An elliptic curve with the underlying field F_2m is formed by choosing the elements a and b within F_2m (the only condition is that b is not 0). As a result of the field F_2m having a characteristic 2, the elliptic curve equation is slightly adjusted for binary representation: $y^2 + xy = x^3 + ax^2 + b$

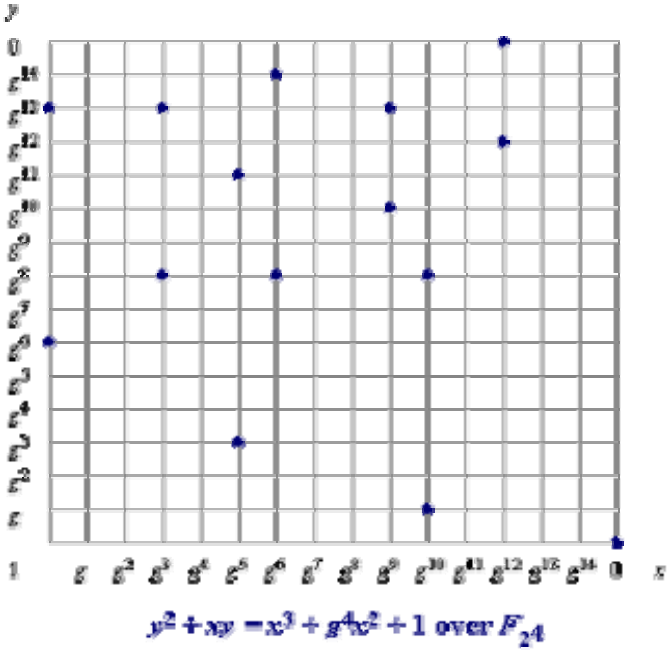
The elliptic curve includes all points (x,y) which satisfy the elliptic curve equation over F_2m (where x and y are elements of F_2m). An elliptic curve group over F_2m consists of the points on the corresponding elliptic curve, together with a point at infinity, O. There are finitely many points on such an elliptic curve. Addition with bit-strings is controlled by an XOR function.

An Example of an Elliptic Curve Group over F_2m

As a very small example, consider the field F_24, defined by using polynomial representation with the irreducible polynomial $f(x) = x^4 + x + 1$. The element g = (0010) is a generator for the field . The powers of g are: g^0 = (0001) g^1 = (0010) g^2 = (0100) g^3 = (1000) g^4 = (0011) g^5 = (0110) g^6 = (1100) g^7 = (1011) g^8 = (0101) g^9 = (1010) g^{10} = (0111) g^{11} = (1110) g^{12} = (1111) g^{13} = (1101) g^{14} = (1001) g^{15} = (0001)

In a true cryptographic application, the parameter m must be large enough to preclude the efficient generation of such a table otherwise the cryptosystem can be broken. In today's practice, m = 160 is a suitable choice. The table allows the use of generator notation (g^e) rather than bit string notation, as used in the following example. Also, using generator notation allows multiplication without reference to the irreducible polynomial $f(x) = x^4 + x + 1$.

Consider the elliptic curve $y^2 + xy = x^3 + g^4x^2 + 1$. Here $a = g^4$ and $b = g^0$ =1. The point (g^5, g^3) satisfies this equation over F_2m : $y^2 + xy = x^3 + g^4x^2 + 1$ $(g^3)^2 + g^5g^3 = (g^5)^3 + g^4g^{10} + 1$ $g^6 + g^8 = g^{15} + g^{14} + 1$ (1100) + (0101) = (0001) + (1001) + (0001) (1001) = (1001) The fifteen points which satisfy this equation are: (1, g^{13}) (g^3, g^{13}) (g^5, g^{11}) (g^6, g^{14}) (g^9, g^{13}) (g^{10}, g^8) (g^{12}, g^{12}) (1, g^6) (g^3, g^8) (g^5, g^3) (g^6, g^8) (g^9, g^{10}) (g^{10}, g) (g^{12}, 0) (0, 1) These points are graphed below:



Arithmetic in an Elliptic Curve Group over F_2m

Elliptic curve groups over F_2m have a finite number of points, and their arithmetic involves no round off error. This combined with the binary nature of the field, F_2m arithmetic can be performed very efficiently by a computer. The following algebraic rules are applied for arithmetic over F_2m :

Adding distinct points P and Q

The negative of the point P = (xP, yP) is the point -P = (xP, xP + yP). If P and Q are distinct points such that P is not -Q, then P + Q = R where s = (yP - yQ) / (xP + xQ)

xR = s^2 + s + xP + xQ + a and yR = s(xP + xR) + xR + yP As with elliptic curve groups over real numbers, P + (-P) = O, the point at infinity. Furthermore, P + O = P for all points P in the elliptic curve group.

Doubling the point P

If xP = 0, then 2P = O Provided that xP is not 0, 2P = R where s = xP + yP / xP xR = s^2+ s + a and yR = xP^2 + (s + 1) * xR .Recall that a is one of the

parameters chosen with the elliptic curve and that s is the slope of the line through P and Q.

Example of Elliptic Curve Groups over F_2m

As an example, assume the use of the field F_23. The field is described here using polynomial representation with the irreducible polynomial $x^3 + x + 1$. A generator for the field is g = (010), and the powers of g are: g^1 = (010) g^2 = (100) g^3 = (011) g^4 = (110) g^5 = (111) g^6 = (101) g^7 = (001) = 1

1). The elliptic curve equation $y^2 + xy = x^3 + g^5x^2 + g^6$ does not define a group over F_23 Since the parameter b = 6 is not zero.

2). The point $P(g^3,g^6)$ is on the elliptic curve $y^2 + xy = x^3 + g^2x^2 + g^6$ over F_23 since that equation holds true: $(g^6)^2 + (g^3)(g^6) = (g^3)^3 + g^2(g^3)^2 + g^6g^5 + g^2 = g^2 + g + g^6(111) + (100) = (100) + (010) + (101)$ (011) = (011) $g^3 = g^3$

However, the point $Q(g^5)(g^2)$ is not on the elliptic curve, since the equation disagrees: $(g^2)^2 + (g^5)(g^2) = (g^5)^3 + g^2(g^5)^2 + g^6\ g^4 + 1 = g + g^5 + g^6(110) + (001) = (001) + (111) + (101)$ (111) = (000) $g^5 = 0$ which is false.

3). The negatives of the elliptic curve points (xP,yP) over F_23 is defined by (xP, xP + yP) Thus $-(P(g^3,g^6) = (g^3, g^3 + g^6) = (g^3, g^4)$

$$-Q\ (g,0) = = (g, g + 0) = (g, g)$$

4). In the elliptic curve group defined by $y^2 + xy = x^3 + g^2x^2 + g^6$ over F_23,

i). P + Q if $P = (g^2,g^6)$ and $Q = (g^5,g^5)$ is $s = (y^P - y^Q) / (x^P + x^Q) = (g^6 + g^5) / (g^2 + g^5) = g / g^3 = g\text{-}2 = g^5$ $x^R = s^2 + s + x^P + x^Q + a = g^3 + g^5 + g^2 + g^5 + g^2 = g^3$ $y^R = s(x^P + x^R) + x^R + y^P = g^5 * (g^2 + g^3) + g^5 + g^6 = g^5 * g^5 + g^3 + g^6 = g^3 + g^3 + g = g^6$ Thus P + Q = (g^3, g^6)

ii). 2P if P = (g^3, g^4) is $s = x^P + y^P / 2x^P = g^3 + g^4 / g^3 = g^3 + g = 1$ $x^R = s^2 + s + a = 1 + 1 + g^2 = g^2$ $y^R = x^P + (s + 1) * x^R = g^3 + 0 * g^2 = g^3$ .Thus 2P = (g^2, g^3)

Chapter-8

Elliptic Curve Groups and The Discrete Logarithm Problem

At the foundation of every cryptosystem is a hard mathematical problem that is computationally infeasible to solve. The discrete logarithm problem is the basis for the security of many cryptosystems including the Elliptic Curve Cryptosystem. More specifically, the ECC relies upon the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP).

Recall that we examined two geometrically defined operations over certain elliptic curve groups. These two operations were point addition and point doubling. By selecting a point in a elliptic curve group, one can double it to obtain the point 2P. After that, one can add the point P to the point 2P to obtain the point 3P. The determination of a point nP in this manner is referred to as Scalar Multiplication of a point. The ECDLP is based upon the intractability of scalar multiplication products.[5]

8.1 Scalar Multiplication

The following animation demonstrates scalar multiplication through the combination of point doubling and point addition. While it is customary to use additive notation to describe an elliptic curve group , some insight is provided by using multiplicative notation. Specifically, consider the operation called "scalar multiplication" under additive notation: that is, computing kP by adding together k copies of the point P. Using multiplicative notation, this operation consists of multiplying together k copies of the point P, yielding the point P*P*P*P&.*P = Pk.

8.2 The Elliptic Curve Discrete Logarithm Problem

In the multiplicative group Zp*, the discrete logarithm problem is: given elements r and q of the group, and a prime p, find a number k such that r = qk mod p. If the elliptic curve groups is described using multiplicative notation, then the elliptic curve discrete logarithm problem is: given points P and Q in the group, find a number that Pk = Q; k is called the discrete logarithm of Q to the base P. When the elliptic curve group is described using additive notation, the elliptic curve discrete logarithm problem is: given points P and Q in the group, find a number k such that Pk = Q

Example:

In the elliptic curve group defined by $y^2 = x^3 + 9x + 17$ over F_{23}, one (naïve) way to find the discrete logarithm k of Q = (4,5) to the base P = (16,5) is to compute multiples of P until Q is found. The first few multiples of P are: P = (16,5) 2P = (20,20) 3P = (14,14) 4P = (19,20) 5P = (13,10) 6P = (7,3) 7P = (8,7) 8P = (12,17) 9P = (4,5) Since 9P = (4,5) = Q, the discrete logarithm of Q to the base P is k = 9. In a real application, k would be large enough such that it would be infeasible to determine k in this manner.

Chapter-9

Elliptic Curve Encryption/Decryption

As with the key exchange system, an encryption / decryption system requires a point *G* and an elliptic group E_q (*a, b*) as parameters. Each user *A* selects a private key n_A and generates a public key $P_A = n_A$ x G.

To encrypt and send a message P_m to *B, A* chooses a random positive integer k and produces the ciphrtext C_m consisting of the pair of points

$$C_m = \{kG, P_m + k P_B\}$$

Note that A has used B's public key. To decrypt the cipher text, B multiplies the first point in the pair by B's secret key and subtracts the result from the second point:

A has masked the message P by adding to it. Nobody but A knows the value of k, so even though is a public key, nobody can remove the mask. However, A also includes a "clue," which is enough to remove the mask if one knows the private key. For an attacker to recover the message, the attacker would have to compute k given G and kG, which is assumed hard.

As an example of the encryption process (taken from [5], take p = 751; E (-1, 188), which is equivalent to the curve +188; and G = (0, 376). Suppose that A wishes to send a message to B that is encoded in the elliptic point P = (562,201) and that A selects the random number k = 386. B's public key is P = (201, 5). We have 386 (0, 376) = (676, 558), and (562, 201) + 386 (201, 5) = (385, 328). Thus A sends the cipher text [(676, 558), (385, 328)]

Chapter-10

Symmetric Key Cryptosystem using Elliptic curves

In this project one symmetric key cryptosystem using elliptic curve has been implemented. The principle of symmetric key ciphers is to use the same key for encryption and decryption.

Steps for Encryption

1).The random bits is concatenated with every block of data to be encrypted. Advantage is that it does not affect the decryption process and ensures that the same message (or data block) is never encrypted the same way twice. The cost is message expansion, which is another reason this is impractical.

A source curve and point are set up as constants within the program.

2).Second step is accomplished by multiplying the key times the source point over the source curve.

3).Now we add the block result of step 1 with the result of step 2.

Cipher text=M*N+KP

Where M is the text to be encrypted

N is the random number

P is the point on the curve.

K is the key.

* is the Concatenation

At the decryption point we subtract the KP from the cipher text .

M*N+KP – KP = M*N

Now we can easily get M as numbers of bits of random bits are fixed.

Chapter-11

An Introduction of Rijndael

The DES algorithm has become obsolete and is in need of replacement. To this end the National Institute of Standards and Technology (NIST) has been holding a competition to develop the Advanced Encryption Standard (AES) as a replacement for DES. Triple DES has been endorsed by NIST as a temporary standard to be used until the AES is finished sometime in 2001.

NIST has been working very closely with industry and the cryptographic community during the development of the Advanced Encryption Standard. The overall goal is to develop a Federal Information Processing Standard (FIPS) that specifies an encryption algorithm (or algorithms) capable of protecting sensitive government information well into the next century. The algorithm(s) is expected to be used by the U.S. Government and, on a voluntary basis, by the private sector.

On January 2, 1997, NIST announced the initiation of the AES development effort and made a formal call for algorithms on September 12 of that year. The call stipulated that the AES would specify an unclassified, publicly disclosed encryption algorithm(s), available royalty-free, worldwide. In addition, the algorithm(s) must implement symmetric key cryptography as a block cipher and (at a minimum) support block sizes of 128 bits and key sizes of 128, 192, and 256 bits.[8]

On August 20, 1998, NIST announced a group of fifteen AES candidate algorithms at the First AES Candidate Conference (AES1). These algorithms had been submitted by members of the cryptographic community from around the world. At that conference and in a simultaneously published Federal Register notice, NIST solicited public comments on the candidates. A Second AES Candidate Conference (AES2) was held in March 1999 to discuss the results of the analysis conducted by the global cryptographic community on the candidate algorithms. The public comment period on the initial review of the algorithms closed on April 15, 1999. Using the analyses and comments received, NIST selected five algorithms from the original fifteen submissions.

The AES finalist candidate algorithms are MARS, RC6, Rijndael, Serpent, and Two fish. Four of the algorithms (MARS, Rijndael, Serpent, and Twofish) are supported by Private Encryptor.

The results of the five finalists can be summarized as shown below in Figure 11.1:

Area\Algorithm	*Rijndael*	*Mars*	*RC6*	*Serpent*	*Twofish*
General Security	× ×	× × ×	× ×	× × ×	× × ×
Implementation of Security	× × ×	×	×	× × ×	× ×
Software Performance	× × ×	× ×	× ×	×	×
Smart Card Performance	× × ×	×	×	× × ×	× ×
Hardware Performance	× × ×	×	× ×	× × ×	× ×
Design Features	× ×	× ×	×	×	× × ×

Figure 11.1. Summary of AES Finalist

With this overall analysis, Rijndael was named the AES on October 2, 2000.

11.1 Terminology

There are terms that are frequently used throughout this paper that need to be clarified.

Block: AES is a block cipher. This means that the number of bytes that it encrypts is fixed. AES can currently encrypt blocks of 16 bytes at a time; no other block sizes are presently a part of the AES standard. If the bytes being encrypted are larger than the specified block then AES is executed concurrently.

This also means that AES has to encrypt a minimum of 16 bytes. If the plain text is smaller than 16 bytes then it must be padded.

Simply said the block is a reference to the bytes that are processed by the algorithm.

State :Defines the current condition (state) of the *block*. That is the block of bytes that are currently being worked on. The state starts off being equal to the block, however it changes as each round of the algorithms executes. Plainly said this is the block in progress.

XOR : Refers to the bitwise operator **Exclusive Or**. XOR operates on the individual bits in a byte in the following way:

0 XOR 0 = 0
1 XOR 0 = 1
1 XOR 1 = 0
0 XOR 1 = 1

For example the Hex digits D4 XOR FF

11010100
XOR 11111111
= 00101011 (Hex 2B)

Another interesting property of the XOR operator is that it is reversible. So Hex 2B XOR FF = D4

Most programming languages have the XOR operator built in.

Programming Language	**XOR Operator**
C	^
C++	^
C#	^
Java	^
Visual Basic	XOR

HEX: Defines a notation of numbers in base 16. This simply means that; the highest number that can be represented in a single digit is 15, rather than the usual 9 in the decimal (base 10) system.

11.2 The Rijndael Algorithm

At its simplest, Rijndael is a Square block cipher with variable block and key sizes. It supports 128-bit, 192-bit, and 256-bit blocks and keys. The structure of Rijndael was designed with the following in mind:

"Resistance against all known attacks;

Speed and code compactness on a wide range of platforms;

Design simplicity;"

This key is expanded into individual sub keys, a sub keys for each operation round. This process is called KEY EXPANSION, which is described later.

An iteration of the above steps is called a round.

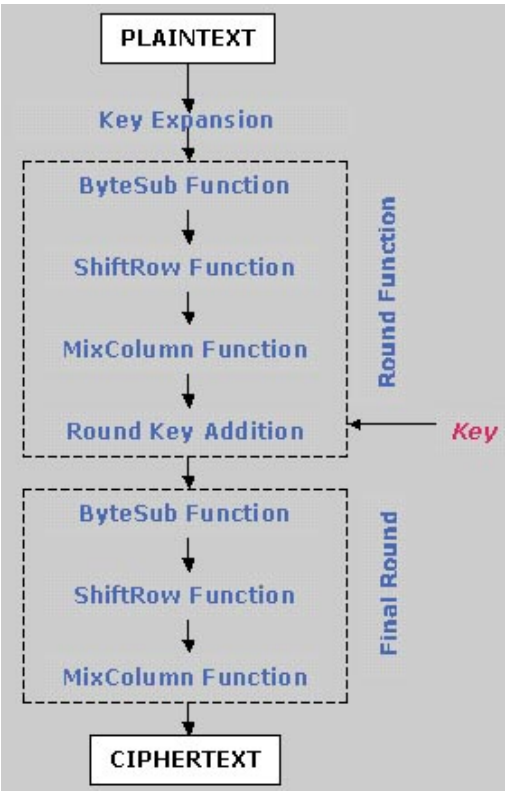


Figure 11.2 Outline of One Round of Rijndael

This is one round of Rijndael. The amount of rounds of the algorithm depends on the key size.

Key Size (bytes)	Block Size (bytes)	Rounds
16	16	10
24	16	12
32	16	14

Fig 11.3 Number of Rounds based on block and key sizes

For each block, Rijndael performs an initial key expansion which produces the "State" .The State is a 2x2 array of 4-byte words consisting of four rows. The number of columns is block size divided by 32. It then performs the round function and a final round on the State to produce the ciphertext. The round function includes four functions: ByteSub, ShiftRow, MixColumn, and AddroundKey. After the round function has performed the specific number of rounds, the result is processed in a final round. This final round is the same as each round function except the mixColumn function is not performed.

11.3 AES Cipher Functions

11.3.1 The ByteSub transformation

The ByteSub Transformation is a non-linear byte substitution, operating on each of the State bytes independently. The substitution table (or S-box) is invertible and is constructed by the composition of two transformations:

1. First, taking the multiplicative inverse in GF(2^8), with the representation defined previously. ‘00’ is mapped onto itself.

2. Then, applying an affine (over GF(2^8)) transformation defined by:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

The application of the described S-box to all bytes of the State is denoted by:

ByteSub(State) .

Figure 11.4 illustrates the effect of the ByteSub transformation on the State.

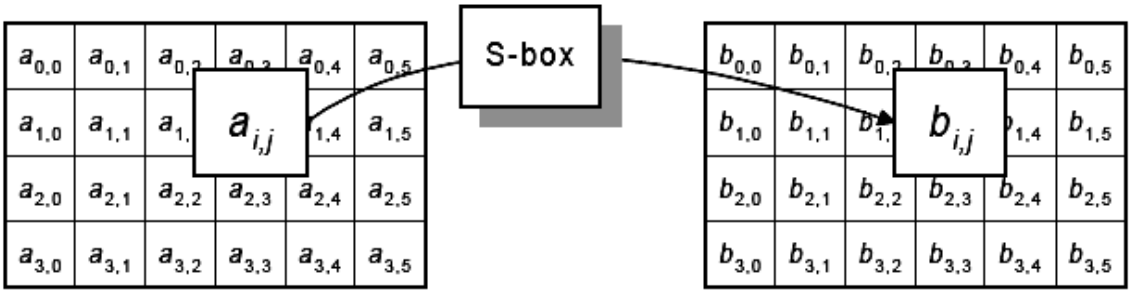


Figure 11.4: ByteSub acts on the individual bytes of the State.

The inverse of ByteSub is the byte substitution where the inverse table is applied. This is

obtained by the inverse of the affine mapping followed by taking the multiplicative inverse in GF(2^8).

11.3.2 The ShiftRow transformation

In ShiftRow, the rows of the State are cyclically shifted over different offsets. Row 0 is not shifted, Row 1 is shifted over C1 bytes, row 2 over C2 bytes and row 3 over C3 bytes. The shift offsets C1, C2 and C3 depend on the block length **Nb**. The different values are specified in Table 2.

Nb	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

Table 5: Shift offsets for different block lengths.

The operation of shifting the rows of the State over the specified offsets is denoted by:

ShiftRow(State)

Figure 11.5 illustrates the effect of the ShiftRow transformation on the State.

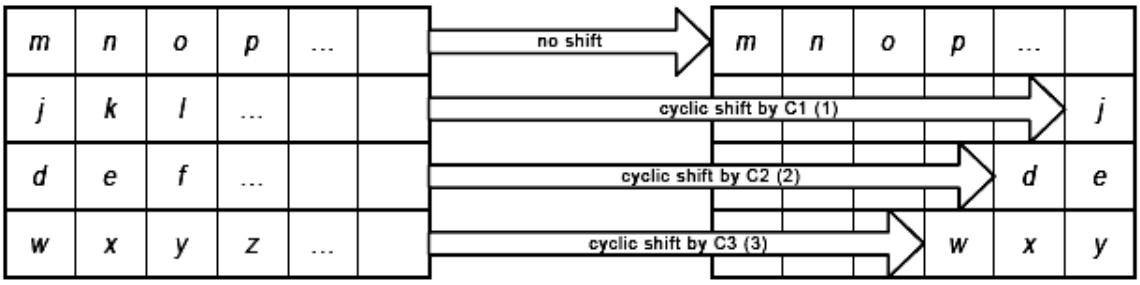


Figure 11.5: ShiftRow operates on the rows of the State.

The inverse of ShiftRow is a cyclic shift of the 3 bottom rows over **Nb**−C1, **Nb**-C2 and **Nb** − C3 bytes respectively so that the byte at position j in row i moves to position (j + **Nb**−Ci) mod **Nb**.

11.3.3 The MixColumn transformation

In MixColumn, the columns of the State are considered as polynomials over GF(2^8) and multiplied modulo x4 + 1 with a fixed polynomial c(x), given by

$$c(x) = \text{‘03’}\ x^3 + \text{‘01’}\ x^2 + \text{‘01’}\ x + \text{‘02’}\ .$$

This polynomial is coprime to $x^4 + 1$ and therefore invertible. This can be written as a matrix multiplication. Let b(x) = c(x) $\otimes$ a(x),

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

The application of this operation on all columns of the State is denoted by MixColumn(State).

Figure 11.6 illustrates the effect of the MixColumn transformation on the State.

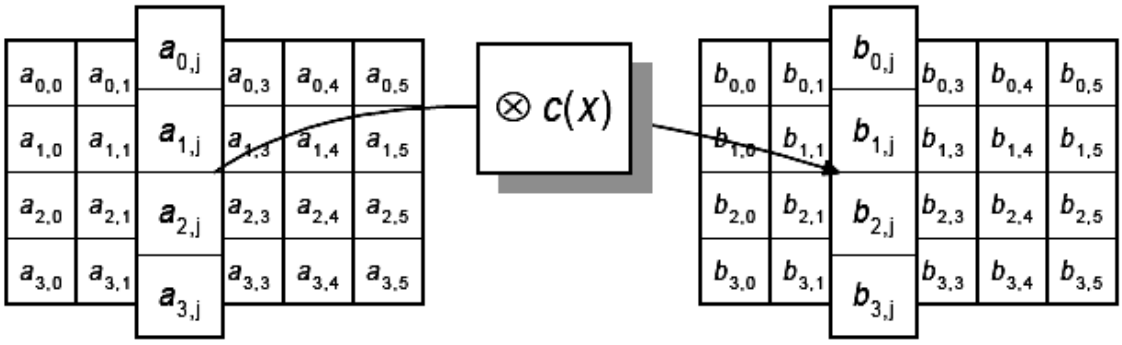


Figure11.6: MixColumn operates on the columns of the State.

The inverse of MixColumn is similar to MixColumn. Every column is transformed by multiplying it with a specific multiplication polynomial d(x), defined by

(‘03’ x^3 + ‘01’ x^2 + ‘01’ x + ‘02’) $\otimes$ d(x) = ‘01’ .

It is given by:

$d(x) = \text{'0B'}\ x^3 + \text{'0D'}\ x^2 + \text{'09'}\ x + \text{'0E'}$.

11.3.4 The Round Key addition

In this operation, a Round Key is applied to the State by a simple bitwise EXOR. The Round Key is derived from the Cipher Key by means of the key schedule. The Round Key length is equal to the block length **Nb**. The transformation that consists of EXORing a Round Key to the State is denoted by: AddRoundKey(State,RoundKey).

This transformation is illustrated in Figure 11.7.

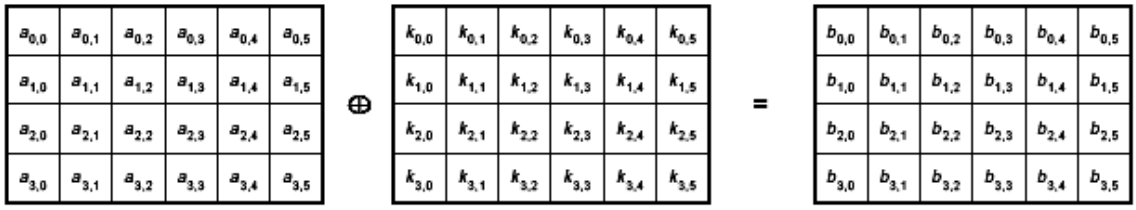


Figure 11.7: In the key addition the Round Key is bitwise EXORed to the State.

AddRoundKey is its own inverse.

11.4 AES Key Expansion

Prior to encryption or decryption the key must be expanded. The expanded key is used in the **Add Round** Key function defined above.

Each time the Add Round Key function is called a different part of the expanded key is XORed against the state. In order for this to work the Expanded Key must be large enough so that it can provide key material for every time the Add Round Key function is executed. The Add Round Key function gets called for each round as well as one extra time at the beginning of the algorithm.

There fore the size of the expanded key will always be equal to:

16 * (number of rounds + 1).

The 16 in the above function is actually the size of the block in bytes. This provides key material for every byte in the block during every round +1

Key Size (bytes)	Block Size (bytes)	Expanded Key (bytes)
16	16	176
24	16	208
32	16	240

Table 6 Expanded key size on the basis of key size and block size.

Since the key size is much smaller than the size of the sub keys, the key is actually “stretched out” to provide enough key space for the algorithm.

The key expansion routine executes a maximum of 4 consecutive functions. These functions are:

ROT WORD
SUB WORD
RCON
EK
K

An iteration of the above steps is called a round. The amount of rounds of the key expansion algorithm depends on the key size.

Key Size (bytes)	Block Size (bytes)	Expansion Algorithm Rounds	Expanded Bytes / Round	Rounds Key Copy	Rounds Key Expansion	Expanded Key (bytes)
16	16	44	4	4	40	176
24	16	52	4	6	46	208
32	16	60	4	8	52	240

Table 7 Number of rounds of key expansion on the basis of key size.

The first bytes of the expanded key are always equal to the key. If the key is 16 bytes long the first 16 bytes of the expanded key will be the same as the original key. If the key size is 32 bytes then the first 32 bytes of the expanded key will be the same as the original key.

Each round adds 4 bytes to the Expanded Key. With the exception of the first rounds each round also takes the previous rounds 4 bytes as input operates and returns 4 bytes.

One more important note is that not all of the 4 functions are always called in each round. The algorithm only calls all 4 of the functions every:

4 Rounds for a 16 byte Key
6 Rounds for a 24 byte Key

8 Rounds for a 32 byte Key

The rest of the rounds only a **K** function result is XORed with the result of the **EK** function. There is an exception of this rule where if the key is 32 bytes long an additional call to the **Sub Word** function is called every 8 rounds starting on the 13th round.

11.4.1 AES Key Expansion Functions

Rot Word (4 bytes)

This does a circular shift on 4 bytes similar to the Shift Row Function.

1,2,3,4 to **2,3,4,1**

Sub Word (4 bytes)

This step applies the S-box value substitution as described in **Bytes Sub** function to each of the **4** bytes in the argument.

Rcon((Round/(KeySize/4))-1)

This function returns a 4 byte value based on the following table

EK(Offset)

EK function returns 4 bytes of the Expanded Key after the specified offset. For example if offset is 0 then EK will return bytes 0,1,2,3 of the Expanded Key

K(Offset)

K function returns 4 bytes of the Key after the specified offset. For example if offset is 0 then **K** will return bytes 0,1,2,3 of the Expanded Key

Chapter-12
IMPLEMENTATION DETAIL

In this project three cryptosystem has been implemented –elliptic curve public key cryptosystem, symmetric key elliptic curve cryptosystem and AES using 'C' programming language.

12.1 ECC implementation detail.

As already described there are several reasons why curves over $F(2^n)$ are preferred, so it has been decided to use them in the elliptic curve cryptosystem implementation. There are three different bases, which can be used to implement fields of characteristic two: polynomial base, normal base, and subfield base, this project is implemented using polynomial base.

The elliptic curve operations require addition, multiplication, squaring and inversion in the underlying field.

Because of polynomial representation all the operation like addition, subtraction and multiplication and inversion can be executed efficiently.

For e.g.

The addition operation is simply by XOR operation.

The subtraction is also by XOR operation.

The Multiplication is done by shifting left.

The division is done by inversion method.

Now some structure which I have used and some important function

1).Curve- contain

```
short int form
BIGINT  a2
BIGINT a6
```

2).Point -

```
BIGINT  x;
BIGINT  y;
```

3).PubKey-

```
POINT p;
POINT q;
CURVE crv;
```

Some important function

1).Shift_left - perform the left shift

2).shift_right – perform the right shift

3).rot_left - perform the left rotation

4) rot_right - perform the right rotation

5).mult - Multiply the two fields

6).inv - perform the inversion operation.
7).esum - add two points
8).esub - subtract the two points
9).edbl - doubled the point
10).emul. – Multiply the two pints.
11)savekey-write the key in the file
12)getkey - get the key from the file
13).keygen- generate the Public\private key
14) printkey-print the key
15elliptic_encrypt- this function encrypt the session key.
16).elliptic_decrypt- this function decrypt the encrypted key.

12.2 AES implementation detail

These are some important modules which are created

1. crypt.c
    - PopulateKeyMaterial
    - ConvertKey
    - This file also call BlockEncrypt and BlockDecrypt

2. rijndael-api-fst.c
    - Makekey
    - blockEncrypt-blockEncrypt function encrypt the block by calling rijndaelEncrypt
    - blockdecrypt-blockDecrypt function encrypt the block by calling rijndaelDecrypt
    - cipherUpdateRounds – this function update the state by calling rijndaelEncryptRound

3. Rijndael-alg-fst.c
    - Mixcolumn-Mix the four bytes of every column in a linear way
    - InvMixcolumn –Mix the four bytes of every column in a linear way this is the opposite of mixcolumn
    - Substitution-Replace every byte of the input by the byte at that place in the nonlinear s-box
    - KeyAddition-Exor corresponding text input and round key inpu bytes

- Shiftrow-shift the row by a variable amount.
- RijndaelencryptRound-Encrypt only a certain number of round
- Rijndaelencrypt-Encryption of one block
- rijndaelDecrypt-Decryption of one block
- RijndaelDecryptRound-Decrypt onlt a certain number of round

Chapter-13

Comparison of AES and ECC symmetric cipher.

Comparison of AES and ECC:-

AES is the Standard which is used almost in every field where encryption is used. For e.g. all the technique- SET, SSL, iKP, Electronic Money used AES and that's why ECC symmetric key cryptosystem has been compared with AES.

For the comparison of the AES and the ECC symmetric cipher I have taken 933MHz Pentium III processor.

For Encryption

S.No.	Data Size (KB)	AES symmetric cipher (Sec)	ECC Symmetric cipher (Sec)
1	.16	.000001	2.25
2	1.6	.000012	19.97
3	16	.00023	200.9
4	160	.005	1997.23
5	1600	.019	20003.43
6	16000	2.47	198986.21
7	160000	19.5	2342322.43

Table 8: Comparison of AES and ECC

For decryption it is taking approximately same time.

It's clear from the table that AES is very fast then ECC symmetric cipher.

Chapter 14 TESTING

Introduction to Testing:

Software Testing is a critical element of software quality assurance and represents the ultimate review of the software designed. Testing is the process of executing a program with the intent of finding errors. Basically there are two testing strategies. Code testing: where the strategy examines the logic of the program. Specification testing: where the program is treated as black box and so the instruction path is not considered.

Testing has been done at various levels such as Unit Testing ,Integration Testing, System Testing ,Acceptance Testing ,Validation.

In unit testing, where a unit will be taken at a time and tested thoroughly at the statement level to find maximum possible errors by employing Visual Basic .

In integration testing, many test modules are combined into sub systems, which are tested for any incompatibilities and errors.

In system testing, the entire system is tested. The test cases are selected as per the functional requirements specified. Each module was supplied with relevant and irrelevant values to test efficiency and all are found to be working satisfactorily.

Acceptance testing is performed with realistic data to demonstrate that the software is working satisfactorily.

To spot all the structural, syntactical and integration errors in the code, a series of test cases are tested for and all of them were answered.

The test cases are given below:

Test Case	When an invalid path is given
Expected Result	System should display an error message followed by a screen to re-enter the path
Observed Result	The system displayed the error message

Table 9 : Test case 1 – Invalid Path

Test Case	When the wrong key is given decryption.
Expected Result	System should display the error message
Observed Result	The system displayed the error message.

Table 10: Test case when wrong key is given.

Test Case	When a file not encrypted using the system was included in the files for decryption
Expected Result	System should recognize the file and should not attempt to decrypt it.
Observed Result	The system recognized the file, skipped it and proceeded to the next file.

Table 11: Test case 3: Not Encrypted

CONCLUSION

The system demonstrates the use of hybrid techniques in the development of a data encryption application. The techniques used to develop this application can be used to develop a Stream based encryption system.

After careful execution of the implemented, we have observed that both the algorithms can be safely and successfully used to encrypt/decrypt any type of file. As the algorithms implemented are relatively new, they present considerable difficulty for new forms of cryptanalysis. The overhead added to the encrypted file is minuscule amounting to less than 1% of the size of a moderate sized text file.

The project has been tested for the required results. It did well as far as the performance part is concerned even with lengthy keys. To stretch the limits of the application, a couple of audio files (each about 4 MB in size) and one video file (around 20 MB in size) were given for encryption and the job was done under 20 seconds. Decryption was also equally fast.

SCOPE FOR FURTHER STUDY

The application demonstrates the use of a moderately complex hybrid algorithm for data encryption –ECC and AES. The future development of the application may use a more complex hybrid cipher techniques.

This project has already implemented the arithmetic operation like addition, subtraction, division efficiently but still some research can be done to make them better,

Although the project has been implemented as a stand-alone system it can also be implemented on the network environment.

The symmetric based ECC cryptosystem is somewhat slow, so some more research is needed to make it fast.

Appendix :

****

CRYPT

The Ultimate File Encryption Software

****

What is new ?

Our software can encrypt any type of file.
(.txt, .dat, .bmp, .jpg, .gif, .mp3.....simply any thing.)

It is more powerful than any of its peers. It uses a powerful Hybrid encryption algorithm that gives more security to your files without compromising on the performance issues.

Introduction

It is the ultimate tool to protect all your (secret) files.

IT is easy to use file encryption software. It protects your valuable data by encrypting files and attaching a password to them. These files cannot be read without decrypting them with the password first.

It is a powerful, yet compact program that encrypts any type of file and makes them Unintelligible to others.
Nevertheless, if you need additional information, Please go through the following sections.

I) Installation
II) System Requirements
III) How to Use ?
IV) Security

I) Installation :

Our software comes with a single executable file
crypt.exe which can be executed directly.

II) System Requirements :

Our Programs should run on any IBM Compatible with out any fuss. Even an obsolete 386 machine is good enough. But a PC with the configuration is recommended:

Pentium I or higher Processor
Minimum 16 MB RAM
Windows 95 or Later O.S

CRYPT is fully compatible with Windows 95, Windows 98, Windows NT 3.5, Windows NT 4.0 and Windows XP. They are implemented using C,

Our software runs totally in Character Mode and so assumes nothing about your Graphics Adapter.

III) How to Use ?

.

CRPYT.exe-

Crypt.exe is a single file which can be executed directly.

In this first the user have to generate the Public key and secret key by passing the pass phrase which is used by the hash function for generating the keys. And then the public key is saved in the file specified by the user..

Ones the keys are generated the Encryption process starts: In this the user enter the input file name and out file name. The session key is generated by the software itself and the same used for encryption process and the encrypted output stored in the output file.
Now the session key is encrypted and stored in the user specified file name.
Here encryption process ends.

For decryption process again software is to be run.

Now user have to enter the filename in which public key and encrypted session key is stored. On getting both the software will ask for the pass phrase, on getting it the software will generate the secret key and also check it whether it is correct or not by using public key. Now the system decrypts the encrypted session key using AES. Now the users enter the decrypted file and the output file and the software will decrypt the file and stored the result in the output file.

Security

Because of the use of ECC and AES, the software gives great level of security.

REFERENCES :

1). Hand Book of Cryptography

Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone

CRC Press

2).Cryptography Theory and Practice

Douglas .R. Stinson

March, 1995, by CRC Press, Inc

3).THE CODE BOOK.

The Secret History of Codes and Code-Breaking

Fourth Estate Publications.UK.

4).Applied Cryptography: Protocols, Algorithms, and Source Code in C,

2nd Edition

Bruce Schneier

1999 John Wiley & Sons

5).Implementing Elliptic Curve Cryptography

Michael Rosing

1998, Manning Publications Company

6).Cryptography and E-Commerce

A Wiley Tech Brief

JonC. Graff

2000 John Wiley & Sons

7).THE ADVANTAGES OF ECC FORWIRELESS SECURITY

KRISTIN LAUTER, MICROSOFT CORPORATION

8)."Linear cryptanalysis method for AES cipher," Advances in Cryptology, Proceedings Eurocrypt'93, LNCS 765, T. Helleseth, Ed., Springer-Verlag, 1994.

9).J. Kelsey, B. Schneier, D. Wagner and Chris Hall, "Cryptanalytic attacks on pseudorandom number generators," Fast Software Encryption, LNCS 1372, S. Vaudenay, Ed.
Springer-Verlag, 1998.

10)."New types of cryptanalytic attacks using related keys," Advances in
Cryptology, Proceedings Eurocrypt'93, LNCS 765, T. Helleseth, Ed., Springer-Verlag, 1993,

11).J. Daemen, L.R. Knudsen and V. Rijmen, "The block cipher Square," Fast Software Encryption, LNCS 1267, E. Biham, Ed., Springer-Verlag, 1997.
available as http://www.esat.kuleuven.ac.be/rijmen/square/fse.ps.gz.

12).Certicom Corp., "The Elliptic Curve Cryptosystem", Certicom whitepaper, http://www.certicom.com, July
2003