



Enhanced Method for RSA Cryptosystem Algorithm

طريقة مطورة في خوارزمية تشفير المفتاح العام RSA

Ibrahem Abdallah Aldariseh

Supervisor

Prof .Dr . Alaa Al-Hamami

**A Thesis submitted in partial fulfillment of the
requirements for the
Degree of Master Science in Computer Science
Computer Science Department
College of Computer Sciences and Informatics
Amman Arab University
October 2011**

Authorization

I, ibrahem Abdullah aldarisch, authorize Amman Arab University the right to provide copies of the dissertation to libraries, institutes, agencies or individual when necessary.

Name : ibrahem Abdullah aldarisch

Signature: 


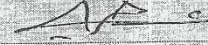

Date: 29.10.2011

The Discussion Committee Decision

This dissertation entitled "Enhanced Method for RSA Cryptosystem Algorithm" was discusses and passed on October 01,2011.

Discussion Committee Members:

Signature:

التوقيع والتاريخ		الإسم الثلاثي
	رئيساً	أ.د. مؤيد عبد الرزاق الزبيدي
	عضواً ومشرفاً	أ.د. علاء عبد الحكيم
	عضواً	أ.م.د. محمد طه
	عضواً	

Dedication

To my father and mother....

To my wife....

To my sons....

Acknowledgement

I would like to thank my supervisor, Prof. Alaa Hussain Al-Hamami, for his invaluable guidance and support throughout the entire process of this thesis research work and for his important suggestions during my research.

Also I would like to thank my family, colleagues and friends for their great support.

Table of contents

Dedication	a
Acknowledgement	b
Table of contents	c
List of Figures	e
List of Tables	h
Abstract	l
Arabic Summary	n
Chapter One {Introduction}	1
1.1 Introduction	1
1.2 Cryptology	3
1.3 Encryption	5
1.4 Network Encryption:	6
1.5. Symmetric Encryption Technology:	7
1.6. Private Key Encryption	8
1.6.1 Public Key encryption	10
1.6.2. Rivest, Shamir, and Adelman (RSA) Algorithm	10
1.6.3 The RSA cryptosystem	12
1.6.4 Security Attacks	13
1.6.5. Attacks on RSA	17
1.6.6 Cryptography Strength of RSA	22
1.6.7 Weaknesses in RSA	23
1.7 Statement of Problem:	24
1.8. Contribution	25
1.9. Thesis Organization	27
Chapter two {Theory & Related Works }	29
2.1. Introduction	29
2.2. Two Categories of Attacks On RSA	30
2.2.1. Mathematical Attacks on RSA	30
2.2.2. Implementation Attacks on RSA	34
2.3.1. Quadratic Sieve.	44
2.3.2. General Number Field Sieve	45
2.4. Literature Reviews	47
"2.4.1. Efficient Method For Breaking RSA Scheme", 2008, [7]	47
"2.4.2. On Using RSA with low exponent in a public key Network", 1986, [37].	49
2.4.3. "Attack on the Cryptographic Scheme", 1994, [38].	52
2.4.4. Factorization of a 512(bit RSA Modulus), 1999, [9].	53
2.4.5. "CRYPTANALYSIS OF RSA USING ALGEBRAIC AND LATTICE	
METHODS", 2002, [48].	54
Chapter three {The Proposed Method}	58

3.1. Introduction	58
3.2. The RSA cryptosystem	58
3.2.1 Mathematics of the RSA Algorithm	58
3.2.2. Proof of the RSA Algorithm.....	61
3.2.3. How It Works	65
3.2.4. Analysis of Rsa Cryptosystem.....	68
3.3.. The New Approach RSA cryptosystem	70
3.3.1 Mathematics of the New Approach RSA Algorithm.....	74
3.3.2. Proof of the New Approach RSA Algorithm	75
3.3.3. ANALYSIS of the RSA Enhance.....	79
3.3.4. Examples of the RSA Enhance.....	81
3.4. Breaking the Original RSA <i>and the RSA Enhance</i>	83
3.4.1. Pollard rho Factoring Method.....	84
3.5. Summary	86
Chapter four The Experimental Work.....	87
4.1. Introduction	87
4.2. Implementations	88
4.2.1.. First application (Encryption & Decryption Text)	88
4.2.2.. Second application:	104
Summary	118
Chapter Five Conclusion and Future Works	119
5.1. Formalistic Appraising	119
5.2. Conclusion	121
3.5. Future Works	122
References	124

List of Figures

Chapter One

- Figure 1.1: Overview on Cryptology 3
- Figure 1.2: Simplified Model of Conventional Encryption 7
- Figure 1.3: Public-Key Cryptography 9
- Figure 1.4: Normal traffic flow pattern from source to destination 13
- Figure 1.5: Traffic interruption from source to destination by an attack 14
- Figure 1.6: Interception of traffic flow from source to destination by an unauthorized third party 15
- Figure 1.7: Modification of traffic flow from source to destination by an unauthorized third party 16
- Figure 1.8: Fabrication of traffic flow from unauthorized third party to destination. Third party is posing as a trusted source of information 16

Chapter Three

- Figure 3.1: Public-Key Cryptography 70

Chapter Four

Figure 4.1: main screen	85
Figure 4.2: Encryption Text By Two Primes	86
Figure 4.3: Encrypted message	86
Figure 4.4: Insert Value To The Right Side	88
Figure 4.5: Decrypt message	88
Figure 4.6: Time Encrypt Text	89
Figure 4.7: Time Decrypt Text	89
Figure 4.8: Encrypt_3_primes	90
Figure 4.9: Encrypt message	91
Figure 4.10: Insert Value To The Right Side	92
Figure 4.11: Decrypt Message	93
Figure 4.12: Time Encrypt Text	93
Figure 4.13: Time Decrypt Text	94
Figure 4.14: Graph Time Encrypt	96
Figure 4.15: Graph Time Decrypt	96
Figure 4.16: Generation and Factorization of the Key	97
Figure 4.17: Generate Key From 2 Primes	98
Figure 4.18: Time Generate Key By 2 Primes	99
Figure 4.19: Factoring N From 2 Primes	99
Figure 4.20: Time Factoring N By 2 Primes	100
Figure 4.21: Generate Key From 3 Primes	100
Figure 4.22: Time Generate Key By 3 Primes	101

Figure 4.23: Factoring N From 2 Primes	101
Figure 4.24: Time Factoring N From 3 Primes	101
Figure 4.25: Some Examples For The Factorization N From 3 Primes	102
Figure 4.26: Factoring And Time To N=35 From 2 Primes	103
Figure 4.27: Factoring And Time To N = 105 From 3 Primes	103
Figure 4.28: Time Generate Key	105
Figure 4.29: Time Generate Key	105

List of Tables

Chapter Two

Table 2.1 : Rules that can help you find the numbers to divide	42
--	----

Chapter Three

Table 3.1: Breaking RSA Algorithm By Pollard Rho Factoring Method	82
---	----

Table 3.2: Breaking RSA Enhance Algorithm By Pollard Rho Factoring Method	82
---	----

Chapter Four

Table 4.1: Encryption And Decryption By using Two Primes	95
--	----

Table 4.2: Encryption And Decryption By using Three Primes	95
--	----

Table 4.3: Generation And Factorization Key From 18 Digits By 2 primes	104
--	-----

Table 4.4: Generation And Factorization Key From 18 Digits By 3 primes	104
--	-----

Table 4.5: Comparison Between The Original Method And The Proposed Method	106
---	-----

List of Abbreviations

ASCII : American Standard Code for Information Interchange.
ATM : Automated Teller Machine .
BPI+ : Baseline Privacy Interface Plus
CPU : Central Processing Unit
CRT : Cathode Ray Tube
DES : Data Encryption Standard
DOS : Disk Operating System
ECM : Elliptic Curve Method
ECMA-334 : European Computer Manufacturers Association
GA : Genetic Algorithm
GCD : Greatest Common Divisor
GNFS : General Number Field Sieve
I/O : Input/Output
IDS : Intrusion Detection System
IEC : International Electro technical Commission
ISO : International Organization for Standardization.
NFS : Number Field Sieve
RAM : Random-Access Memory
RSA : Rivest, Shamir and Adleman
SSL : Secure Sockets Layer
TDES : Triple Data Encryption Standard
VPN : Virtual Private Network
WWI : World War I.

Useful Terminology

Much of the terminology of cryptography can be linked back to the time when only written messages were being encrypted and the same terminology is still used regardless of whether it is being applied to a written message or a stream of binary code between two computers

CODE: An unvarying rule for replacing a piece of information with another object, not necessarily of the same sort e.g. ASCII.

CRYPTANALYSIS: The science (and art) of recovering information from ciphers without knowledge of the key.

CRYPTOGRAPHY The science of the enciphering and deciphering of messages in secret code or cipher.

CRYPTOSYSTEM: A system for encrypting information.

DECRYPTION: The process of converting the CIPHER back into PLAINTEXT.

ENCRYPTION: The process of converting the PLAINTEXT into a CIPHER.

KEY: The secret information known only to the transmitter and the receiver which is used to secure the PLAINTEXT.

MONOALPHABETIC SUBSTITUTION: A method of encryption where a letter in the plaintext is always replaced by the same letter in the ciphertext.

PLAINTEXT: The source information to be secured.

POLYALPHABETIC SUBSTITUTION: A method of encryption where a letter in the plaintext is not always replaced by the same letter in the ciphertext.

RSA Enhance: New Approach RSA Algorithm

Enhanced Method for RSA Cryptosystem Algorithm

Abstract

The motive of this thesis is the desire to improve the protection of information and data on the use of encryption algorithms. Improving the specific encryption algorithm, due to the increment transmission of data and information, will protect the privacy and satisfy authentication.

In this thesis, it was proposed to enhance the RSA (which stands for Rivest, Shamir and Adleman who first publicly described it) algorithm through the use of additional third prime number in the composition of the public key and private key. This will increase the analysis complexity of the variable (N), where the process of its analysis with the development of equipments and tools become much easier nowadays.

The existence of three prime numbers will give the ability to the developed encryption method to increase the difficulty of analysis of the variable (n), In addition to increasing the speed of the process encryption and decryption. It is possible to generate a public key and private key without the need to use major calculations. To generate a variable (n) using the original RSA algorithm, which contributes to generate the public key and private key, large and have a number of 300 digits. To use two primes

numbers each of them is a number composed of 150 digits. In this case if the multiplication process has been used, it will take longer than the time of generating the same variable (n) by using the no more proposed method which uses three prime numbers where each number length 100 digits.

We have conducted experiments on a set of numbers randomly, as it proved that the Enhanced Method for RSA Cryptosystem Algorithm gone a significant results in terms of speed other than the original algorithms. In terms of encryption and decryption hand to generate the public key and private key, as well as showing that the analysis of the variable (n) takes a long time in the proposed method for algorithm RSA about the original method and this indicates the increasing difficulty in the way of analysis developed.

طريقة مطورة في خوارزمية تشفير المفتاح العام RSA

Arabic Summary

الملخص

كان دافع هذه الرسالة الرغبة في تحسين حماية المعلومات والبيانات باستخدام خوارزميات التشفير وذلك عن طريق تحسين خوارزمية تشفير معينة نظراً لتناقل البيانات والمعلومات وحماية الخصوصية وتلبية المصادقية.

في هذه الرسالة تم اقتراح تطوير خوارزمية RSA وذلك عن طريق استخدام رقم أولي ثالث في تكوين المفتاح العام والمفتاح الخاص للتحسين من صعوبة تحليل المتغير (ن) حيث ان عملية التحليل للمتغير (ن) مع تطور الاجهزة والادوات يصبح أسهل بكثير بالوقت الحالي.

تعطي طريقة التشفير المطورة المقدره على زيادة صعوبة التحليل للمتغير (ن) فضلاً عن السرعة في عملية التشفير وفك التشفير حيث انه نستطيع ان نولد مفتاح عام ومفتاح خاص دون الحاجة الى استخدام عمليات حسابية كبيرة ، لتوليد متغير (ن) بإستخدام الخوارزمية الأصلية (RSA) الذي يساهم في توليد المفتاح العام والمفتاح الخاص بشكل كبير وعلى ان يكون عبارة عن رقم من 300 خانة فإنه يجب ان نستخدم رقمين أوليين يكون كل منهما عبارة عن رقم مكون من 150 خانة. في هذه الحالة لو استخدمنا عملية الضرب سوف تأخذ وقت اطول من ان تولد المتغير (ن) بإستخدام الطريقة المقترحة التي تستخدم ثلاث اعداد اولية كل عدد يجب أن يكون عبارة عن رقم مكون من 100 خانة .

لقد اجريت التجارب على مجموعة من الارقام بشكل عشوائي حيث انه اتضح ان الطريقة المطورة في تشفير خوارزمية المفتاح العام RSA اسرع من الخوارزمية الاصلية من ناحية التشفير وفك التشفير ومن ناحية توليد المفتاح العام والمفتاح الخاص ، وكذلك تبين ان عملية التحليل للمتغير (ن) تأخذ وقت اطول في الطريقة المطورة في تشفير خوارزمية المفتاح العام RSA عن الطريقة الاصلية وهذا يدل على زيادة صعوبة التحليل في الطريقة المطورة.

Chapter One {Introduction}

1.1 Introduction

The link with the internet Imposed new security challenges for large corporate networks; the last ten years thousands of companies entered to the Internet, where you created these companies up sites on the Internet [1] .

Provided staff services, e-mail and internet browsers and bringing to the external user armed with some knowledge and some of the goals malicious way. New to infiltrate the internal regulations, as soon as the intruder inside the corporate network, he/she can wander where and destroy or alter data, or stealing, causing damage of various kinds. Even if we take more web applications use an e-mail, it is not considered safe, can anyone who has Analyst protocols and access to routers and network devices .

Other dealing with e-mail as it moves from one network to the network via the Internet to read or alter the message sent. If you do not take certain steps to ensure the integrity, acting in some companies, if the security challenges were not a real threat are looking forward to the structure infrastructure of the Internet, as a relatively cheap, to link two or several local networks geographically isolated with each other or to link with the remote network .

It should be noted that the business on the Internet, which require millions of exchanges, banking secrecy has become close to the reach of many, and respond market network security quickly for challenges of security the Internet by adopting the techniques of verification and encryption are available in this area to apply to links the Internet, and through the development of new products in the field of information security, today's markets are in chaos, standards, techniques and products [1].

The internet is currently the biggest carrier for data and information. Sometimes we have transferred the sensitive data and information (like bank information's) as encrypted forms. If we want to protect and save this information from crackers or hackers we should protect them.

The developed hardware and tangible tools are not sufficient to protect the data from unauthenticated parties and-more and more-most of encumbrance is under the software systems and controlling responsibilities. Therefore, the experts, researchers and developers have to build and develop security systems, protect the information and prevent the attackers from misuse with the very important source (information). For this reason, the term "Encryption" was brought out, and it is the main factor that should be available in protection system and take for a real process to manipulate and generate the security system. Figure (1.1) shows an overview of Cryptology [16].

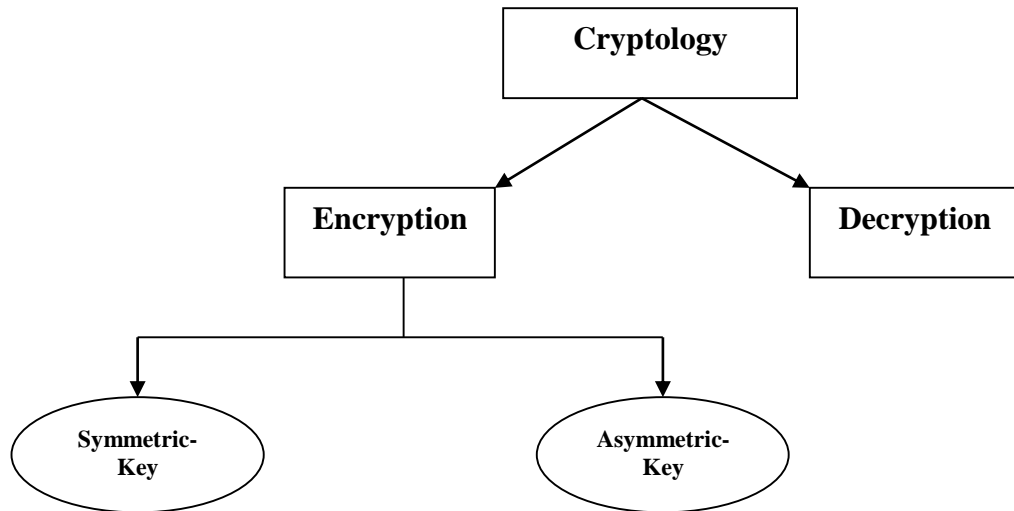


Figure 1.1: Overview on Cryptology.

1.2.Cryptology

Cryptography or cryptology {from Greek (κρυπτός), (hidden) ; (kryptos), (secret); and (γράφειν), (writing), or (λογία), (study), respectively}[12] is the practice and study of hiding information. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical 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 apparent nonsense. The sender retained the ability to decrypt the information and therefore avoid unwanted persons being able to read

it. Since WWI and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread [13].

Modern cryptography follows a strongly scientific approach, and designs cryptographic algorithms around computational hardness assumptions that are assumed hard to break by an adversary. Such systems are not unbreakable in theory but it is infeasible to do so for any practical adversary. Information-theoretically secure schemes that provably cannot be broken exist, but they are less practical than computationally-secure mechanisms. An example of such systems is the one-time pad. Alongside the advancement in cryptology-related technology, the practice has raised a number of legal issues, some of which remain unresolved [14].

Cryptographic systems are generally grouped according to three facts about them [16]:

The mathematical operation that changes the plaintext into the ciphertext using the encryption key.

Whether a block or a stream cipher is produced.

The type of key system used - single or two key.

1.3.Encryption

The Encryption is the process of transforming information, using an algorithm, to make it unreadable to anyone except those possessing special knowledge. The result of the process is encrypted information. In many contexts, Encryption process depends on transforming the plain text to cipher text [2].

The actual cryptographic process is generally a complicated mathematical formulation, the more complex -- the more difficult to break. A key is supplied to the recipient so that they can then decipher the message. Keys for encryption algorithms are described in terms of the number of bits. The higher the number of bits - the more difficult that cryptosystem would be to break.

In the science of encryption, the encryption strength of information depends on several things, including [1]:

The algorithms.

The length of key.

The time of the encryption.

Tools used in the encryption.

The algorithm is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output. An algorithm is thus a sequence of computational steps that transform the input into the output.

We can also view an algorithm as a tool for solving a well-specified computational problem. The statement of the problem specifies in general terms the desired input/output relationship. The algorithm describes a specific computational procedure for achieving that input/output relationship [3].

The key is a code used within the algorithm, so as to increase the Complexity of the encryption process, which is also used in decryption and in order to convert information encrypted to the information to be understand.

There are two categories of encryption dependently of key.

Symmetric Encryption technology.

Asymmetric Encryption technology.

1.4 Network Encryption:

Network is a collection of computers linked to each other, either through wired or wireless network. The exchange of information among themselves, and because of the Web has become almost all devices of the world are connected with each other and the world has become a very small village roaming through a computer. Although there are a lot of good people, is not it also draws the bad guys, who are trying to spy on information, take and how to dispose of what they

want, and here it was necessary to the protection of information on you making it there is something called encryption networks, which is to protect your information from hackers through a computer network.

1.5. Symmetric Encryption Technology:

Symmetric encryption is the oldest and best-known technique. A secret key, which can be a number, a word, or just a string of random letters, is applied to the text of a message to change the content in a particular way. This might be as simple as shifting each letter by a number of places in the alphabet. As long as both sender and recipient know the secret key, they can encrypt and decrypt all messages that use this key. Figure (1.2) shows the conventional Encryption [1].

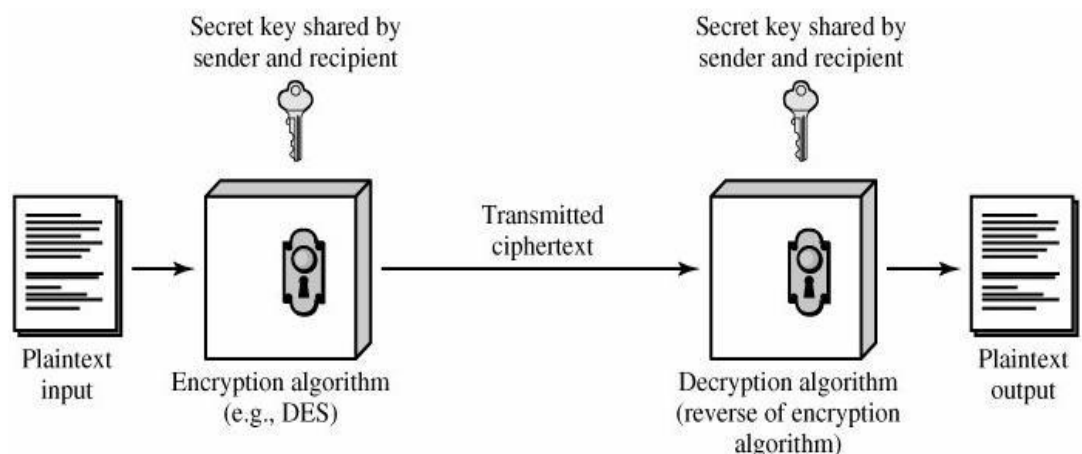


Figure 1.2: Simplified Model of Conventional Encryption

1.6. Private Key Encryption

Private Key encryption is the standard form. Both parties share an encryption key, and the encryption key is also the one used to decrypt the message. The difficulty is sharing the key before you start encrypting the message - how do you safely transmit it?

Many private key encryption methods use public key encryption to transmit the private key for each data transfer session.

If Bob and Alice want to use private key encryption to share a secret message, they would each use a copy of the same key. Bob writes his message to Alice and uses their shared private key to encrypt the message. The message is then sent to Alice. Alice uses her copy of the private key to decrypt the message. Private key encryption is like making copies of a key. Anyone with a copy can open the lock. In the case of Bob and Alice, their keys would be guarded closely because they can both encrypt and decrypt messages [18].

Asymmetric Encryption Technology:

The problem with secret keys is exchanging them over the Internet or a large network while preventing them from falling into the wrong hands. Anyone who knows the secret key can decrypt the message. One answer is asymmetric encryption, in which there are two related keys--a key pair. A public key is made freely available to anyone who might want to send you a message. A second, private key is kept secret, so that only you know it [18].

Any message (text, binary files, or documents) that are encrypted by using the public key can only be decrypted by applying the same algorithm, but by using the matching private key. Any message that is encrypted by using the private key can only be decrypted by using the matching public key.

This means that you do not have to worry about passing public keys over the Internet (the keys are supposed to be public). A problem with asymmetric encryption, however, is that it is slower than symmetric encryption. It requires far more processing power to both encrypt and decrypt the content of the message [1]. Figure (1.3) shows the asymmetric Encryption. It is considered the most powerful examples applied to the public key are RSA code [20].

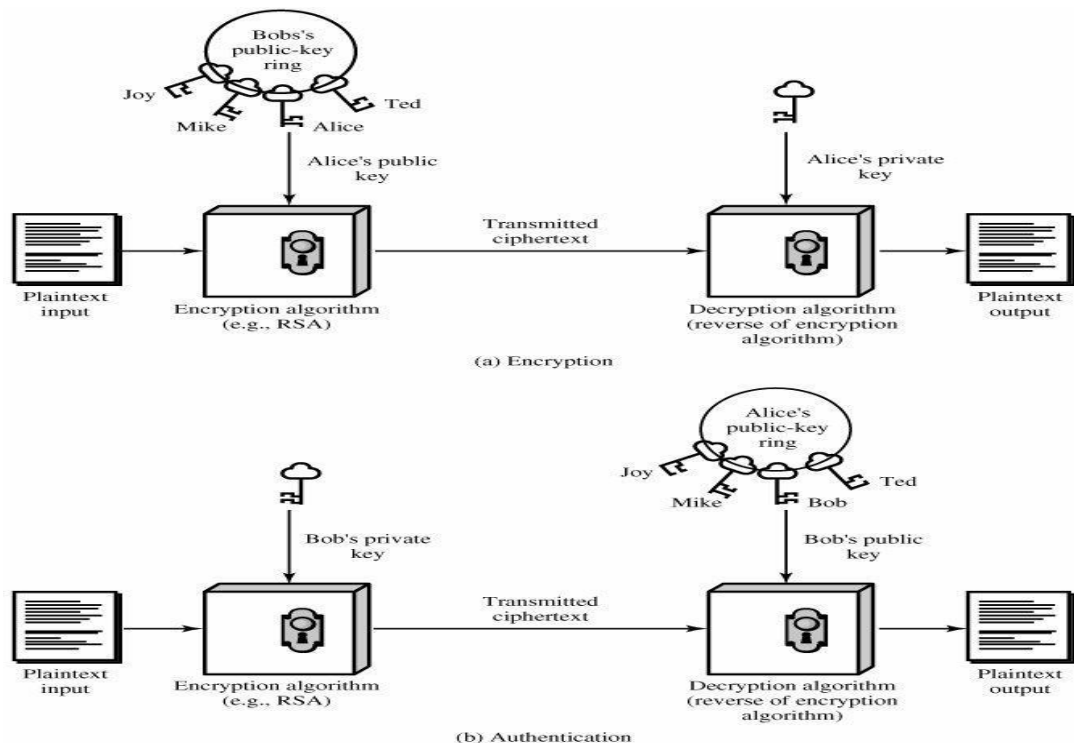


Figure 1.3: Public-Key Cryptography

1.6.1 Public Key encryption

Public key encryption uses two keys - one to encrypt, and one to decrypt the message, and sends the encrypted message to the receiver. Only the receiver can then decrypt the message - even the sender cannot read the encrypted message [19].

When Bob wants to share a secret with Alice using public key encryption, he first asks Alice for her public key. Next, Bob uses Alice's public key to encrypt the message. In public key encryption, only Alice's private key can unlock the message encrypted with her public key. Bob sends his message to Alice. Alice uses her private key to decrypt Bob's message.

The things that make public key encryption work are that Alice very closely guards her private key and freely distributes her public key. She knows that it will unlock any message encrypted with her public key [20].

1.6.2. Rivest, Shamir, and Adelman (RSA) Algorithm

The RSA Algorithm is a form of public key encryption. Public key encryption is a process where each user is given two keys, one which is public and seen by anyone who wishes to see it and one which is kept strictly private. The thought of making one of your keys public seems bizarre at first, but we will soon see why this is so effective and secure.

In public key encryption algorithms, such as RSA, each message is encrypted using your recipient's public key. You can get your

recipient's public key easily- it's public. However, only your recipient has the corresponding private key to decode it. So suppose someone intercepted your message. They would only have access to the public keys, but almost no way to actually decrypt the message. The only way would be to reverse the algorithm, which is extremely difficult [11].

It is named after its inventors: Ron Rivest, Adi Shamir, and Leonard Adleman [5], Its security comes from the difficulty of factoring large numbers, The public and private keys are functions of a pair of large prime numbers (e.g., 1024 bits) .

To generate a key pair, choose two large random prime numbers, p and q , for maximum security, p and q should be of equal length [5].

The public key code contains five components .

Plain text: it is any text or readable data that will be used with algorithms as inputs.

Encryption algorithm.

Two keys which are Public key and private key.

Cipher text: it is the output of encryption processes of original plain text by the encryption algorithm that used.

Decryption algorithm: where the cipher text will be as input paralleled with known public key to return the plain text.

1.6.3 The RSA cryptosystem

In the RSA public-key cryptosystem, a participant creates his/her public and secret keys with the following procedure [3].

Select at random two large prime numbers p and q such that $p \neq q$.

The primes p and q might be, say, 512 bits each.

Compute n by the equation $n = p * q$.

Compute $\phi(n)$ by the equation $\phi(n) = (p-1) * (q-1)$.

Select a small odd integer e that is relatively prime to $\phi(n)$, the $\text{GCD}(e, \phi(n))$ must equal 1 and $1 < e < \phi(n)$.

Compute d as the multiplicative inverse of e , modulo $\phi(n)$ by the equation $d * e = 1 \pmod{\phi(n)}$.

Publish the RSA public key $P = (e, n)$.

Keep the RSA secret key $S = (d, n)$.

The encrypted message c will be made up of message blocks c_i of about the same length [5].

The encryption formula is:

$$c_i = m_i^P \pmod n$$

To decrypt a message, take each encrypted block c_i and compute:

$$m_i = c_i^S \pmod n$$

Since

$$c_i^S = (m_i^P)^S = m_i^{k(p-1)(q-1)+1} = m_i m_i^{k(p-1)(q-1)} = m_i * 1 = m_i; \text{ all } \pmod n$$

($m_i^{k(p-1)(q-1)} \equiv 1 \pmod n$ follows from Euler's Theorem.)

This recovers the message .

1.6.4 Security Attacks

We can divide the attacks on the security networks or computer system in view of the computer to function as a source of data; we can say that there is a source of flow data, which is the sender (such as sending a file or folder) and there is also destination, or the target (receiving a file or folder) .

Normal flow – Under normal conditions traffic should pass unobstructed from source to destination as in figure (1.4). A source of data is sending that data to an intended destination and that destination is receiving this data unobstructed and without it being compromised in any way, and in a perfect world this would be always be the case.

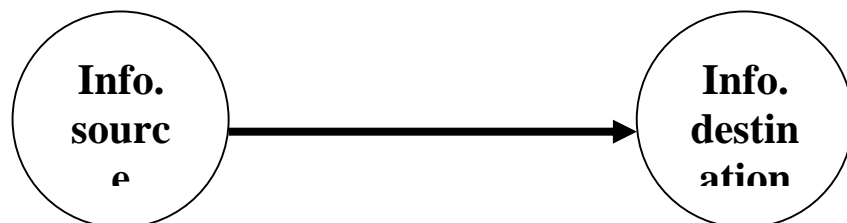


Figure 1.4: Normal Traffic Flow Pattern From Source To Destination

Mostly this flow is not going in this way because the attacks pursue data that causing the data flow to be abnormal. There are four types of attacks .

A) Interruption – The first type of attack and among the simplest would be total packet flow interruption.

This category can include lots of different types of network attacks such as physical, discussed previously, which could be carried out by a person physically turning off the router or otherwise disabling it. The interruption of service may also be caused by an intruder gaining unrestricted access by means of telnet or some type of out of band management and commanding the routers interfaces to shut down thereby interrupting traffic. A Denial of Service (DoS) attack is an example of an attack where its purpose is the interruption of information as shown in Figure (1.5).

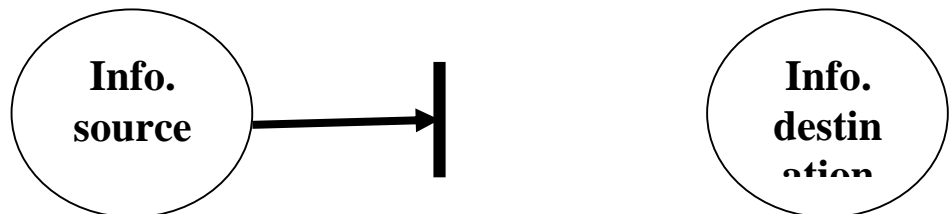


Figure 1.5: Traffic Interruption From Source To Destination By An Attack

B) Interception - A common type of attack, this attack is performed by snooping on network traffic to try to obtain data such as passwords, credit card numbers, or other types of sensitive information that may be transmitted in clear text. A Man in the Middle attack is an example of this category. The industry has developed many ways to try to protecting the hijacking of information in this way. Encryption means such as SSL, VPN, 3DES, BPI+ are deployed to encrypts the flow of information from source to destination so that if someone is able to snoop in on the flow of traffic, all the person will see is ciphered text

. The use of “strong” encryption is always preferable since even though the text is encrypted the intruder does have the ability to capture and save this information and try to decrypt it passively. Some encryption methods are more easily broken than others so as the data being sent becomes more sensitive in nature, more care will need to be taken to protect that data from this type of intrusion as shown in Figure (1.6).

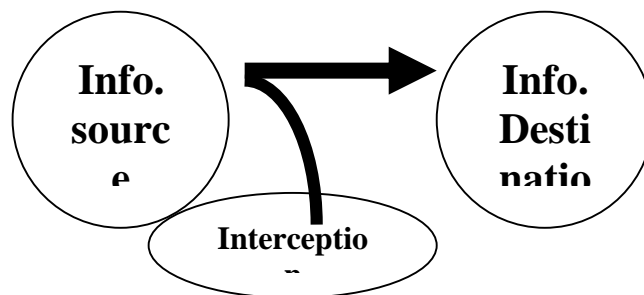


Figure 1.6: Interception Of Traffic Flow From Source To Destination By An Unauthorized Third Party.

C) Modification – Port Redirection would be a way for this type of attack to occur. In this case an attacker has been able to force traffic to flow from source to destination un-detected through a 3rd party host to be able to modify or falsify data as it passes through. If done properly and if no detection methods are in place both the source and destination will have no way of knowing the traffic is being altered. An intruder performing this kind of attack could take incoming data from the source and attach viruses or other malicious code and send it along its way undetected to the destination. Mitigation techniques for

this type of attack could be the introduction of intrusion detection systems (IDS) which could look for different signatures which represent an attack. In the case of this example, the IDS may spot an unusual spike in latency for data to reach the destination. This increase in time would be from the intruder redirecting traffic and then altering traffic as shown in Figure (1.7).

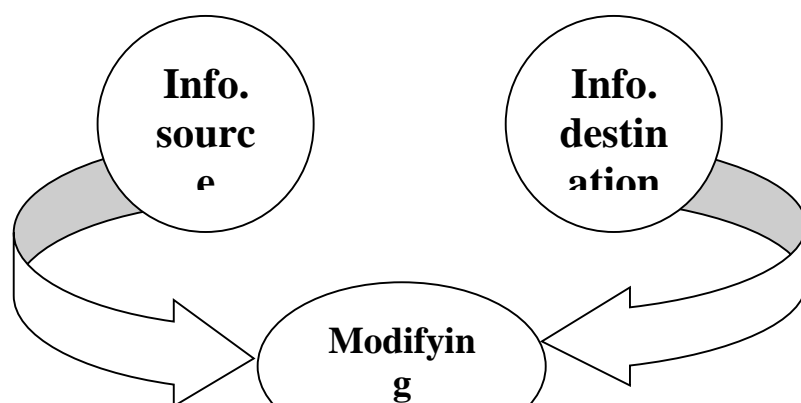


Figure 1.7: Modification Of traffic Flow From Source To Destination By An Unauthorized Third Party.

D) Fabrication – Much like Modification, Fabrication incorporates a 3rd party host sending data to the original destination of traffic. In this case the 3rd party could be spoofing the credentials of the original source thereby appearing to the end destination to be a valid and trusted source of information. This would be a more intrusive way to perform an attack if someone is monitoring the original source. In this case the original source is no longer sending or receiving traffic and may raise suspicions as shown in Figure (1.8).

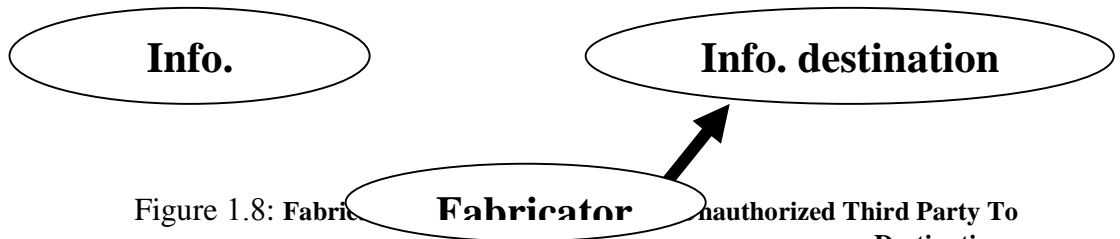


Figure 1.8: Fabricator Authorized Third Party To Destination.
Third Party Is Posing As A Trusted Source Of Information

1.6.5. Attacks on RSA

The saying "A chain is no stronger than its weakest link" is a very suitable for describing attacks on cryptosystems. The attackers' instinct is to go for the weakest point of defense, and to exploit it. Sometimes the weakness may have appeared insignificant to the designer of the system, or maybe the cryptanalyst will discover something that was not seen by anyone before. The important thing to remember (and this has been proven time and time again in the history of cryptography) is that no matter how secure you think your system is, there may be something you have not considered.

At the moment RSA seems to be extremely secure. It has survived over 20 years of scrutiny and is in widespread use throughout the world. The attack that is most often considered for RSA is the factoring of the public key. If this can be achieved, all messages written with the public key can be decrypted. The point is that with very large numbers, factoring takes an unreasonable amount of time (see the factorization section for more details of the difficulty). It has

not been proven that breaking the RSA algorithm is equivalent to factoring large numbers (there may be another, easier method), but neither has it been proven that factoring is not equivalent.

I mentioned before that a chain is only as strong as its weakest link. In cryptology terms, the links in the chain include key generation, key management, the cryptographic algorithm and the cryptographic protocol. If there is a weakness in any one of these areas, it undermines the entire system. Imagine an eavesdropper was able to generate session keys in the same order that an e-commerce site web server used to get credit card details securely from customers over the Internet; this would allow the eavesdropper to read all the transactions. The section on random number generators discusses this topic.

It's now time to get into the details of attacks on RSA.

Searching the Message Space

One of the seeming weaknesses of public key cryptography is that one has to give away to everybody the algorithm that encrypts the data. If the message space is small, then one could simply try to encrypt every possible message block, until a match is found with one of the ciphertext blocks. In practice this would be an insurmountable task because the block sizes are quite large.

Guessing d

Another possible attack is a known ciphertext attack. This time the attacker knows both the plaintext and ciphertext (they simply have to encrypt something). They then try to crack the key to discover the private exponent, d . This might involve trying every possible key in the system on the ciphertext until it returns to the original plaintext. Once d has been discovered it is easy to find the factors of n (for example use the algorithm in chapter 8 of The Handbook of Applied Cryptography). Then the system has been broken completely and all further ciphertexts can be decrypted.

The problem with this attack is that it is slow. There are an enormous number of possible d s to try. This method is a factorizing algorithm as it allows us to factor n . Since factorizing is an intractable problem we know this is very difficult. This method is not the fastest way to factorize n . Therefore one is suggested to focus effort into using a more efficient algorithm specifically designed to factor n . This advice was given in the original paper.

Cycle Attack

This attack is very similar to the last. The idea is that we encrypt the ciphertext repeatedly, counting the iterations, until the original text appears. This number of re-cycles will decrypt any ciphertext. Again this method is very slow and for a large key it is not a practical attack. A generalization of the attack allows the modulus to be factored and

it works faster the majority of the time. But even this will still have difficulty when a large key is used. Also the use of p -strong primes aids the security.

The bottom line is that the generalized form of the cipher attack is another factoring algorithm. It is not efficient, and therefore the attack is not good enough compared with modern factoring algorithms (e.g. Number Field Sieve).

I noticed an improvement on this algorithm. The suggested way is to use the public exponent of the public key to re-encrypt the text. However any exponent should work so long as it is prime to $(p-1).(q-1)$ (where p, q are factors of the modulus). So I suggest using an exponent such as $2^{16} + 1$. This number has only two 1s in its binary representation. Using binary fast exponentiation, we use only 16 modular squaring and 1 modular multiplication. This is likely to be faster than the actual public exponent. The trouble is that we cannot be sure that it is prime to $(p-1).(q-1)$. In practice, many RSA systems use $2^{16} + 1$ as the encrypting exponent for its speed.

Common Modulus

One of the early weaknesses found was in a system of RSA where the users within an organization would share the public modulus. That is to say, the administration would choose the public modulus securely and generate pairs of encryption and decryption exponents (public and private keys) and distribute them all the employees/users. The reason for doing this is to make it convenient to manage and to write software for.

However, Simmons shows how this would allow any eavesdropper to view any messages encrypted with two keys; for example when a memo is sent to several employees. DeLaurentis went further to demonstrate how the system was at even more risk from insiders, who could break the system completely, allowing them to view all messages and sign with anybody's key.

Faulty Encryption

Joye and Quisquater showed how to capitalize on the common modulus weakness due to a transient error when transmitting the public key. Consider the situation where an attacker, Malory, has access to the communication channel used by Alice and Bob. In other words, Malory can listen to anything that is transmitted, and can also change what is transmitted. Alice wishes to talk privately to Bob, but does not know his public key. She requests by sending an email, to which Bob replies. But during transmission, Malory is able to see the public key and decides to flip a single bit in the public exponent of Bob, changing (e,n) to (e',n) .

When Alice receives the faulty key, she encrypts the prepared message and sends it to Bob (Malory also gets it). But of course, Bob cannot decrypt it because the wrong key was used. So he lets Alice know and they agree to try again, starting with Bob re-sending his public key. This time Malory does not interfere. Alice sends the message again, this time encrypted with the correct public key.

Malory now has two ciphertexts, one encrypted with the faulty exponent and one with the correct one. She also knows both these exponents and the public modulus. Therefore she can now apply the common modulus attack to retrieve Alice's message, assuming that Alice was foolish enough to encrypt exactly the same message the second time.

A demonstration of the Common Modulus attack and the Faulty Encryption attack can be found in this Mathematical notebook.

Low Exponent

In the cycle attack section above, I suggested that the encrypting exponent could be chosen to make the system more efficient. Many RSA systems use $e=3$ to make encrypting faster. However, there is vulnerability with this attack. If the same message is encrypted 3 times with different keys (that is same exponent, different module) then we can retrieve the message. The attack is based on the Chinese Remainder Theorem. See The Handbook of Applied Cryptography for an explanation and algorithm.

Factoring the Public Key

Factoring the public key is seen as the best way to go about cracking RSA.

1.6.6 Cryptography Strength of RSA

Encryption algorithms rely on strength in certain matters. Each algorithm has properties of the mismatch depending strongly on

the encryption key, which depends on the arrangement of, and denounced it tries to integrate the key with the text of certain deviation.

The algorithm RSA based on the variable N consisting of multiplying each of the P and Q , which are relying on that of where to find the variable d , as the variable d is, hence the higher value of n . The variable d increases its size, the higher value of p and q the value of d increases, which means that the algorithm depends entirely on the adoption of the primes numbers because they generate a key d , depending on p and q are already primes numbers [21] .

1.6.7 Weaknesses in RSA

These weaknesses RSA algorithm when we use two primes number and the following points are used to break the algorithm in most cases.

A) Small encryption exponent

If you use a small exponent like $e=3$ and send the same message to different recipients and just use the RSA algorithm without adding random padding to the message, then an eavesdropper could recover the plaintext.

B) Using the same key for encryption and signing

Given that the underlying mathematics is the same for encryption and signing, only in reverse, if an attacker can convince a key holder to

sign an unformatted encrypted message using the same key then she gets the original.

c) Acting as an oracle

There are techniques to recover the plaintext if a user just blindly returns the RSA transformation of the input. So don't do that.

1. 7 Statement of Problem:

The purpose of this thesis is to provide new approach to the theory of RSA. The proposed method is to use initial three prime's numbers instead of two in the original algorithm to find the variables, the public key and private key. Then there will be a comparison between the original and the new algorithm in terms of:

Strength.

Execution time.

Break the public key or private key.

The problem we have here is the ability to proof theory and proof strength of the algorithm, execution time of implementation and the implementation mechanism. We will try to proof the key strengths through using three prime numbers respect to the key strengths of using two prime numbers.

It is well known that the algorithm RSA using two primes numbers to generate a public key and private key and depend on the length of the variable N in terms of the length the variable and using the primes numbers are too large to work the value of N is too long to, So as not

to break through the method of analysis of the factors, and with the progress and development of technology, the equipment can analyze the variable N , and solve this problem, the experts and developers to increase the length of the variable N and through the use of primes numbers larger.

1.8. Contribution

Classically, an RSA modulus has been composed from two primes. However, there are very practical reasons why using more than two primes might be preferred.

The primes are smaller and key generation takes less time despite there being more of them.

There are two possible methods for attacking an RSA key if it is built from more than two primes. The first is Number Field Sieve (NFS), which has already been discussed. The second method is the Elliptic Curve Method (ECM) [6].

We will discuss in this thesis all the things that had previously reported. We will analyze the algorithm we have imposed and compared with the original algorithm for RSA in terms of:

Running time.

Analysis attack algorithm.

We will also strengthen the comparisons through a computer program compares the encryption of the original algorithm and the algorithm proposed, as well as we will rely on equations to prove the analysis and comparison of the algorithm.

The main specifications of machine that apply the proposed method have to be displayed and considered like the CPU speed capacity, because the main specifications play an important role to manipulations and calculations speed, where everybody know how significant time it. So, the workspace machine specification is:

CPU= E7400 2.8GHz

RAM: 3 GB (system memory).

The process of analyzing the number n usually be convert to figure out binary system and then divided by digits and therefore do some calculations to find one prime numbers that constituent the key and noted that the number n in the way currently consists two-digit number so if we find one of these numbers, the second will be known to have implied, In the new method, the process of analyzing the number n be the hardest so that if we found one of the prime number , it remains two numbers is unknown here that the new hard way.

1.9. Thesis Organization

After the discussion of what is the security system, what is the significant of these systems, the risks that surrounded them, and the phases of development of these systems, we explicated the main topics that is related to cryptosystems in general, and too close to RSA cryptosystem, the factors, facts and specifications of RSA cryptosystem and its algorithm. The last session was for main emerged challenges and problems when somebody attempts to enhance the RSA cryptosystems. These topics discussed in chapter one.

In chapter two, some related works that describe recent works will be given to give an insight to the latest research advancements systems related to advance reservation attempts to enhance and break the RSA cryptosystem.

In the chapter three, the new design will be discussed and the custom to demonstrate the benefits of adding the third prime number will be explained. We have added on an algorithm which landed on the one hand to generate the available public and private key, as well as in terms of its impact on the variable n , and by the complex which will add third variable to the algorithm. We will process the breaking of the original algorithm and the proposed new small examples in order to demonstrate the complexity of both methods.

In the chapter four, we will develop a practical application in order to implement the algorithms the original and the new proposal and through this practice will judge any methods more effective in all respects, whether in terms of the speed factor or complex, or a security standpoint and we will work table summarizes all the differences between the original method and proposed method.

Finally, in chapter five, the contribution of the proposed method will be reviewed; the achievements of proposed method to what extent this method has succeeded in completing the process of enhance the algorithm. Some future works will be suggested to improve the proposed solution or to attract the attention to the weakness points of RSA algorithm.

Chapter two {Theory & Related Works }

2.1.Introduction

When they are producing a new security system, all researchers who are interested in the field of security they have to analyze the system and to find its strengths and weaknesses. When it is to find weaknesses, researchers to work on, scholars Good work to take the weaknesses and improve them to get them to regulations that are more power and safety. Through the results that they come out, the researchers bad take the weaknesses of harm to others.

The problem with an algorithm (such as RSA) on the process of analysis is the primary factors. As all operations to break the algorithm based on factorizing method; there are few possible interpretations of breaking RSA. The most damaging would be for an attacker to discover the private key corresponding to a given public key and forge signatures. The obvious way to carry out this attack is to factor the public modulus, n , into its two prime factors; p and q . From p , q and E , the public exponent, the attacker can easily calculate d , the private exponent. The hard part is factoring n ; the security of RSA depends on factoring being difficult. In fact, the task of recovering the private key is equivalent to the task of factoring the modulus; you can use d to factor n , as well as use the factorization of n to find d .

2.2. Two Categories of Attacks On RSA

There is a straight method, to enumerate all elements in the multiplicative group of N until M is found, but such method results in an exponential running time, $O(n^e)$. Therefore, Prefer very closely efficiency of the algorithm with a substantial lower running time. During the past years of attacking on RSA, such efficient algorithms can be classified into two categories: Mathematical Attacks and Implementation Attacks.

2.2.1. Mathematical Attacks on RSA

Mathematical attacks focus on attacking the underlying structure of RSA function. The first intuitive attack is the attempt to factor the modulus N . Because knowing the factorization of N , one may easily obtain $\Theta(N)$, from which d can be determined by $d = 1/e \pmod{\Theta(N)}$. However, at present, the fastest factoring algorithm runs in exponential time. Our objective is to survey RSA attacks that decrypts message without directly factoring N [30].

Elementary attacks

Generally speaking, Elementary attacks revealed blatant misuse of RSA. One common example of such misuse would be choosing common modulus N to serve multiple users. Let's assume the same N is used by all users, and Alice is sending a message M to Bob, which has been encrypted by the RSA function, $C = M^{(eb)} \pmod N$. It looks like Marvin can not decrypt C since he does not know db [22].

However, in fact, Marvin is able to use his own keys, e_m and d_m , to factor n , and in turn recover Bob's private key, d_b . So the resulting system is no longer secure [22].

Common Modulus

To avoid generating a different modulus $n = pq$ for each user one may wish to fix n once and for all. The same n is used by all users. A trusted central authority could provide user i with a unique pair e_i, d_i from which user i forms a public key (n, e_i) and a secret key (n, d_i) [30]. At first glance this may seem to work: a ciphertext $C = M e_a \pmod N$ intended for Alice cannot be decrypted by Bob since Bob does not possess d_a . However, this is incorrect and the resulting system is insecure. By Fact 1 Bob can use his own exponents e_b, d_b to factor the modulus N . Once N is factored Bob can recover Alice's private key d_a from her public key e_a . This observation, due to Simmons, shows that an RSA modulus should never be used by more than one entity [24].

Blinding

Let (n, d) be Bob's private key and (n, e) be his corresponding public key. Suppose an adversary Marvin wants Bob's signature on a message $M \in \mathbb{Z}^*_N$. Being no fool, Bob refuses to sign M . Marvin can try the following: he picks a random $r \in \mathbb{Z}^*_N$ and sets $M' = r e M \pmod N$. He then asks Bob to sign the random message M' . Bob may be

willing to provide his signature S' on the innocent-looking M' . But recall that $S' = (M')^d \pmod N$. Marvin now simply computes $S = S' / r \pmod N$ and obtains Bob's signature S on the original M . Indeed [30],

$$S^e = (S')^e / r^e = (M')^{ed} / r^e \equiv M' / r^e = M \pmod n$$

This technique, called blinding, enables Marvin to obtain a valid signature on a message of his choice by asking Bob to sign a random "blinded" message. Bob has no information as to what message he is actually signing. Since most signature schemes apply a "one-way hash" to the message M prior to signing [25], the attack is not a serious concern. Although we presented blinding as an attack, it is actually a useful property of RSA needed for implementing anonymous digital cash (cash that can be used to purchase goods, but does not reveal the identity of the person making the purchase) [30].

Attacks on the actual RSA cryptosystem

Small Private Key attacks

To improve the RSA decryption performance in the matter of running-time, Alice might tend to use a small value of d , rather than a large random number. A small private key indeed will improve performance dramatically, but unfortunately, an attack posed by M. Wiener [26] shows that a small d leads to a total collapse of RSA cryptosystem. This break of RSA is based on Wiener's Theorem, which in general provides a lower constraint for d . Wiener has proved that Marvin may efficiently find d when $d < 1/3 * N^{1/4}$ [22].

In addition to his success in RSA-attack, Wiener also discovered a number of techniques that enable fast decryption and are not susceptible to his attack. Two sample techniques are illustrated as the following [30].

Choosing large public key: Replacing e by e' , where $e' = e + t * \Theta(N)$ for some large t . When e' is sufficient large, i.e. $e' > N^{0.5}$, then Wiener's attack can not be mounted regardless of how small d is [22].

Using Chinese Remainder Theorem: Suppose one chooses d such that both $dp = d \bmod (p - 1)$ and $dq = d \bmod (q - 1)$ are small, then a fast decryption of C can be carried out as follows: first compute $Mp = C^{dp} \bmod p$ and $Mq = C^{dq} \bmod q$. Then use the CRT to compute the unique value $M \in \mathbb{Z}_N$ satisfying $M = Mp \bmod p$ and $M = Mq \bmod q$. The resulting M satisfies $M = C^d \bmod N$ as required. The point is that the attack by Wiener's Theorem does not apply here because the value of $d \bmod \Theta(N)$ can be large [30].

Small Public Key Attacks

Similar to the private key preferences, to reduce encryption time, it is customary to use a small public key (e), but unlike the previous situation, attacks on small e turn out to be much less effective. The most powerful attacks on small e are based on Coppersmith's Theorem [24]. This theorem provides an algorithm for efficiently finding all roots of N that are less than $x = N^{1/d}$. For brevity reason

, we will bypass the details of Coppersmith's Theorem; rather focus on its impact. One example of applications based on this theorem is known as "Hastad's Broadcast Attack" [27].

** Hastad's Broadcast Attack

Suppose Bob wishes to send an encrypted message M to a number of parties $P_1; P_2; \dots; P_k$. Each party has its own RSA key, $\langle N_i, e_i \rangle$. Hastad showed that a linear-padding to M prior to encryption is insecure, and further more, by eavesdropping Marvin learns $C_i = f_i(M)^{e_i} \bmod N_i$ for $i = 1..k$, if enough parties are involved, Marvin can recover the plaintext M from all the ciphertext [27]. His discovery stands on the mathematical analysis on solving system of equations: $g_i(M) = 0 \bmod N_i$ (1). He proved that a system of univariate equations modulo relatively prime composites, such as (1), could be efficiently solved if sufficiently many such equations are provided [22].

2.2.2. Implementation Attacks on RSA

Securely implementing RSA is not a trivial task. Attacks falling into this category take on the implementation pitfalls of RSA cryptosystems. A clever attack posed by Kocher, known as "Timing Attacks" [23], is a typical example of attacks on the RSA implementation.

Suppose a smartcard that stores a private RSA key is used, and Marvin may not be able to examine its contents and expose the key. However, by precisely measuring the time it takes the smartcard to perform an RSA decryption, Marvin can quickly discover the private decryption exponent d . This is referred to as “Timing Attacks” [22].

Timing Attacks

Consider a smartcard that stores a private RSA key. Since the card is tamper resistant, an attacker Marvin may not be able to examine its contents and expose the key. However, a clever attack due to Kocher [28] shows that by precisely measuring the time it takes the smartcard to perform an RSA decryption (or signature), Marvin can quickly discover the private decryption exponent d [24].

We explain how to mount the attack against a simple implementation of RSA using the

“Repeated squaring algorithm”. Let $d = d_n d_{n-1} \dots d_0$ be the binary representation of d (i.e., $d = (\sum_{i=0}^n 2^i d_i)$ with $d_i \in \{0; 1\}$). The repeated squaring algorithm computes $C = Md \pmod N$, using at most $2n$ modular multiplications. It is based on the observation that $C = \prod_{i=0}^n M^{2^i d_i} \pmod N$ [24].

The algorithm works as follows:

Set z equal to M and C equal to 1. For $i = 0, \dots, n$, do these steps:

(a) If $d_i = 1$ set C equal to $C * z \bmod N$,

(b) Set z equal to $z^2 \bmod N$.

At the end, C has the value $M^d \bmod N$ [24].

The variable z runs through the set of values $M^{2^i} \bmod N$ for $i = 0, \dots, n$. The variable C "collects" the appropriate powers in the set to obtain $M^d \bmod N$.

To mount the attack, Marvin asks the smartcard to generate signatures on a large number of random messages $M_1, \dots, M_k \in \mathbb{Z}_N^*$ and measures the time T_i it takes the card to generate each of the signatures [24].

The attack recovers bits of d one at a time beginning with the least significant bit. We know d is odd. Thus $d_0 = 1$. Consider the second iteration. Initially $z = M^2 \bmod N$ and $C = M$. If $d_1 = 1$, the smartcard computes the product $C * z = M * M^2 \bmod N$. Otherwise, it does not. Let t_i be the time it takes the smartcard to compute $M_i * M_i^2 \bmod N$. The t_i 's differ from each other since the time to compute $M_i * M_i^2 \bmod N$ depends on the value of M_i (simple modular reduction algorithms take a different amount of time depending on the value being reduced). Marvin measures the t_i 's offline (prior to mounting the attack) once he obtains the physical specifications of the card [24].

Kocher observed that when $d_1 = 1$, the two ensembles (t_i) and (T_i) are correlated. For instance, if, for some i , t_i is much larger than its expectation, then T_i is also likely to be larger than its expectation. On the other hand, if $d_1 = 0$, the two ensembles (t_i) and (T_i) behave as independent random variables. By measuring the correlation, Marvin can determine whether d_1 is 0 or 1 [24].

Continuing in this way, he can recover d_2, d_3 , and so on. Note that when a low public exponent e is used, the partial key exposure attack of the previous section shows that Kocher's timing attack needs only to be employed until a quarter of the bits of d are discovered [24].

There are two ways to defend against the attack. The simplest may be to add appropriate delay so that modular exponentiation always takes a fixed amount of time. The second approach, due to Rivest, is based on blinding. Prior to decryption of M the smartcard picks a random $r \in \mathbb{Z}_N^*$ and computes $M' = M * r^e \text{ mod } N$. It then applies d to M' and obtains $C' = (M')^d \text{ mod } N$. Finally, the smartcard sets $C = C'/r \text{ mod } N$. With this approach, the smartcard is applying d to a random message M' unknown to Marvin. As a result, Marvin cannot mount the attack [24].

Kocher recently discovered another attack along these lines called *power cryptanalysis*. Kocher showed that by precisely measuring the smartcard's *power consumption* during signature generation, Marvin can often easily discover the secret key. As it turns out, during a multi-precision multiplication the card's power consumption is higher than normal. By measuring the length of high consumption periods, Marvin can easily determine if in a given iteration the card performs one or two multiplications, thus exposing the bits of d [24].

Random Faults

Implementations of RSA decryption and signatures frequently use the Chinese Remainder Theorem to speed up the computation of $M^d \bmod N$. Instead of working modulo N , Bob first computes the signatures modulo p and q and then combines the results using the *Chinese Remainder Theorem*. More precisely, Bob first computes

$$C_p = M^{d_p} \bmod p \text{ and } C_q = M^{d_q} \bmod q$$

where $d_p = d \bmod (p - 1)$ and $d_q = d \bmod (q - 1)$. He then obtains the signature C by setting

$$C = T_1 C_p + T_2 C_q \bmod N$$

Where:

$$T1 = \left\{ \begin{array}{l} 1 \text{ mod } p \\ 0 \text{ mod } q \end{array} \right\} \quad \text{And} \quad T2 = \left\{ \begin{array}{l} 0 \text{ mod } p \\ 1 \text{ mod } q \end{array} \right\}$$

The running time of the last CRT step is negligible compared to the two exponentiations. Note that p and q are half the length of N . Since simple implementations of multiplication take quadratic time, multiplication modulo p is four times faster than modulo N . Furthermore, d_p is half the length of d and consequently computing $M^{d_p} \text{ mod } p$ is eight times faster than computing $M^d \text{ mod } N$. Overall signature time is thus reduced by a factor of four. Many implementations use this method to improve performance [24].

Boneh, DeMillo, and Lipton [29] observed that there is an inherent danger in using the CRT method. Suppose that while generating a signature, a glitch on Bob's computer causes it to miscalculate in a single instruction. Say, while copying a register from one location to another, one of the bits is flipped. (*A glitch may be caused by ambient electromagnetic interference or perhaps by a rare hardware bug, like the one found in an early version of the Pentium chip.*) Given an invalid signature, an adversary Marvin can easily factor Bob's modulus N [24].

We present a version of the attack as described by A. K. Lenstra. Suppose a single error occurs while Bob is generating a signature. As a result, exactly one of C_p or C_q will be computed incorrectly. Say C_p is correct, but C_q is not. The resulting signature is $C' = T_1 C_p + T_2 C'_q$. Once Marvin receives C' , he knows it is a false signature since $C'^e \neq M \pmod N$. However, notice that $C'^e = M \pmod p$ while $C'^e \neq M \pmod q$

As a result, $\gcd(N, C'^e - M)$ exposes a nontrivial factor of N [24].

For the attack to work Marvin must have full knowledge of M , namely we are assuming Bob does not use any random padding procedure. Random padding prior used to signing defeats the attack. A simpler defense is for Bob to check the generated signature before sending it out to the world. Checking is especially important when using the CRT speedup method. Random faults are hazardous to many cryptographic systems. Many systems, including a non-CRT implementation of RSA, can be attacked using random faults [29]. However, these results are far more theoretical [24].

2.3. Integer Factorization

"Factors" are the numbers you multiply to get another number. For instance, the factors of 15 are 3 and 5, because $3 \times 5 = 15$. Some numbers have more than one factorization (more than one way of being factored). For instance, 12 can be factored as 1×12 ,

2×6 , or 3×4 . A number that can only be factored as 1 times it self is called "prime". The first few primes are 2, 3, 5, 7, 11, and 13. The number 1 is not regarded as a prime, and is usually not included in factorizations, because 1 goes into everything. (The number 1 is a bit boring in this context, so it gets ignored) [31].

You most often want to find the "prime factorization" of a number: the list of all the prime-number factors of a given number. The prime factorization does not include 1, but does include every copy of every prime factor. For instance, the prime factorization of 8 is $2 \times 2 \times 2$, not just "2". Yes, 2 is the only factor, but you need three copies of it to multiply back to 8, so the prime factorization includes all three copies [32].

On the other hand, the prime factorization includes only the prime factors, not any products of those factors. For instance, even though $2 \times 2 = 4$ and even though 4 is a divisor of 8, 4 is NOT in the PRIME factorization of 8. That is because 8 does not equal $4!$, It's equal $2 \times 2 \times 2$, This accidental over-duplication of factors is another reason why the prime factorization is often best: it avoids counting any factor too many times. Suppose that you need to find the prime factorization of 24. Sometimes a student will just list all the divisors of 24: 1, 2, 3, 4, 6, 8, 12, and 24. Then the student will do something like make the product of all these divisors: $1 \times 2 \times 3 \times 4 \times 6 \times 8 \times 12 \times 24$. But this equals 331776, not 24. So it's best to stick to the prime factorization, even if the problem doesn't require it, in order to avoid

either omitting a factor or else over-duplicating one [32].

In the case of 24, you can find the prime factorization by taking 24 and dividing it by the smallest prime number that goes into 24: $24 \div 2 = 12$. (Actually, the "smallest" part is not as important as the "prime" part; the "smallest" part is mostly to make your work easier, because dividing by smaller numbers is simpler.) Now divide out the smallest number that goes into 12: $12 \div 2 = 6$. Now divide out the smallest number that goes into 6: $6 \div 2 = 3$. Since 3 is a prime, you're done factoring, and the prime factorization is $2 \times 2 \times 2 \times 3$ [31].

By the way, there are some divisibility rules that can help you find the numbers to divide by. There are many divisibility rules, but the simplest to use are these [31]:

num	Table 2.1 : Rules That Can Help You Find The Numbers To Divide divisibility rules
2 :	If the last digit is even, the number is divisible by 2.
3 :	If the sum of the digits is divisible by 3, the number is also divisible.
4 :	If the last two digits form a number divisible by 4, the number is also divisible.

5 :	If the last digit is a 5 or a 0, the number is divisible by 5.
6 :	If the number is divisible by both 3 and 2, it is also divisible by 6.
7 :	Take the last digit, double it, and subtract it from the rest of the number; if the answer is divisible by 7 (including 0), then the number is also.
8 :	If the last three digits form a number divisible by 8, then so is the whole number.
9 :	If the sum of the digits is divisible by 9, the number is also.
10:	If the number ends in 0, it is divisible by 10.
11:	Alternately add and subtract the digits from left to right. (You can think of the first digit as being 'added' to zero.) If the result (including 0) is divisible by 11, Example: to see whether 365167484 is divisible by 11, start by subtracting: $[0+]3-6+5-1+6-7+4-8+4 = 0$; therefore 365167484 is divisible by 11.

12:	If the number is divisible by both 3 and 4, it is also divisible by 12.
13:	Delete the last digit from the number, then subtract 9 times the deleted digit from the remaining number. If what is left is divisible by 13, then so is the original number.

These rules are effective in the process of factorizing the numbers, but the numbers are relatively small, any of the numbers consists of 8 digits on the upper limit, either large numbers, the process of factorization have different theories, such as.

2.3.1. Quadratic Sieve.

General Number Field Sieve.

Quadratic Sieve

Quadratic sieve is an “efficient” algorithm for modern *integer factorization* up to about 100 decimal digits [33], the second fastest method known (after the *general number field sieve*), and is considerably simpler than the number field sieve. It is a general purpose factorization algorithm, meaning that its running time depends solely on the size of the integer to be factored, and not on special structure or properties [33].

2.3.2. General Number Field Sieve

In number theory, the *general number field sieve (GNFS)* is the most efficient classical algorithm known for factoring integers larger than 100 digits. Heuristically, its complexity for factoring an integer n (consisting of $\log n$ bits) is of the form [35].

$$\exp \left(\left(\sqrt[3]{\frac{64}{9}} + o(1) \right) (\log n)^{\frac{1}{3}} (\log \log n)^{\frac{2}{3}} \right) = L_n \left[\frac{1}{3}, \sqrt[3]{\frac{64}{9}} \right]$$

(in L-notation) [34]. It is a generalization of the *special number field sieve*: while the latter can only factor numbers of a certain special form, the general number field sieve can factor any number apart from prime powers (which are trivial to factor by taking roots). When the term *number field sieve (NFS)* is used without qualification, it refers to the general number field sieve [35].

The principle of the number field sieve (both special and general) can be understood as an improvement to the simpler rational sieve or quadratic sieve. When using such algorithms to factor a large number n , it is necessary to search for smooth numbers (i.e. numbers with small prime factors) of order $n^{1/2}$. The size of these values is exponential in the size of n (see below). The general number field sieve, on the other hand, manages to search for smooth numbers that are sub exponential in the size of n [35].

Since these numbers are smaller, they are more likely to be smooth than the numbers inspected in previous algorithms. This is the key to the efficiency of the number field sieve. In order to achieve this speed-up, the number field sieve has to perform computations and factorizations in number fields. This results in many rather complicated aspects of the algorithm, as compared to the simpler rational sieve [35].

Note: - "B - Smooth number"

A positive integer is called B-smooth if none of its prime factors are greater than B. For example, 1620 has prime factorization $2^2 \times 3^4 \times 5$; therefore 1620 is 5-smooth because none of its prime factors are greater than 5. 5-smooth numbers are also called regular numbers or Hamming numbers, 7-smooth numbers are sometimes called highly composite [36].

** Hamming numbers: are also known as ugly numbers and also 5-smooth numbers (numbers whose prime divisors are less or equal to 5).

that B does not have to be a prime factor. If the largest prime factor of a number is p then the number is B-smooth for any $B \geq p$. Usually B is given as a prime, but composite numbers work as well. A number is B-smooth if and only if it is p -smooth, where p is the largest prime less than or equal to B [36].

"Power smooth numbers"

Further, m is called B -power smooth if all prime powers dividing m satisfy:

For example, 243251 is 16-powersmooth since its greatest prime factor power is $2^4 = 16$. The number is also 17-powersmooth, 18-powersmooth, 19-powersmooth, etc [36].

2.4. Literature Reviews

There are many published works related to breaking RSA algorithms some of them are as the following:

"2.4.1.Efficient Method For Breaking RSA Scheme", 2008, [7].

Such method is the algorithm to attack the system (RSA), this algorithm aims to obtain the private key from the public key. This method depends on the analysis of the factors of the variable n is effective in certain circumstances, which are usually an easy process if abounded of iterations, and is placed in appropriate limits to restrict the public key. It is better than some existing algorithms because they are faster and take fewer times respectively.

THE ATTACK ALGORITHM

The steps of the proposed algorithm are as follows :

1. Obtain entity A public key (e, n).
2. Convert the modulus n to binary bits.
3. Let b represent the number of bits of n.
4. Compute $d = [b / 4]$.
5. Find $ed \equiv 1 + k(n - s + 1) \pmod{2^b}$.
6. Repeat k from 1 to e until $p^2 - s * p + n \equiv 0 \pmod{2^b}$ is true
 - a. Compute $ed \equiv 1 + k(n - s + 1) \pmod{2^d}$
 - b. Compute $p^2 - s * p + n \equiv 0 \pmod{2^d}$
7. Compute $p_0 \equiv p \pmod{2^d}$.
8. Compute $q_0 * p_0 \equiv n \pmod{2^d}$.
9. Compute $\theta(n)$ as follows:
 - Compute
 $n \equiv (2^d * x + p_0) * (2^d * y + q_0)$
 - Compute $p = (2^d * x + p_0)$
 - Compute $q = (2^d * y + q_0)$
 - $\theta(n) = (p - 1)(q - 1)$
10. Compute $d = e * d - k * \theta(n) = 1$.

That the proposed method works like the previous method, but here we generate the key from the three primes numbers not two, and through this method will overcome the weak point in the old way especially if we look at the points 8 and 9, a process factorization of

the number N so well that more the number of prime numbers, consisting of the number N , it will increase the difficulty of the analysis.

From here will be the process of generating the number N is faster than the old way will therefore be the generation of public and private key is much faster than the old method.

"2.4.2. On Using RSA with low exponent in a public key Network", 1986,[37].

In order to speed up RSA encryption (and signature verification) it is useful to use small value for the public exponent e , say $e = 3$. However, this opens up RSA to the following attack, discovered by Hastad .

Let us start with a simpler version. Suppose Bob wishes to send the same message M to k recipients, all of whom are using public exponent equal to 3. He obtains the public keys $\{N_i, e_i\}$ for $i = 1, \dots, k$, where $e_i = 3$ for all i . Naively, Bob computes the ciphertext $C_i = M^3 \bmod N_i$ for all i and sends C_i to the i th recipient .

A simple argument shows that as soon as $k > 3$, the message M is no longer secure. Suppose Eve intercepts $C_1, C_2,$ and C_3 , where $C_i = M^3 \bmod N_i$. We may assume $\gcd(N_i, N_j) = 1$ for all $i \neq j$ (otherwise, it is possible to compute a factor of one of the N_i 's.) By the Chinese

Remainder Theorem, she may compute $C \in \mathbb{Z}^*_{N_1 N_2 N_3}$ such that $C \equiv C_i \pmod{N_i}$. Then $C \equiv M^3 \pmod{N_1 N_2 N_3}$; however, since $M < N_i$ for all i , we have $M^3 < N_1 N_2 N_3$. Thus $C = M^3$ holds over the integers, and Eve can compute the cube root of C to obtain M .

Hstad proves a much stronger result. To understand it, consider the following naive defense against the above attack. Suppose Bob applies a pad to the message M prior to encrypting it so that the recipients receive slightly different messages. For instance, if M is m bits long, Bob might encrypt $i \cdot 2^m + M$ and send this to the i th recipient. Hstad proved that this linear padding scheme is not secure. In fact he showed that any fixed polynomial applied to the message will result in an insecure scheme.

Theorem 3.1 (Hstad) Suppose N_1, \dots, N_k are relatively prime integers and set $N_{\min} = \min_i(N_i)$. Let $g_i(x) \in \mathbb{Z}_{N_i}[x]$ be k polynomials of maximum degree d . Suppose there exists a unique $M < N_{\min}$ satisfying

$$g_i(M) = 0 \pmod{N_i} \quad \text{for all } i \in \{0, \dots, k\}.$$

Furthermore suppose $k > d$. There is an efficient algorithm which, given $\{N_i, g_i(x)\}$ for all i , computes M .

Proof Since the N_i are relatively prime, we may use the Chinese Remainder Theorem to compute coefficients T_i satisfying $T_i \equiv 1 \pmod{N_i}$ and $T_i \equiv 0 \pmod{N_j}$ for all $i \neq j$. Setting $g(x) := \sum_i T_i g_i(x)$ we see $g(M) \equiv 0 \pmod{\prod N_i}$. Since the T_i are nonzero we have that $g(x)$ is not identically zero. If the leading coefficient of $g(x)$ is not one, then we may multiply by its inverse to obtain a monic polynomial $g(x)$ [48].

The degree of $g(x)$ is at most d . By Coppersmith's Theorem we may compute all integer roots x_0 satisfying $g(x_0) \equiv 0 \pmod{\prod N_i}$ and $|x_0| < (\prod N_i)^{1/d}$. But we know $M < N_{\min} < (\prod N_i)^{1/k} < (\prod N_i)^{1/d}$, so M is such a root .

This can be applied to the problem of broadcast RSA as follows. Suppose the i th plaintext is padded with a polynomial $f_i(x)$, so that $C_i \equiv (f_i(M))^{e_i} \pmod{N_i}$. Then the polynomials $g_i(x) := (f_i(x))^{e_i} - C_i$ satisfy the above relation. The attack succeeds once $k > \max_i(e_i \cdot \deg f_i)$ [48].

We note that Hastad's original result was significantly weaker, requiring $k = O(d^2)$ messages where $d = \max_i(e_i \cdot \deg f_i)$. This is because the original result used the Hastad method for solving polynomial congruence's instead of the full Coppersmith method.

This attack suggests that randomized padding should be used in RSA encryption.

2.4.3."Attack on the Cryptographic Scheme", 1994, [38].

The author has introduced a new type of attacks on RSA which enable a passive adversary to recover such message from the corresponding cipher text .This attack is of practical importance since many public key encryption schemes have been proposed which require the encryption of polynomial related messages[39], For instance include the key distribution protocol of Tatebayashi, Matsuzaki, and Newman ,and the verifiable signature scheme of Franklin and Haber [40].

Coppersmith [41] introduced an efficient method for finding a root of a polynomial of degree k over zn , where n is the RSA modulus, provided that there is a root smaller than $n^{1/k}$. The method produced two types of attacks on RSA with small encryption public key. When $e = 3$ and if an opponent knows an encrypted message c and more than $2/3$ of the message m related to c then the opponent can efficiently discover the remainder of the message m . Assume now that messages are padded with random bit strings and encrypted with public key $e = 3$. If the opponent knows two encrypted messages $c1$ and $c2$ which correspond to two encryptions of the same message m with different padding, then the opponent can efficiently retrieval m

given that the padding is less than $1/9$ of the size of the modulus n . The second attack proposes that care should be exercised if employing random padding in conjunction with a small encryption public key".

2.4.4. Factorization of a 512(bit RSA Modulus),1999,[9].

In this research, they try to breaking the code (RSA 512), have been using the system network of computers linked with each other to break the code using the analysis of the factors of the variable n and then we can find a public key and private key. The code encrypted messages can be broken in this way.

Another way to break RSA is to find a technique to compute e -th roots mod n . Since $c = m^e \text{ mod } n$, the e -th root of $c \text{ mod } n$ is the message m . This attack would allow someone to recover encrypted messages and forge signatures even without knowing the private key. This attack is not known to be equivalent to factoring. No general methods are currently known that attempt to break RSA in this way. However, in special cases where multiple related messages are encrypted with the same small exponent, it may be possible to recover the messages.

The attacks just mentioned are the only ways to break RSA in such a way as to be able to recover all messages encrypted under a given key. There are other methods, however, that aim to recover single messages; success would not enable the attacker to recover other messages encrypted with the same key. Some people also studied whether part of the message can be recovered from an encrypted message .

2.4.5."CRYPTANALYSIS OF RSA USING ALGEBRAIC AND LATTICE METHODS", 2002, [48].

To speed up RSA decryption and signing, it is tempting to use a small secret exponent d rather than a random $d \leq \theta(N)$. Since modular exponentiation takes time linear in $\log_2 d$, using a d that is substantially shorter than N can improve performance by a factor of 10 or more. For instance, if N is 1024 bits in length and d is 80 bits long, this results in a factor of 12 improvements while keeping d large enough to resist exhaustive search .

Unfortunately, a classic attack by Wiener shows that a sufficiently short d leads to an efficient attack on the system. His method uses approximations of continued fractions. This attack is stated in the following theorem .

Theorem 3.2 (Wiener) Suppose $N = pq$ and $(N)^{0.5}/2 < q < p < N^{0.5}$. Furthermore suppose $d < 1/3 N^{1/4}$. There is an algorithm which, given N and e , generates a list of length $\log N$ of candidates for d , one of which will equal d . This algorithm runs in time linear in $\log N$.

Proof: Since $ed \equiv 1 \pmod{\theta(N)}$, there is some k such that $ed - k\theta(N) = 1$. We may write this as:

$$|e/\theta(N) - k/d| = 1/d\theta(N).$$

Hence $e/\theta(N)$ is an approximation to k/d . The attacker does not know $\theta(N)$, but he does know N . Since $(N)^{0.5}/2 < q < p < 2N^{0.5}$ we have $p + q - 1 < 3(N)^{0.5}$ and thus $N - \theta(N) < 3(N)^{0.5}$. Now if the attacker uses e/N as an approximation we find :

$$|e/\theta(N) - k/d| = |(ed - k\theta(N) + k\theta(N) - kN) / dN|$$

$$= |1 - k(N - \theta(N)) / Nd| \leq |3kN^{0.5} / Nd| = |3k / dN^{0.5}|$$

Since $e < \theta(N)$, we know $k < d < 1/3 N^{1/4}$ Thus

$$|e/N - k/d| \leq 1/dN^{1/4} < 1/2d^2$$

This is a classic approximation relation, and there are well-known methods to solve it. Such methods produce a list of all integers pairs (k_i, d_i) satisfying $\gcd(k_i, d_i) = 1$ and

$$|e/N - k_i/d_i| < 1/2d_i^2$$

This list is of length at most $\log N$. Since $ed - k\theta(N) = 1$ we know $\gcd(k, d) = 1$.

Hence, $d = d_i$ for some $i \in \{1, \dots, \log N\}$.

2.5. Summary

The previous section was a presentation of related literature, most of which is very important for the thesis, it's used to view what is being reached by the experts and researchers to find out where your site where it is now, and how to proceed in the next step or develop things more.

This knowledge, experts are keen to use many methods to create a competition to find the best solutions to lead them to know the private key of the RSA. Each of them has a special method, and most of them succeeded in finding the private key. But whenever the algorithm is broken, the system is to impose protection system safer.

And of the most important queries and questions that evaluate the system decryption

Securities are the following:

Method finds private key from composed p and q value.

Method could break the public key through the length of n value.

Method was depending on easy algorithm or hard algorithm to break RSA algorithms.

Method was needed a little effort or big effort of machines and personnel to find this key.

And so, each researcher or intruder beginning of the organization and develop a special method by relying on one or more of these factors.

Factoring of n .

p or q .

d .

Chapter three {The Proposed Method}

3.1.Introduction

In this chapter we will explain the RSA algorithm, considering its details and then present the new algorithm on the same existing style of the original algorithm.

Then we prove the new algorithm through mathematical equations that have been proven on the validity of its works and give some examples for explanation. Then we conducted a study to explain the benefits of the new algorithm in terms of generating the key and complexity in terms of the authentication.

3.2.The RSA cryptosystem

This algorithm contains two keys, public key and private key, the public key is known by anyone and is used to encrypt messages, while the private key is confidential and used to decrypt messages.

This algorithm is based mainly on remnants mathematics division, which will summarize the following points.

3.2.1Mathematics of the RSA Algorithm

Given: $n = pq$ where p and q are distinct primes [11].

$$\text{Gcd} (e , \Theta(n)) = 1$$

$$de = 1 \text{ mod } \Theta (n)$$

When $y = x^e \text{ mod } n$ and $X = y^d \text{ mod } n$

Where: $x < \min\{ p, q \}$

Prove that: $X = x \pmod n \forall x < n$

Proof: $X = x^{de} \pmod n$

$$de = 1 \pmod{\Theta(n)}$$

$\Theta(n) = (p-1)(q-1)$ if p and q are distinct primes

$$de = 1 + k(p-1)(q-1)$$

$$X = x^{1+k(p-1)(q-1)}$$

$$X = x \cdot (x^{(p-1)})^{k(q-1)}$$

But $x^{(p-1)} = x^{\Theta(p)}$ and $x \in \mathbb{Z}_p^*$

So $x^{(p-1)} = 1 \pmod p$... *Fermat/Euler Theorem*

$$\text{So } X = x \cdot (1 \pmod p)^{k(q-1)}$$

$$\text{So } X = x \pmod p$$

$$\text{Similarly } X = x \pmod q$$

Because p and q are co-prime we can use the *Chinese remainder Theorem*

Therefore $X = x \pmod{pq}$.

$$\square \Rightarrow X = x \pmod n$$

* *Fermat/Euler Theorem*

Theorem $\forall x \in \mathbb{Z}_n^*, x^{\Theta(n)} \equiv 1 \pmod n$
--

Proof $\mathbb{Z}_n = \{ 1, 2, \dots, (n-1) \} \pmod n$ [43].

$$\mathbb{Z}_n^* = \{ x \in \mathbb{Z}_n : \gcd(x, n) = 1 \}$$

The order of \mathbb{Z}_n^* is $\Theta(n)$ and is called the *Euler Function*.

We let $u_1, \dots, u_{\Theta(n)}$ be an enumeration of all the elements of \mathbb{Z}_n^*

:

It is clear that $x.u_1, \dots, x.u_{\Theta(n)}$ is also an enumeration of all the elements of Z_n^* :

Therefore $x.u_1, \dots, x.u_{\Theta(n)} = u_1, \dots, u_{\Theta(n)}$

So $x^{\Theta(n)} \cdot u_1, \dots, u_{\Theta(n)} = u_1, \dots, u_{\Theta(n)}$

We let $g = u_1, \dots, u_{\Theta(n)}$

$g \in Z_n^* \iff g^{-1} \in Z_n^*$

So $x^{\Theta(n)} \cdot u_1, \dots, u_{\Theta(n)} \cdot g^{-1} = u_1, \dots, u_{\Theta(n)} \cdot g^{-1}$

So $x^{\Theta(n)} = 1 \pmod n$

**** Chinese Remainder Theorem**

Theorem $x = y \pmod p$
 $x = y \pmod q$
 $x = y \pmod{pq}$

Proof $x = y \pmod p$ [44].
 $\iff p \mid (x - y)$
 $x = y \pmod q$
 $\iff q \mid (x - y)$

p and q are co-prime
 $\iff pq \mid (x - y)$
 $\iff x = y \pmod{pq}$

3.2.2. Proof of the RSA Algorithm

Given positive integers n , e , and d such that [45]
 $n = pq$, where p and q are distinct primes.....(1).

$$\gcd(e, \phi(n)) = 1.....(2).$$

$$de \equiv 1 \pmod{\phi(n)}.....(3).$$

Define the public and private key algorithms of a message m to be respectively, for $0 \leq m < n$,

$$RSA\ Public(m) = m^e \pmod{n}.....(4).$$

$$RSA\ Private(m) = m^d \pmod{n}.....(5).$$

Prove that

$$m = RSA\ Private(RSA\ Public(m)), \text{ and that}.....(6).$$

$$m = RSA\ Public(RSA\ Private(m)).....(7).$$

(Prove that the two algorithms (6) and (7) can be used inversely to obtain the message m or "Does RSA Encryption actually work?")

Proof. By substituting equations (4) and (5) into (6) and (7) respectively, we can say that [45].

$$\begin{aligned} RSA\ Private(RSA\ Public(m)) &= (m^e \pmod{n})^d \pmod{n} \\ &= m^{de} \pmod{n}. \end{aligned}$$

We can also say that

$$\begin{aligned} RSA\ Public(RSA\ Private(m)) &= (m^d \pmod{n})^e \pmod{n} \\ &= m^{de} \pmod{n}. \end{aligned}$$

Therefore, equations (6) and (7) are equivalent, or

$$RSA\ Private(RSA\ Public(m)) = RSA\ Public(RSA\ Private(m)).$$

If we can prove

$$m = m^{de} \pmod n$$

Then the proof will be complete [45].

It is given that

$$de \equiv 1 \pmod{\Theta(n)} \dots \dots \dots (3).$$

By definition of mods we can write (3) as

$$\Theta(n) \mid de - 1 \dots \dots \dots (8).$$

Since $\Theta(n) = \Theta(p)\Theta(q)$ only when p and q are relatively primes, as in this case, we have

$$\Theta(n) = \Theta(p)\Theta(q)$$

And by substitution into (8) we have [45].

$$\Theta(p)\Theta(q) \mid de - 1.$$

By properties of divisors, we can write

$$\Theta(p) \mid de - 1,$$

$$\Theta(q) \mid de - 1$$

Where there must be an integer k such that

$$de - 1 = k \Theta(p).$$

Since p is prime, the *Euler phi function* states that $\Theta(p) = p - 1$, so

$$de - 1 = k(p - 1) \dots \dots \dots (9).$$

By the symmetric property of mods, we can write [45].

$$\begin{aligned} m^{de} &\equiv m^{de} \pmod p \\ &\equiv m^{de-1+1} \pmod p \end{aligned}$$

Which can also be written as?

$$m^{de} \equiv (m^{de-1}) * m \pmod p \dots \dots \dots (10).$$

Substituting (9) into (10), we obtain

$$m^{de} \equiv (m^{k(p-1)}) * m \pmod{p} \dots\dots\dots (11).$$

Since p is a prime, any integer m for (11) will be either [45].

- 1) Relatively prime to p or will
- 2) Be a multiple of p.

When

- 1) m is relatively prime to p, Fermat's Little Theorem states that [45]

$$m^{p-1} \equiv 1 \pmod{p}.$$

By properties of mods, we can write

$$m^{k(p-1)} \equiv 1^k \pmod{p}, \text{ or}$$

$$m^{k(p-1)} \equiv 1 \pmod{p} \dots\dots\dots(12).$$

By combining (11) and (12), we obtain

$$m^{de} \equiv 1 * m \pmod{p}, \text{ or}$$

$$m^{de} \equiv m \pmod{p} \dots\dots\dots(13).$$

In the second case where

- 2) m is a multiple of p, if $p \mid m$, then for any integer k $p \mid m^k$ [45].

From the properties of mods we can write

$$m^{de} \equiv 0 \pmod{p},$$

$$m \equiv 0 \pmod{p}.$$

Thus we can write

$$m^{de} \equiv m \pmod{p}.$$

Therefore, for all m,

$$m^{de} \equiv m \pmod{p} \dots\dots\dots(14)$$

and applying the same process for q we can write

$$m^{de} \equiv m \pmod{q}.$$

By the modular property of congruence which states that when m and n are relatively prime (as in our given statements), $a \equiv b \pmod{m}$, and $a \equiv b \pmod{n}$, then $a \equiv b \pmod{mn}$, we can write [45].

$$m^{de} \equiv m \pmod{pq}$$

$$\equiv m \pmod{n}.$$

By the modular property of symmetry, we can write [45].

$$m \equiv m^{de} \pmod{n} \dots\dots\dots (15)$$

Since we have limited m to $0 \leq m < n$, only one integer will satisfy (15), and so

$$m = m^{de} \pmod{n} \dots\dots\dots (16)$$

If we substitute equation (16) with the original equations [45] we get:

$$RSAPrivate (RSAPublic(m)) = m^{de} \pmod{n}, \text{ and}$$

$$RSAPublic (RSAPrivate(m)) = m^{de} \text{ mod } n$$

We obtain, for $0 \leq m < n$,

$$RSAPrivate (RSAPublic(m)) = m, \text{ \& } RSAPublic (RSAPrivate(m)) = m.$$

3.2.3. How It Works

RSA cryptography is based on the following theorems:

Theorem 1 (Fermat's Little Theorem) If p is a prime number, and a is an integer such that $(a, p) = 1$, then [42]

$$a^{p-1} = 1(\text{mod } p).$$

Proof: Consider the numbers $(a \cdot 1), (a \cdot 2), \dots, (a \cdot (p - 1))$ for all modulo p . They are all different. If any of them were the same, say $a \cdot m = a \cdot n \text{ (mod } p)$, then $a \cdot (m - n) = 0(\text{mod } p)$ so $m - n$ must be a multiple of p . But since all m and n are less than p , $m = n$.

Thus $a \cdot 1, a \cdot 2, \dots, a \cdot (p-1)$ must be a rearrangement of $1, 2, \dots, (p-1)$. So modulo p , we have [42]:

$$\prod_{i=1}^{p-1} i = \prod_{i=1}^{p-1} a \cdot i = a^{p-1} \prod_{i=1}^{p-1} i,$$

So $a^{p-1} = 1(\text{mod } p)$.

Theorem 2 (Fermat's Theorem Extension): If $(a, m) = 1$ then $a^{\Theta(m)} = 1 \pmod{m}$, where $\Theta(m)$ is the number of integers less than m that are relatively prime to m . The number m is not necessarily to be prime [42].

Proof: Same idea as above. Suppose $\Theta(m) = n$. Then suppose that the n numbers less than m that is relatively prime to m is:

$$a_1, a_2, a_3, \dots, a_n.$$

Then $a \cdot a_1, a \cdot a_2, \dots, a \cdot a_n$ are also relatively prime to m , and must all be different,

so they must just be a rearrangement of the a_1, \dots, a_n in some order.

Thus [42]:

$$\prod_{i=1}^n a_i = \prod_{i=1}^n a \cdot a_i = a^n \prod_{i=1}^n a_i,$$

Modulo m , so $a^n = 1 \pmod{m}$.

Theorem 3 (Chinese Remainder Theorem) Let p and q be two numbers (not necessarily primes), but such that $(p, q) = 1$. Then if $a = b \pmod{p}$ and $a = b \pmod{q}$ we have $a = b \pmod{pq}$ [42].

Proof: If $a = b \pmod{p}$ then p divides $(a - b)$. Similarly, q divides $(a - b)$. But p and q are relatively prime, so pq divides $(a - b)$. Consequently, $a = b \pmod{pq}$.

Proof of the Main Result:

Based on the theorems above, here is why the RSA encryption scheme works.

Let p and q be two different (large) prime numbers, let $0 \leq M < pq$ be a secret message, let d be an integer (usually small) that is relatively prime to $(p - 1)(q - 1)$, and let e be a number such that $de = 1 \pmod{(p - 1)(q - 1)}$. (We will see later how to generate this e given d) [42].

The encoded message is $C = M^e \pmod{pq}$, so we need to show that the decoded message is given by $M = C^d \pmod{pq}$.

Proof: Since $de = 1 \pmod{(p-1)(q-1)}$, $de = 1 + k(p-1)(q-1)$ for some integer

k . Thus:

$$C^d = M^{de} = M^{1+k(p-1)(q-1)} = M \cdot (M^{(p-1)(q-1)})^k.$$

If M is relatively prime to p , then

$$M^{de} = M \cdot (M^{p-1})^{k(q-1)} = M \cdot (1)^{k(q-1)} = M \pmod{p} \dots \dots \dots (1)$$

By the *extension of Fermat's Theorem* giving $M^{p-1} = 1 \pmod{p}$ followed by a multiplication of both sides by M . But if M is not relatively primes to p , then M is a multiple of p , so *equation 1* still holds because both sides will be zero, *modulo p* [42].

By exactly the same reasoning,

$$M^{de} = M \cdot M^{q-1} = M \pmod{q} \dots\dots\dots(2)$$

If we apply the *Chinese remainder theorem* to equations 1 and 2, we obtain the result we want:

$$M^{de} = M \pmod{pq} \dots\dots\dots(3)$$

Finally, given the integer d , we will need to be able to find another integer e such that $de = 1 \pmod{(p-1)(q-1)}$. To do so we can use the *extension of Fermat's theorem* to get $d^{\ominus((p-1)(q-1))} = 1 \pmod{(p-1)(q-1)}$, so $d^{\ominus((p-1)(q-1))-1} \pmod{(p-1)(q-1)}$ is a suitable value for e [42].

3.2.4. Analysis of Rsa Cryptosystem

The RSA Cryptosystem requires the use of a public key and a private key. Both these keys must fulfill certain conditions to ensure the integrity of the system. The following steps illustrate the key generation algorithm for RSA [47]:

1. Choose two large prime numbers of approximately the same size, namely p and q .
2. Compute the product of these two primes, $n = pq$.
3. Also, compute the value of $\phi(n) = (p-1)(q-1)$.

4. Choose an integer e between 1 and $\varphi(n)$ such that $\gcd(e, \varphi(n)) = 1$.
5. Finally, compute d whereby $d = e^{-1} \text{ mod}(\varphi(n))$.

The public key is (n, e) whereas the private key is (n, d) .

COMPLEXITY FOR STEP 1:

1. For selecting the first prime number p , the complexity can be computed as the product of the number of numbers to be tested for primality and the complexity of one primality test. Complexity of MILLER-RABIN [46] gives the above mentioned complexities so the complexity for finding a prime number is $O(s. (\log_2 p)^3 . \ln p)$ [47].

2. Similarly for the second prime number q , complexity is $O(s. (\log_2 q)^3 . \ln q)$.

COMPLEXITY FOR STEP 2:

As step 2 involves only the computation of n , which is the product of p and q . So the complexity of step 2 is $O(\log_2 p. \log_2 q)$ binary operations [47].

COMPLEXITY FOR STEP 3:

By MODULAR-EXPONENTIATION, the complexity for the second part is $\Theta(n) - 1$. Therefore complexity of the step 3 is [47]

$$O((\log_2(p-1). (q-1))^3. ((p-1). (q-1)-1)).$$

COMPLEXITY FOR STEP 4:

The complexity for step 4 is $O(\log_2 (p-1).(q-1) + \gcd(e, (p-1).(q-1)))$, as we know that e and $\Theta(n)$ are prime to each other so $\gcd(e, (p-1).(q-1)) = 1$, and so complexity is [47].

$$O((\log_2(\log_2 p-1).(\log_2 q-1))^3 + 1)$$

3.3.. The New Approach RSA cryptosystem

The RSA Enhance

From here came the subject of research, which instead of using two primes numbers to generate a public key and private key, now use three primes numbers with reduced size, generates the variable N

large and the process of analysis of the factors it more difficult than the original way, as well as using three primes numbers, it increases the ease of generating Public key and private key, which means that it saves us time and effort.

To illustrate this speech louder image take the example of the use of an algorithm RSA original and proposed algorithm.

The key strength of the RSA depends on the two prime numbers p and q . The process of factorizing of n will lead to gain the values of p and q . It is much easier to find two numbers from factoring n than finding the value of three numbers from n .

The proposed method is to use three prime numbers to construct the value of n . In this case it is very difficult for the intruder to find the three values from factoring n . Therefore the new algorithm will be as follows.

Encrypts messages through the following algorithm, which is divided into three steps:

1. Key Generation

(a). Choose three distinct prime numbers p , q and s .

(b). Find n such that $n = p * q * s$.

n will be used as the modulus for both the public and private keys.

(c). Find the Phi of n , $\varphi(n) = (p-1)(q-1)(s-1)$.

(d). Choose an e such that $1 < e < \varphi(n)$, and such that e and $\varphi(n)$ share no Divisors other than 1 (e and $\varphi(n)$ are relatively prime). e is kept as the public key exponent.

(e). Determine d (using modular arithmetic) which satisfies the congruence relation

$$d * e \equiv 1 \pmod{\varphi(n)}.$$

In other words, pick d such that $de - 1$ can be evenly divided by $(p-1)(q-1)(s-1)$, the Phi, or $\varphi(n)$. This is often computed using the

Extended Euclidean Algorithm, since e and $\varphi(n)$ are relatively prime and d is to be the modular multiplicative inverse of e , d is kept as the private key exponent.

The public key has modulus n and the public (or encryption) exponent e . The private key has modulus n and the private (or decryption) exponent d , which is kept secret .

2. Encryption

(a). A transmits his/her public key (modulus n and exponent e) to B, keeping his/her private key secret.

(b). When Person B wishes to send the message " M " to Person A, she/he first converts M to an integer such that $0 < m < n$ by using agreed upon reversible protocol known as a padding scheme.

(c). Person B computes, with Person A's public key information, the cipher text c corresponding to

$$c \equiv m^e \pmod{n}.$$

(d). Person B now sends message " M " in cipher text, or c , to Person A .

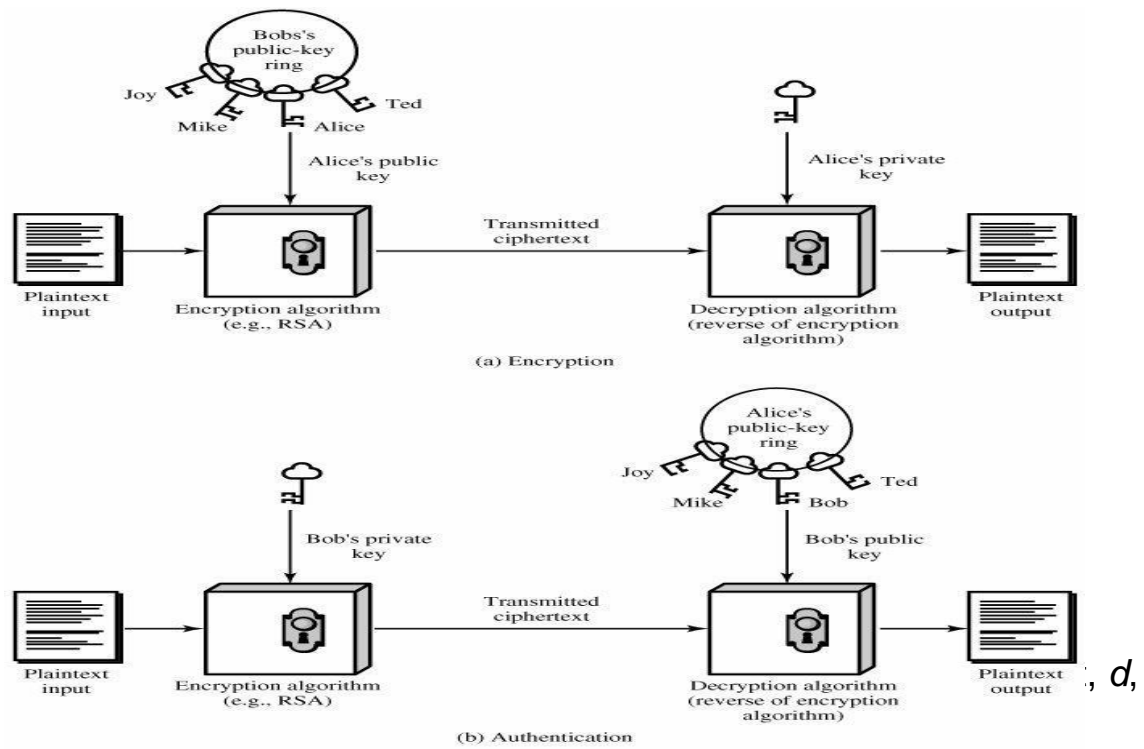


Figure 3.1: Public-Key Cryptography

$$m \equiv c^d \pmod{n}.$$

Given m , Person A can recover the original message " M " by reversing the padding scheme.

This procedure works since

$$c \equiv m^e \pmod{n},$$

$$c^d \equiv (m^e)^d \pmod{n},$$

$$c^d \equiv m^{de} \pmod{n}.$$

By the symmetry property of modular we have that

$$m^{de} \equiv m^{de} \pmod{n}.$$

Since $de = 1 + k \phi(n)$, we can write

$$m^{de} \equiv m^{1 + k \phi(n)} \pmod{n},$$

$$m^{de} \equiv m(m^k)^{\varphi(n)} \pmod{n},$$

$$m^{de} \equiv m \pmod{n}.$$

From Euler's Theorem and the Chinese Remainder Theorem, we can show that this is true for all m and the original message $c^d \equiv m \pmod{n}$, is obtained .

3.3.1 Mathematics of the New Approach RSA Algorithm

Given: $n = pqs$ where p , q and s are distinct primes .

$$\gcd(e, \Theta(n)) = 1$$

$$de \equiv 1 \pmod{\Theta(n)}$$

When $y = x^e \pmod{n}$ and $X = y^d \pmod{n}$

Where: $x < \min\{p, q, s\}$

Prove that: $X = x \pmod{n} \forall x < n$

Proof: $X = x^{de} \pmod{n}$

$$de \equiv 1 \pmod{\Theta(n)}$$

$\Theta(n) = (p-1)(q-1)(s-1)$ if p , q and s are distinct primes

$$de = 1 + k(p-1)(q-1)(s-1)$$

$$X = x^{1+k(p-1)(q-1)(s-1)}$$

$$X = x \cdot ((x^{(p-1)})^{(q-1)})^{k(s-1)}$$

But $x^{(p-1)} = x^{\Theta(p)}$ and $x \in \mathbb{Z}_p^*$

So $x^{(p-1)} = 1 \pmod{p}$...Fermat/Euler Theorem

$$\text{So } X = x \cdot (1 \pmod{p})^{k(q-1)(s-1)}$$

So $X = x \pmod{p}$

Similarly $X = x \pmod{q}$

$$X = x \text{ mod } s$$

Because p , q and s are co-prime we can use the *Chinese remainder Theorem*

Therefore $X = x \text{ mod } pqs$.

$$\square \Rightarrow X = x \text{ mod } n$$

3.3.2. Proof of the New Approach RSA Algorithm

Given positive integers n , e , and d such that .

$n = pqs$, where p , q and s are distinct primes.....(1).

$$\gcd (e, \Theta(n)) = 1 \dots\dots\dots(2).$$

$$de \equiv 1 \pmod{\Theta(n)} \dots\dots\dots(3).$$

Define the public and private key algorithms of a message m to be respectively, for $0 \leq m < n$,

$$RSAPublic(m) = m^e \text{ mod } n, \dots\dots\dots(4).$$

$$RSAPrivate(m) = m^d \text{ mod } n. \dots\dots\dots(5).$$

Prove that

$$m = RSAPrivate (RSAPublic(m)), \text{ and that} \dots\dots\dots(6).$$

$$m = RSAPublic (RSAPrivate(m)) \dots\dots\dots(7).$$

(Prove that the two algorithms (6) and (7) can be used inversely to obtain the message m , or "Does RSA Encryption actually work?")
 Proof. By substituting equations (4) and (5) into (6) and (7) respectively, we can say that .

$$\begin{aligned} \text{RSAPrivate}(\text{RSAPublic}(m)) &= (m^e \bmod n)^d \bmod n \\ &= m^{de} \bmod n. \end{aligned}$$

We can also say that

$$\begin{aligned} \text{RSAPublic}(\text{RSAPrivate}(m)) &= (m^d \bmod n)^e \bmod n \\ &= m^{de} \bmod n. \end{aligned}$$

Therefore, equations (6) and (7) are equivalent, or
 $\text{RSAPrivate}(\text{RSAPublic}(m)) = \text{RSAPublic}(\text{RSAPrivate}(m)).$

If we can prove

$$m = m^{de} \bmod n$$

Then the proof will be complete .

It is given that

$$de \equiv 1 \pmod{\Theta(n)} \dots \dots \dots (3).$$

By definition of mods we can write (3) as

$$\Theta(n) \mid de - 1 \dots \dots \dots (8).$$

Since $\Theta(n) = \Theta(p)\Theta(q) \Theta(s)$ only when p and q are relatively prime, as in this case, we have

$$\Theta(n) = \Theta(p)\Theta(q) \Theta(s)$$

And by substitution into (8) we have .

$$\Theta(p)\Theta(q)\Theta(s) \mid de - 1.$$

By properties of divisors, we can write

$$\Theta(p) \mid de - 1,$$

$$\Theta(q) \mid de - 1,$$

$$\Theta(s) \mid de - 1$$

Where there must be an integer k such that

$$de - 1 = k \Theta(p).$$

Since p is prime, the *Euler phi function* states that $\Theta(p) = p - 1$, so

$$de - 1 = k(p - 1) \dots \dots \dots (9).$$

By the symmetric property of mods, we can write .

$$\begin{aligned} m^{de} &\equiv m^{de} \pmod{p} \\ &\equiv m^{de - 1 + 1} \pmod{p} \end{aligned}$$

Which can also be written as?

$$m^{de} \equiv (m^{de - 1}) * m \pmod{p} \dots \dots \dots (10).$$

Substituting (9) into (10), we obtain

$$m^{de} \equiv (m^{k(p - 1)}) * m \pmod{p} \dots \dots \dots (11).$$

Since p is prime any integer m for (11) will be either .

- 1) Relatively prime to p or will
- 2) Be a multiple of p .

When

1) m is relatively prime to p , Fermat's Little Theorem states that .

$$m^{p - 1} \equiv 1 \pmod{p}.$$

By properties of mods, we can write

$$m^{k(p-1)} \equiv 1^k \pmod{p}, \text{ or}$$

$$m^{k(p-1)} \equiv 1 \pmod{p} \dots \dots \dots (12).$$

By combining (11) and (12), we obtain

$$m^{de} \equiv 1 * m \pmod{p}, \text{ or}$$

$$m^{de} \equiv m \pmod{p} \dots \dots \dots (13).$$

In the second case where

2) m is a multiple of p , if $p \mid m$, then for any integer k $p \mid m^k$.

From the properties of mods we can write

$$m^{de} \equiv 0 \pmod{p},$$

$$m \equiv 0 \pmod{p}.$$

Thus we can write

$$m^{de} \equiv m \pmod{p}.$$

Therefore, for all m ,

$$m^{de} \equiv m \pmod{p} \dots \dots \dots (14)$$

and applying the same process for q and s we can write

$$m^{de} \equiv m \pmod{q}.$$

$$m^{de} \equiv m \pmod{s}.$$

By the modular property of congruence which states that when m and n are relatively prime (as in our given statements), $a \equiv b \pmod{m}$, and $a \equiv b \pmod{n}$, then $a \equiv b \pmod{mn}$, we can write .

$$m^{de} \equiv m \pmod{pqs}$$

$$\equiv m \pmod{n}.$$

By the modular property of symmetry, we can write .

$$m \equiv m^{de} \pmod{n} \dots \dots \dots (15)$$

Since we have limited m to $0 \leq m < n$, only one integer will satisfy (15), and so

$$m = m^{de} \pmod{n} \dots \dots \dots (16)$$

If we substitute equation (16) with our original equations we get:

$$RSAPrivate (RSAPublic(m)) = m^{de} \pmod{n}, \text{ and}$$

$$RSAPublic (RSAPrivate(m)) = m^{de} \pmod{n}$$

We obtain, for $0 \leq m < n$,

$$RSAPrivate (RSAPublic(m)) = m, \text{ \& } RSAPublic (RSAPrivate(m)) = m.$$

3.3.3. ANALYSIS of the RSA Enhance

The RSA Enhance requires the use of a public key and a private key. Both these keys must fulfill certain conditions to ensure the integrity of the system. The following steps illustrate the key generation algorithm for the proposed RSA :

1. Choose three large prime numbers of approximately the same size, namely p , q and s
2. Compute the product of these three primes, $n = pqs$.

3. Also, compute the value of $\varphi(n) = (p-1)(q-1)(s-1)$.
 4. Choose an integer e between 1 and $\varphi(n)$ such that $\gcd(e, \varphi(n)) = 1$.
 5. Finally, compute d whereby $d = e^{-1} \text{ mod}(\varphi(n))$.
- The public key is (n, e) whereas the private key is (n, d) .

COMPLEXITY FOR STEP 1:

1. For selecting the first prime number p , the complexity can be computed as the product of the number of numbers to be tested for primality and the complexity of one primality test. Complexity of MILLER-RABIN [46] gives the above mentioned complexities so the complexity for finding a prime number is $O(s \cdot (\log_2 p)^3 \cdot \ln p)$.

Similarly for the second prime number q , complexity is $O(s \cdot (\log_2 q)^3 \cdot \ln q)$.

Similarly for the third prime number s , complexity is $O(s \cdot (\log_2 s)^3 \cdot \ln s)$.

COMPLEXITY FOR STEP 2:

As step 2 involves only the computation of n , which is the product of p , q and s . So the complexity of step 2 is $O(\log_2 p \cdot \log_2 q \cdot \log_2 s)$ binary operations.

COMPLEXITY FOR STEP 3:

By MODULAR-EXPONENTIATION, the complexity for the second part is $\Theta(n) - 1$. Therefore complexity of the step 3 is :

$$O((\log_2(p-1) \cdot (q-1) \cdot (s-1))^3 \cdot ((p-1) \cdot (q-1) \cdot (s-1) - 1)).$$

COMPLEXITY FOR STEP 4:

The complexity for step 4 is $O(\log_2(p-1) \cdot (q-1) \cdot (s-1) + \gcd(e, (p-1) \cdot (q-1) \cdot (s-1)))$, as we know that e and $\Theta(n)$ are prime to each other so $\gcd(e, (p-1) \cdot (q-1) \cdot (s-1)) = 1$, and so complexity is :
 $O((\log_2(\log_2 p - 1) \cdot (\log_2 q - 1) \cdot (\log_2 s - 1))^3 + 1)$.

3.3.4. Examples of the RSA Enhance

Example One:

Let: $p = 7$, $q = 13$, $s = 19$

Calculate n value

$$n = p \cdot q \cdot s$$

$$n = 7 \cdot 13 \cdot 19$$

$$n = 1729$$

Calculate $\phi(n)$ value

$$\phi(n) = (p-1) \cdot (q-1) \cdot (s-1)$$

$$\phi(n) = (6) \cdot (12) \cdot (18)$$

$$\phi(n) = 1296$$

Select E value: the E value between 1 and $\phi(n)$ and small odd and

$$\text{GCD}(E, \phi(n)) = 1$$

$$E = 17$$

Calculate d

$$D = (1+k*(\phi(n)))/E$$

$$D = (1+k*1296)/17$$

$$k=4$$

$$D = 305$$

Public key= (17,1729)

Private key= (305,1729)

Encryption the M value =88

$$C = 88^{17} \bmod 1729 = 1395$$

Decryption the C value =1395

$$M = 1395^{305} \bmod 1729 = 88$$

Example Two:

Let: p= 4993, q= 4327, s= 4363

Calculate n value

$$n = p * q * s$$

$$n = 4993 * 4327 * 4363$$

$$n = 94261354093$$

Calculate $\phi(n)$ value

$$\phi(n) = (p-1) * (q-1) * (s-1)$$

$$\phi(n) = (4992) * (4326) * (4363)$$

$$\phi(n) = 94199099904$$

Select E value: the E value between 1 and $\phi(n)$ and small odd and $\text{GCD}(E, \phi(n))=1$

$$E = 92333$$

Calculate d

$$D = (1+k \cdot \phi(n))/E$$

$$D = (1+k \cdot 94261354093)/92333$$

$$D = 20955124517$$

Public key= (92333, 94261354093)

Private key= (20955124517, 94261354093)

Encryption the M value =88

$$C = 88^{32416188127} \bmod 481399399260648937 = 19301800791$$

Decryption the C value =1395

$$M = 19301800791^{79836446209} \bmod 481399399260648937 = 88$$

3.4. Breaking the Original RSA and the RSA Enhance

In this part will work the process of comparing in terms

of the factorization of the variable

N, and the numbers will be small and

will take the way Pollard rho Factoring Method and Rabin modulus to be factored.

3.4.1. Pollard rho Factoring Method

This method is based on a combination of two ideas that are also useful for various other factoring methods [7].

THE ATTACK ALGORITHM

set $a = b = 2$;

for $l = 1, 2, \dots$, do

$a = a^2 + 1 \pmod n$

$b = b^2 + 1 \pmod n$

$b = b^2 + 1 \pmod n$

$d = \text{gcd}(a-b, n)$;

If $1 < d < n$ then return d and terminate with success

if $d = n$ then terminate the algorithm with failure

Will try this method on small numbers until we see the power of analysis in the original algorithm and the algorithm is assumed.

Example on RSA algorithm

Pollard rho algorithm for finding a non-trivial factor of $n = 455459$. The following table lists the values of variables a , b , and d and at the end of each an iteration of step for loop of algorithm.

Table 3.1: Breaking RSA Algorithm By Pollard Rho Factoring Method

a	b	d
5	26	1
26	2871	1
677	179685	1
2871	155260	1
44380	416250	1
179685	43670	1
121634	164403	1
155260	247944	1
44567	68343	743

Hence two non-trivial factors of 455459 are 743 and $455459/743=613$.

Example on the new algorithm assumed

$$P= 97 , q= 91 , s= 79$$

$$N= 697333$$

Table 3.2: Breaking RSA Enhance Algorithm By Pollard Rho Factoring Method

a	b	d
5	26	7

$n/7=99619$ is not prime ,it is divisible by 13

$99619/13= 7663$ is not prime, it is divisible by 79.

3.5.Summary

We note in this chapter that we can prove the correctness of the proposed algorithm through the mathematical equations and we have shown its strength through the examples and it was working correctly. As for the algorithm complexity, it is proved by the assumed equations that it is more difficult than the original algorithm. The generation of the variable n will be faster mathematically because it is composed of three prime numbers of less value to be composed of two numbers.

Hence, we note that the algorithm will be faster in terms of generating public key and private key and we proved it through the practical application, either in terms of breaking the algorithm through the factoring, all methods that depend on the analysis of the variable n is composed of two numbers will be ineffective because the algorithm assumed based on three numbers to be a variable n .

The standard way to analyze the variable n is if we have the first component of the variable n , it will find the second number which is also ineffective because if we found first number of the component of the variable n , still there are two numbers are unknown, which will increase the complexity in the analysis.

Chapter four The Experimental Work

4.1. Introduction

Experimental work is indispensable session, where practical implementations for overall proposed system would be applied, and judge the extent of the proposed system is good and useful.. The selected programming language to implement the proposed system is Visual C sharp. Net (C#.net).

C#. net is a multi-form programming language encompassing strong typing, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft within its .NET initiative and later approved as a standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270). C# is one of the programming languages designed for the Common Language Infrastructure.

C# is intended to be a simple, modern, general-purpose, object-oriented programming language [49]. Its development team is led by Anders Hejlsberg. The most recent version is C# 4.0, which was released on April 12, 2010.

4.2. Implementations

Here, we will make a trial run sample to trace the steps of the proposed system on a practical exercise.

4.2.1.. First application (Encryption & Decryption Text)

Now, let's state a simple example for the proposed system to make the matters easier and displaying how much reduce time and effort, see how the first program works and what objective we can gain.

The application of the proposed system has been made to review how to create a public key, private key, encryption, and decryption process using the proposed RSA.

Run application of Encryption

Figure (4.1) shows the main screen.

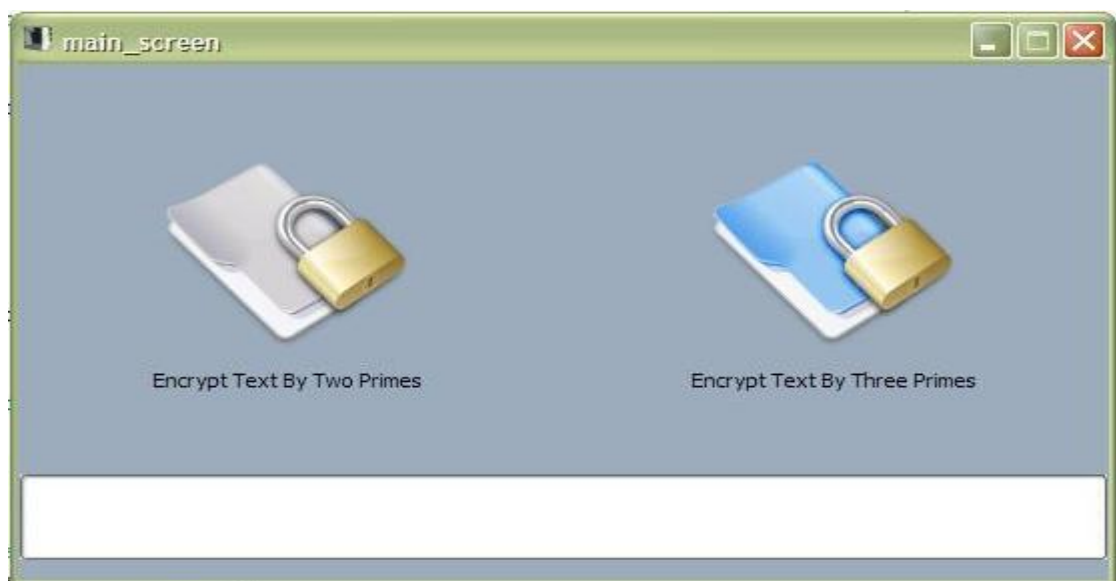


Figure 4.1: main screen

Here, we open the main screen and choose the system that have to work on it. The main system is used two primes numbers, or the proposed system which uses three primes' numbers.

Select The " Encryption Text By Two Primes "

Figure (4.2) shows in the left side, you should write the message (plain text) to be encrypted and then click on the Encrypt button.

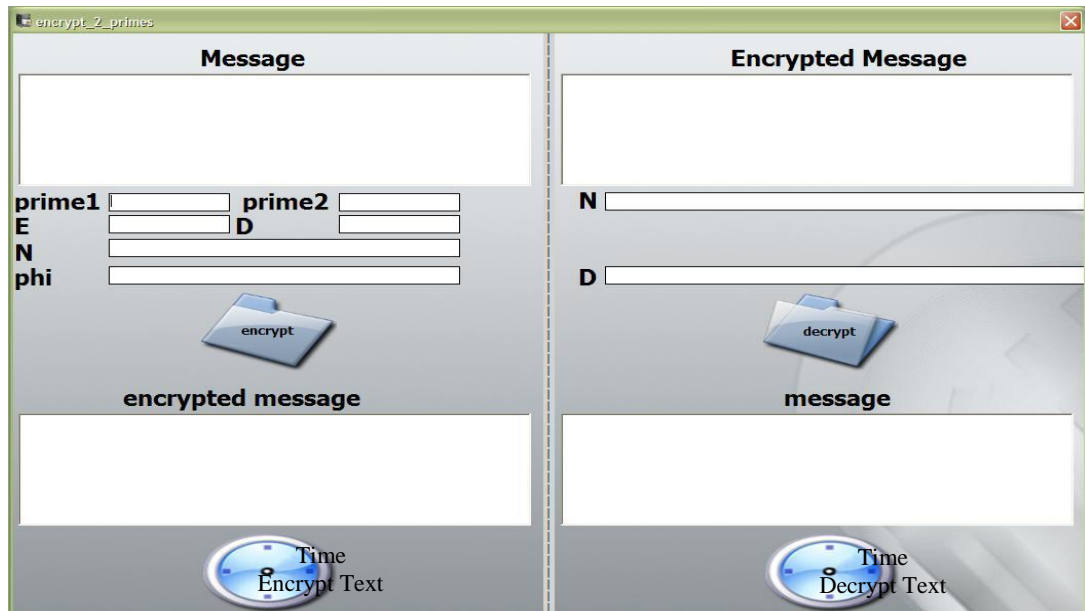


Figure 4.2: Encryption Text By Two Primes

Figure (4.3) shows the encrypted message and all the variables which are appear after inserting the two prime numbers.

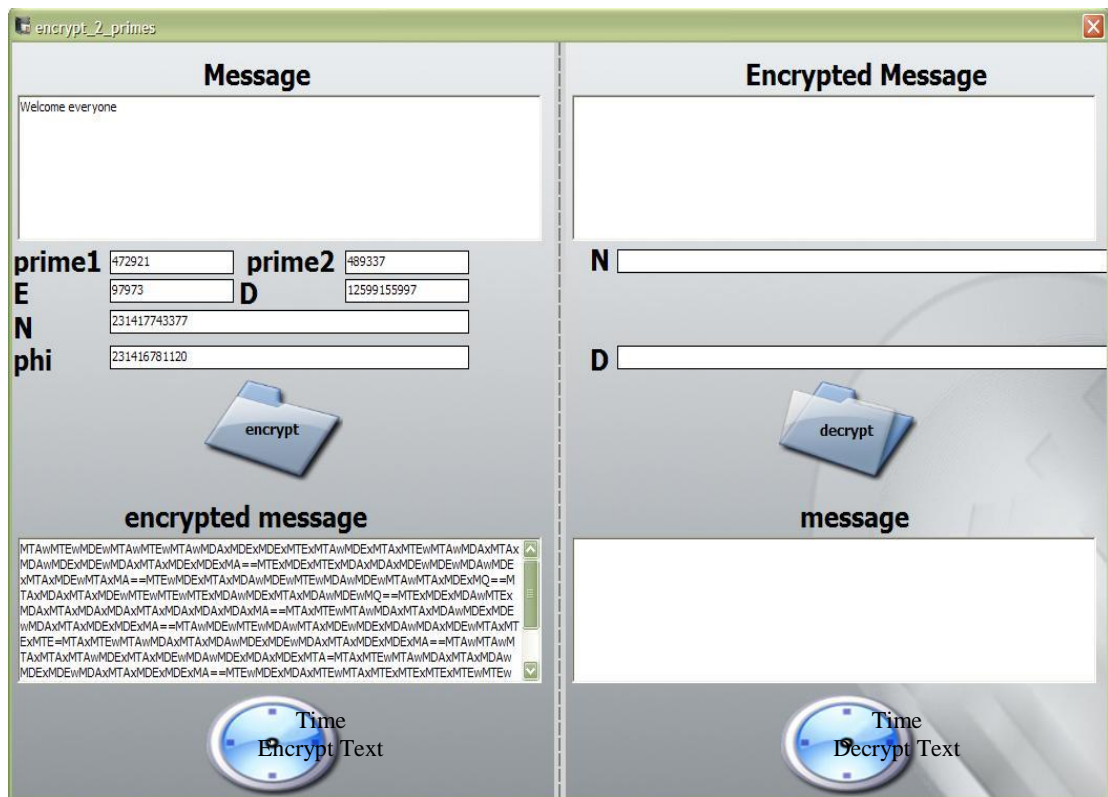


Figure 4.3: Encrypted message

Here, after we wrote the message and clicking on encrypt button, which makes the program to encrypt the message at the bottom and give us all the variables as follows:

Message = welcome everyone

Prime1 = 472921

Prime2 =489337

$N (\text{prime1} * \text{prime2}) = 231417743377$

$\text{Phi} ((\text{prime1}-1) * (\text{prime2}-1)) = 231416781120$

E (public key) = 97973

D (private key) = 12599155997

Encrypted message:

MTAwMTEwMDEwMTAwMTEwMTAwMDAxMDExMDExMTExMTA
wMDExMTAxMTEwMTAwMDAxMTAxMDAwMDExMDEwMDAxMT
AxMDExMDExMA==MTExMDExMTExMDAxMDAxMDEwMDEwMD
AwMDExMTAxMDEwMTAxMA==MTEwMDExMTAxMDAwMDEwM
TEwMDAwMDEwMTAwMTAxMDExMQ==MTAxMDAxMTAxMDEw
MTEwMTEwMTExMDAwMDExMTAxMDAwMDEwMQ==MTExMDE
xMDAwMTExMDAxMTAxMDAxMDAxMTAxMDAxMDAxMDAxMA==
MTAxMTEwMTAwMDAxMTAxMDAwMDExMDEwMDAxMTAxMDEx
MDExMA==MTAwMDEwMTEwMDAwMTAxMDEwMDExMDAwMD
AxMDEwMTAxMTExMTE=MTAxMTEwMTAwMDAxMTAxMDAwMD
ExMDEwMDAxMTAxMDExMDExMA==MTAwMTAwMTAxMTAxMT
AwMDExMTAxMDEwMDAwMDExMDAxMDExMTA=MTAxMTEwMT
AwMDAxMTAxMDAwMDExMDEwMDAxMTAxMDExMDExMA==MT
EwMDExMDAxMTEwMTAxMTExMTExMTEwMTEwMDEwMT
AxMA==MTAwMDEwMDExMDEwMDEwMDExMDAxMDExMDEwM
DEwMTAxMDEwMDE=MTAxMDAxMTAxMDEwMTEwMTEwMTEx
MDAwMDExMTAxMDAwMDEwMQ==MTEwMTAwMDExMTExMDA
xMTEwMDExMTEwMTAxMDExMDEwMTAwMQ==MTAxMTEwMTA
wMDAxMTAxMDAwMDExMDEwMDAxMTAxMDExMDExMA==

Now, we take the encrypted part of the message and place it in the right side of the screen in Figure (4.4), and we enter the value of N and private key D.



Figure 4.4: Insert Value To The Right Side
 After entering the values we are clicking on decrypt button as in Figure (4.5).



Figure 4.5: Decrypt message

Note that, the encrypted message returned to its original text form after the decrypted process has been taken place.

-- Time Encrypt text

In this part of the program, the program calculates the time from the beginning of the implementation of the program until the end of it, in other words from the moment of clicking on the Encrypt button till we received the encrypted text, see Figure (4.6).



Figure 4.6: Time Encrypt Text

This time on the screen is a time of encrypting the plain text; you will notice that the time varies from one computer to another according to its architecture, because it depends on the specifications of the device.

-- Time of Text Decryption

In this part of the program, it calculates the time from the moment of clicking on the Decrypt button till you get the Plain text, as in Figure (4.7).



Figure 4.7: Time Decrypt Text

This time on the screen is the Decrypting time for the plain text; you will notice that the time varies from one computer to another, because it depends on the specifications of the device.

Select The " Encryption Text By Three Primes "

From the main screen you can select the "Encryption Text By using Three Primes " rather than choosing " Encryption Text By using two Primes ", this will appear to us the next screen as in Figure (4.8).

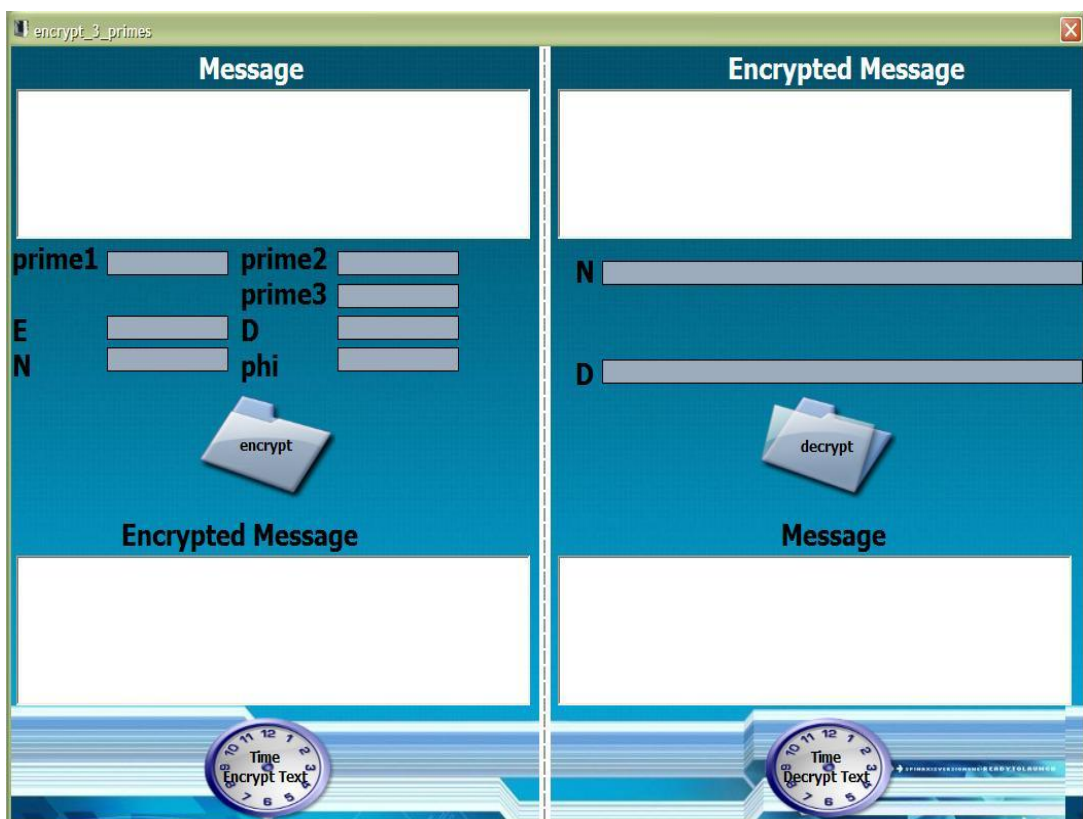


Figure 4.8:Encrypt_3_primes

This screen is an application of the proposed method for algorithm RSA, here, in the left side, you write the message you want to be encrypted and then click on the Encrypt button.

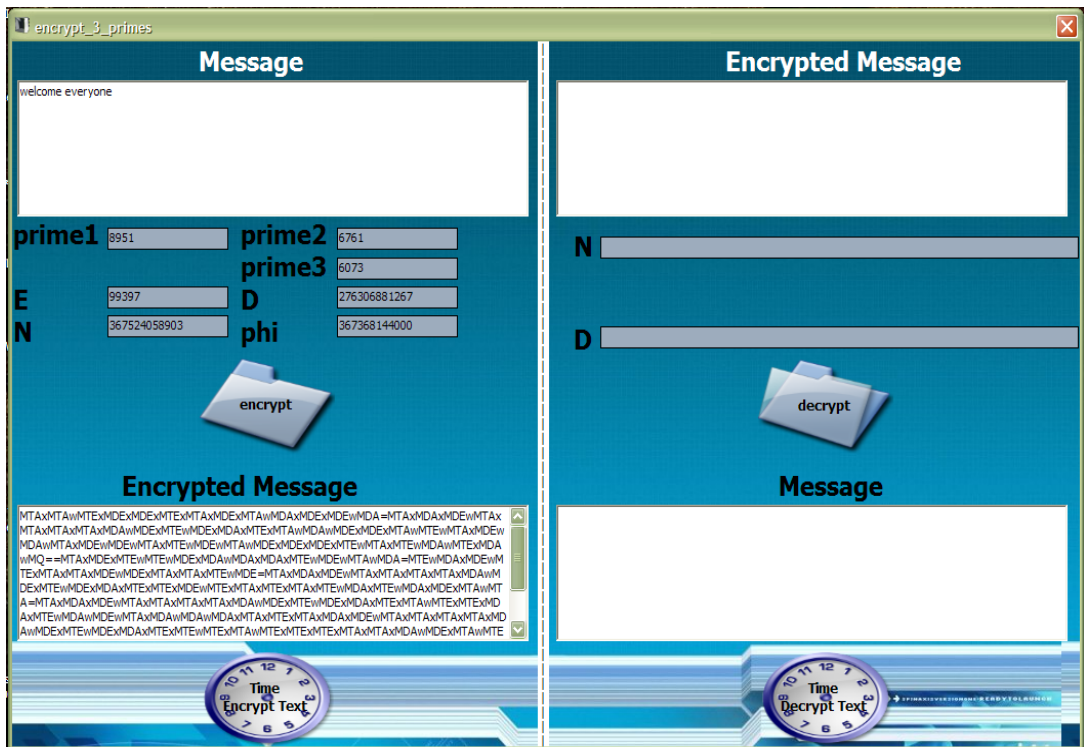


Figure 4.9: Encrypt message

In Figure (4.9) we wrote the message and clicking on encrypt button, this will activate the program to encrypt the message at the bottom and gives us all the variables as follows:

Message = welcome everyone

Prime1 = 8951

Prime2 = 6761

Prime3 = 6073

$N (\text{prime1} * \text{prime2} * \text{prime3}) = 367524058903$

$\text{Phi} ((\text{prime1}-1) * (\text{prime2}-1) * (\text{prime3}-1)) = 367368144000$

E (public key) = 99397

D (private key) = 276306881267

Encrypted message:

MTAxMTAwMTExMDExMDExMTExMTAxMDExMTAwMDAxMDEx
MDEwMDA=MTAxMDAxMDEwMTAxMTAxMTAxMTAxMDAwMDEx
MTEwMDExMDAxMTExMTAwMDAwMDExMDExMTAwMTEwMTA
xMDEwMDAwMTAxMDEwMDEwMTAxMTEwMDEwMTAwMDExM
DExMDExMTEwMTAxMTEwMDAwMTExMDAwMQ==MTAxMDExM
TEwMTEwMDExMDAwMDAxMDAxMTEwMDEwMTAwMDA=MTEw
MDAxMDEwMTExMTAxMTAxMDEwMDExMTAxMTAxMTEwMDE=
MTAxMDAxMDEwMTAxMTAxMTAxMTAxMDAwMDExMTEwMDEx
MDAxMTExMTExMDEwMTExMTAxMTExMTAxMTEwMDAxMTEw
MDAxMDExMTAwMTA=MTAxMDAxMDEwMTAxMTAxMTAxMTAx
MDAwMDExMTEwMDExMDAxMTExMTAwMTExMTExMDAxMTEw
MDAwMDEwMTAxMDAwMDAwMDAxMTAxMTExMTAxMDAxMDE
wMTAxMTAxMTAxMTAxMDAwMDExMTEwMDExMDAxMTExMTE
wMTExMTAwMTExMTExMTExMTAxMTAxMDAwMDExMTAwMTEx
MA==MTAwMDEwMDAwMTAxMTEwMTExMTAxMTAxMTAwMDEx
MDEwMTAwMTA=MTAxMDExMTEwMTEwMDExMDAwMDAxMDA
xMTEwMDEwMTAwMDA=MTAwMDAxMDEwMTEwMDAxMDAxMT
AwMDEwMDAxMTEwMDEwMTExMDA=MTAxMDAxMDEwMTAxM
TAxMTAxMTAxMDAwMDExMTEwMDExMDAxMTEx

Now, we take the encrypted part of the message and place it in the right side of the program, and we will enter a value of N and private key D as in Figure (4.10).



Figure 4.10: Insert Value To The Right Side

After we entered the private key and N values we are clicking on decrypt button as in Figure (4.11).

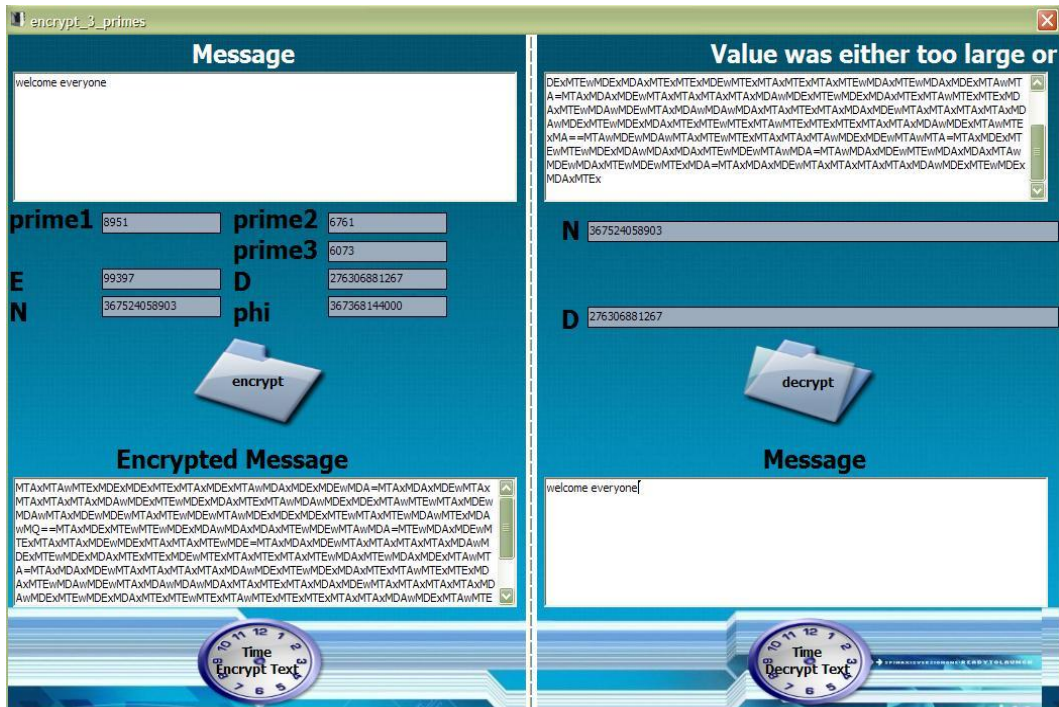


Figure 4.11: Decrypt Message

Note that, the encrypted message (Cipher Text) returned to its original format (Plain Text) that we entered.

-- Time Encrypt text

In this part of the program, the program calculates the time from the beginning of the implementation of the program until the end of it, in other words from the moment of clicking the Encrypt button till gives you an encrypted text as in Figure (4.12).



Figure 4.12: Time Encrypt Text

This time on the screen is a time of encrypting the plain text; you will notice that the time varies from one computer to another, because it depends on the specifications of the device.

-- Time Decrypt Text

In this part of the program, the program calculates the time from the moment of clicking the Decrypt button till gives you the Plain text as in Figure (4.13).



Figure 4.13: Time Decrypt Text

This time on the screen is a time of Decryption of the plain text; you will notice that the time varies from one computer to another, because it depends on the specifications of the device.

At the end, we note that the process of encryption and decryption and the way of original method and proposed method are working without any error. As well as encryption and decryption process by using the proposed method are faster, and to prove it will repeat the process of encryption and decryption over time using the two methods, and you will notice the results.

When you repeat the process of encryption and decryption of the text "welcome everyone ", we will get the following results as in Table (4.1) and Table (4.2):

Table 4.1:Encryption And Decryption By using Two Primes

Prime1	Prime2	N	phi	Public key	Private key	Time Encrypt mSec	Time Decrypt mSec
21107	21023	443732461	443690332	90481	426669641	1.11	1.32
21023	21163	444909749	444867564	90371	419456687	1.46	1.46
21179	21139	447702881	447660564	90379	191285623	1.46	1.46
21169	21169	448126561	44804224	90187	29258851	1.62	1.47
21247	21187	450160189	450117756	90053	76909841	1.62	1.47

2116 3	2127 7	4502851 51	4502427 12	9007 3	2582798 5	1.6 3	1.4 7
2113 9	2134 7	4512542 33	4512117 48	9017 3	2840783 09	1.6 3	1.4 7
2112 1	2140 1	4520105 21	4519680 00	9000 1	2739900 01	1.7 9	1.4 7
2132 3	2121 1	4522821 53	4522396 20	9008 9	2015498 09	1.9 4	1.6 2
2141 9	2139 1	4581738 29	4581310 20	9016 3	1475512 87	2.6 2	1.6 2

Table 4.2: Encryption And Decryption By using Three Primes

Prime1	Prime2	Prime3	N	phi	Public key	Private key	Time Encrypt mSec	Time Decrypt mSec
677	691	709	331675 163	33023 9520	904 01	2593 3744 1	1.09	1.3
733	829	683	415029 731	41335 7472	900 53	4764 5837	1.21	1.31

769	709	773	421455 833	41977 0368	903 79	5154 0643	1.25	1.31
677	887	751	450974 749	44920 2000	901 07	2047 4243	1.47	1.46
683	839	797	456710 489	45492 6736	904 07	2578 3895	1.47	1.47
823	821	691	466896 953	46508 7600	900 07	8719 7143	1.47	1.47
727	769	839	469053 857	46724 1984	904 39	3064 1266 3	1.62	1.47
911	757	823	567563 021	56550 3120	902 17	5341 0543 3	1.62	1.47
809	821	857	569209 973	56715 1360	900 01	4905 2376 1	1.63	1.52
907	761	919	634318 613	63209 8080	903 73	5427 3159 7	1.78	1.6

Now we will compare between the two tables and you will notice results in Figures (4.14 & 4.15):

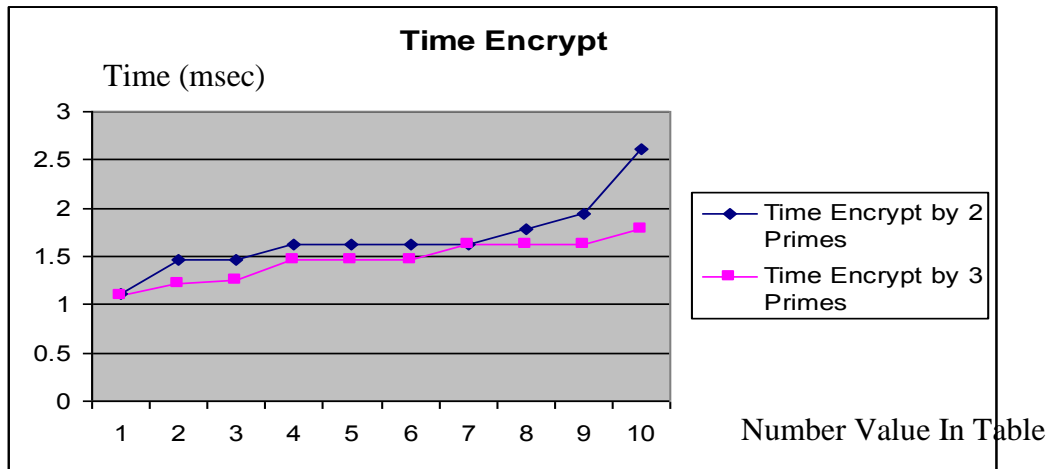


Figure 4.14: Graph Time Encrypt

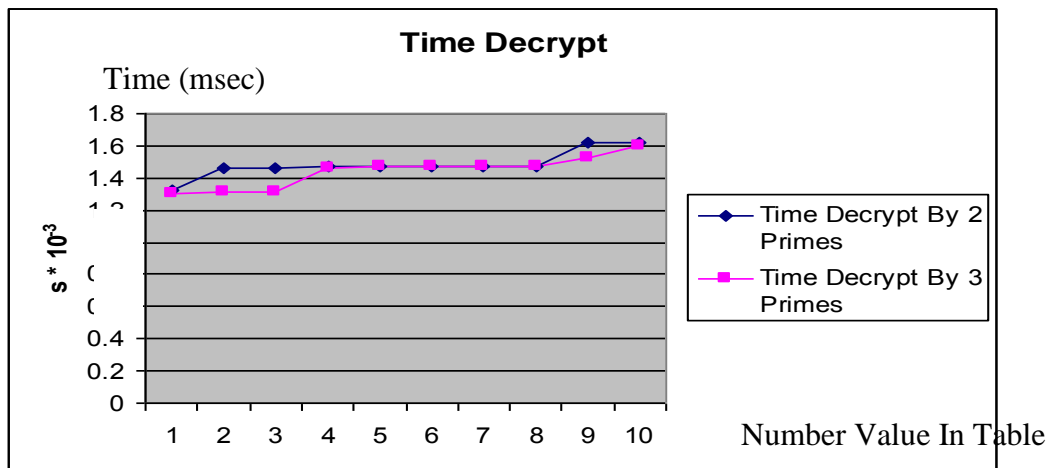


Figure 4.15: Graph Time Decrypt

From Table (4.1): Encryption and Decryption by Two Primes

Average of N = 449863773

Average of Phi (n) = 409493354

Sum of Time Encrypt = 16.88 msec

Average of Time Encrypt = 1.688 msec

Sum of Time Decrypt = 14.83 msec

Average of Time Decrypt = 1.483 msec

From Table(4. 2): Encryption and Decryption by Three Primes

Average of N = 478288838

Average of Phi (n) = 476457824

Sum of Time Encrypt = 14.61 msec

Average of Time Encrypt = 1.461 msec

Sum of Time Decrypt = 14.38 msec

Average of Time Decrypt = 1.438 msec

Now we note that the process of encryption and decryption in the proposed method are faster than the original method by the apparent results.

4.2.2.. Second application:

(Generation & Factorization of The Key, Component From 18 Digits)

-- run application : Figure (4.16) shows the main screen.



Figure 4.16: Generation and Factorization of the Key

In this program we can generate the key component of (2) primes or (3) primes number. The program will calculate each of the N, Phi, public key (E), private key (D) and find the Greatest Common Divisor (GCD) for E & Phi and the number of iterations until we find the private key. Also it can calculate the required time to generate the key, whether it consist of 2 or 3 primes, as well as we can make factorization for the variable n, whether it has been generated from 2 or 3 primes. Also it calculates the time of factorization, allows the insertion of the variable n manually, factorizes it and calculates the factorization time as in figure (4.16).

Now we will explain how the program works:-

A- Clicking on Generate Key for 2 Primes Button.

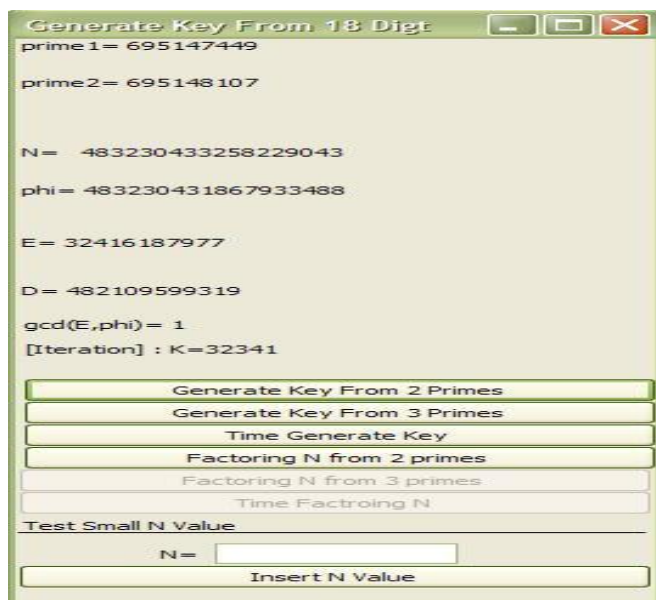


Figure 4.17: Generate Key From 2 Primes

We note that, it identified 2 primes, find N, Phi, public key

(E), private key (D), find GCD (E , Phi) and Iteration (k) till it finds the private key, and we will note that it will activates the button "Factoring from 2 Primes" as in figure (4.17).

B- Clicking on Generate Key Time Button

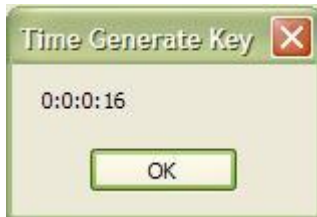


Figure 4.18 : Time Generate Key By 2 Primes

This time on the screen is a time of Generate Key from 2 Primes; you will notice that the time varies from one computer to another, because it depends on the specifications of the device see figure (4.18).

C- Clicking on Factoring N from 2 primes Button.

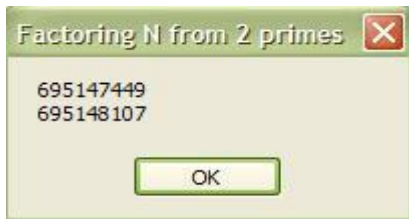


Figure 4.19: Factoring N From 2 Primes

Figure (4.19) shows the factorization of the variable n and the value of the 2 prime numbers will be displayed.

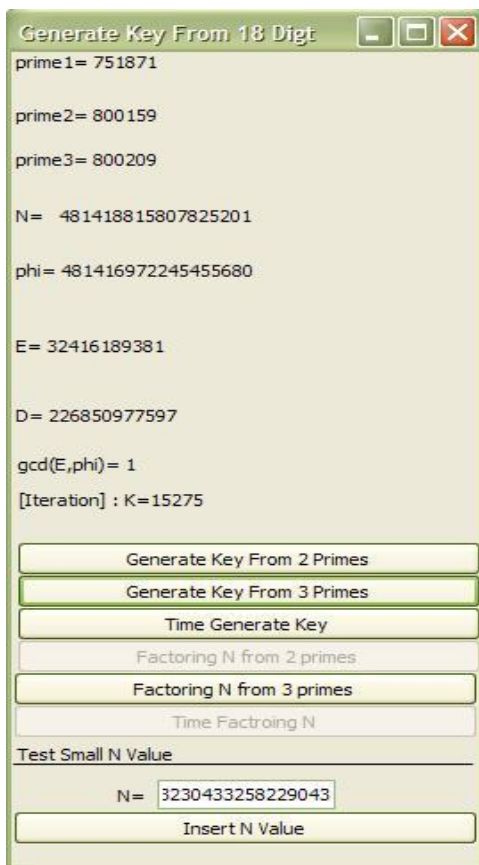
Clicking on Time Factoring N Button, the result is shown in Figure (4.20).



Figure 4.20: Time Factoring N By 2 Primes

This time on the screen is the time of Factoring N into 2 Primes; you will notice that the time varies from one computer to another, because it depends on the specifications of the device.

E- Clicking on Generate Key from 3 Primes Button



we note that identified 3 primes, find each of N, Phi, public key (E), private key(D), find GCD(E , Phi) and Iteration (k) till it find the private key, and we note that it was activating the button "Factoring from 2 Primes" as in Figure (4.21).

Figure 4.21: Generate Key From 3 Primes

F- Clicking on Time Generate Key Button

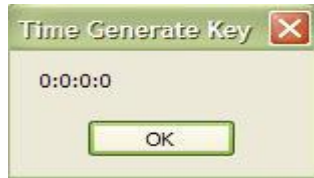


Figure 4.22 : Time Generate Key By 3 Primes

This time on the screen is the time of Generate Key by 3 Primes; you will notice that the time varies from one computer to another, because it depends on the specifications of the device as in Figure (4.22).

G- Clicking on Factoring N from 3 primes

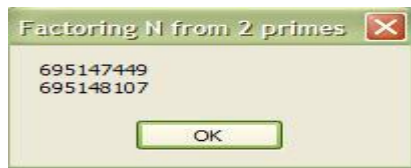


Figure 4.23: Factoring N From 2 Primes

Here is the analysis of the variable n and the program to find primes numbers of its constituent as in Figure (4.23).

Clicking on Time Factoring N Button

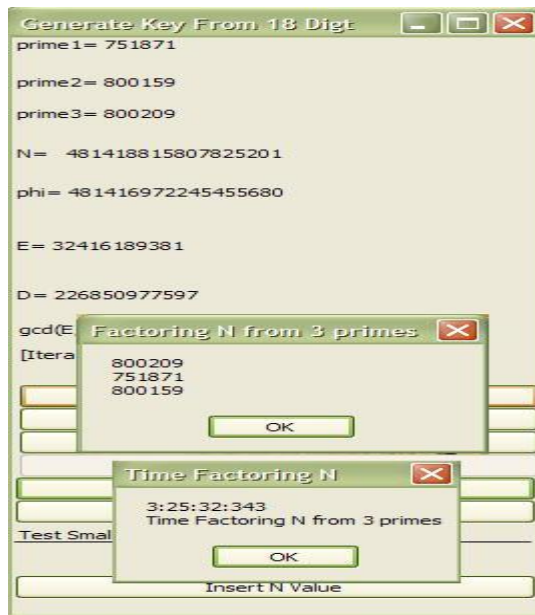


Figure 4.24: Time Factoring N From 3 Primes

This time on the screen is a time of Factoring N from 3 Primes; you will notice that the time varies from one computer to another, because it depends on the specifications of the device as in Figure (4.24).

We'll show some examples for the factorization of the variable n and the time difference of factorization as in Figure (4.25).

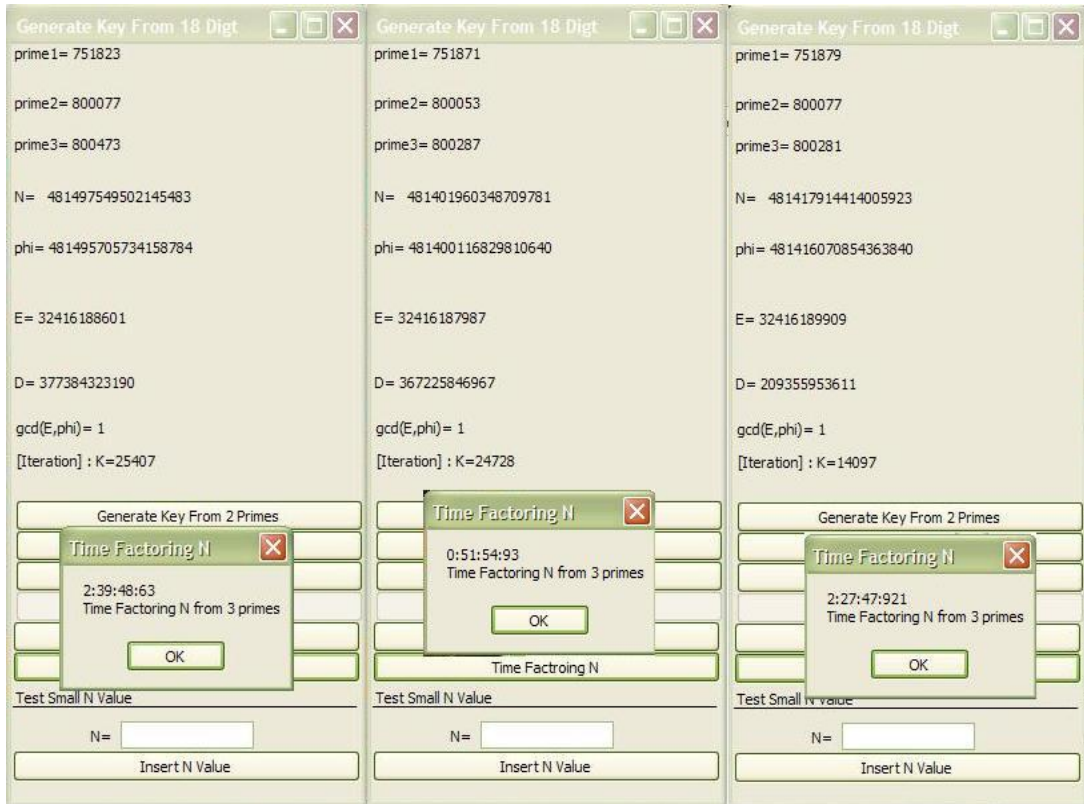


Figure 4.25: Some Examples For The Factorization N From 3 Primes

In this program we can also manually enter the variable n and we have the process of factorizing whether 2 Primes or 3 Primes and the program can give you the time for the factorization process as in figure (4.26).

Example on 2 primes:
N = 35

Example on 3 primes:
N = 105



Figure 4.26: Factoring And Time To N=35 From 2 Primes Figure 4.27: Factoring And Time To N = 105 From 3 Primes

At the end, we note that the process of generate key and factorization for variables N and the way of original method and proposed method it is working without any error. As well as the factorization using the proposed method is faster, and to prove it, it will repeat the process of factorization using two methods, and you will notice the results, see Figure (4.26).

When you repeat the process of Generation and factorization of the variables N, you will we get the following results as in Tables (4.3 & 4.4):

Table 4.3: Generation And Factorization Key From 18 Digits By 2 primes

Prime1	Prime2	N	Phi	Public key	Private key	Time generate key mSec	Time factorizing key mSec
6951 4789 1	6951 4616 3	48322 93891 46192 000	483229 387755 898000	3241 6188 191	1133 9337 5007 1	0.15	2,858
6951 4661 9	6951 4834 7	48323 00231 20488 000	483230 021730 193000	3241 6190 071	1832 5246 2554	0.16	3,455
6951 4675 7	6951 4828 7	48323 00773 42155 000	483230 075951 860000	3241 6189 987	1170 3383 4167 8	0.2	4,125

		48323					
6951	6951	11409	483231	3241	3415		
4822	4834	19030	139528	6182	9603		
7	7	000	734000	61	0093	0.31	4,812

Table 4.4: Generation And Factorization Key From 18 Digits By 3 primes

Pri me 1	Pri me 2	Pri me 3	N	Phi	Pub lic key	Priva te key	Time generat e key mSec	Time factorin g key mSec
75	80	80	4814 0196 0348	481400	324 161	3672		
18	00	02	7090	116829	879	2584		3,114.9
71	53	87	00	810000	87	6967	0.15	30
75	80	80	4814 1791 4414	481416	324 161	2093		
18	00	02	0050	070854	899	5595		8,867.9
79	77	81	00	363000	09	3611	0.16	20

			4814					
			1881		324			
75	80	80	5807	481416	161	2268		
18	01	02	8250	972245	893	5097		9,588.6
71	59	09	00	455000	81	7597	0.16	30
			4814					
			9754		324			
75	80	80	9502	481495	161	3773		
18	00	04	1450	705734	886	8432		12,332.
23	77	73	00	158000	01	3190	0.31	340

Now we will compare between the two tables and you will notice the following results:

The values of the variable N in the two tables are almost equal.

Total times of generation key in the first table = 0.82s and Total times of generation key in the second table = 0.78s, we note that the process of generating the key in the second table is faster than the first table.

Total times factorization of the key in first table = 15.250s and Total times factorization the key in second table = 33,903.820s, we note that the total time of the factorizing of the variable n in the first table is much faster than in the second table.

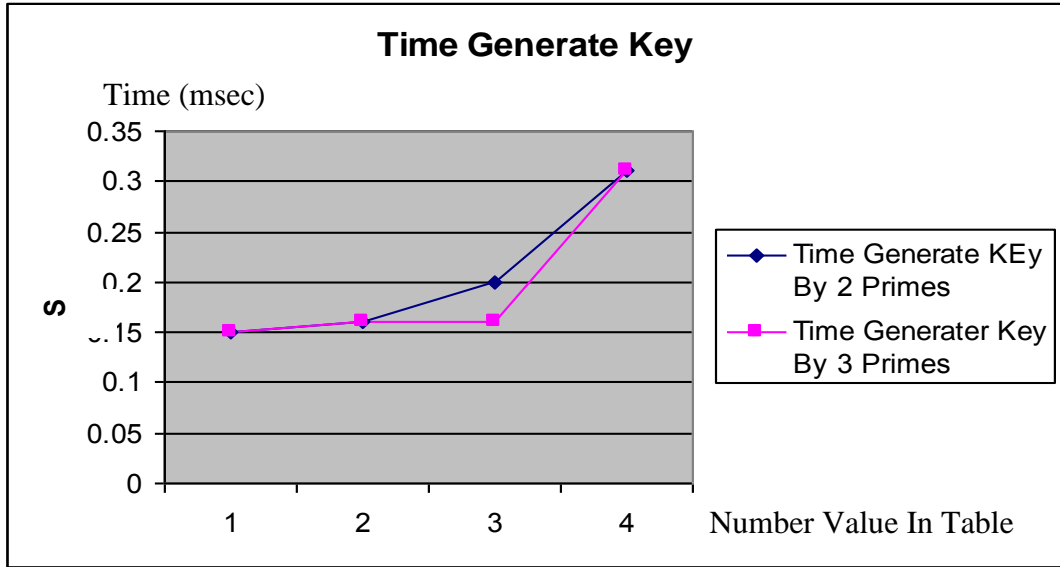


Figure 4.28: Time Generate Key

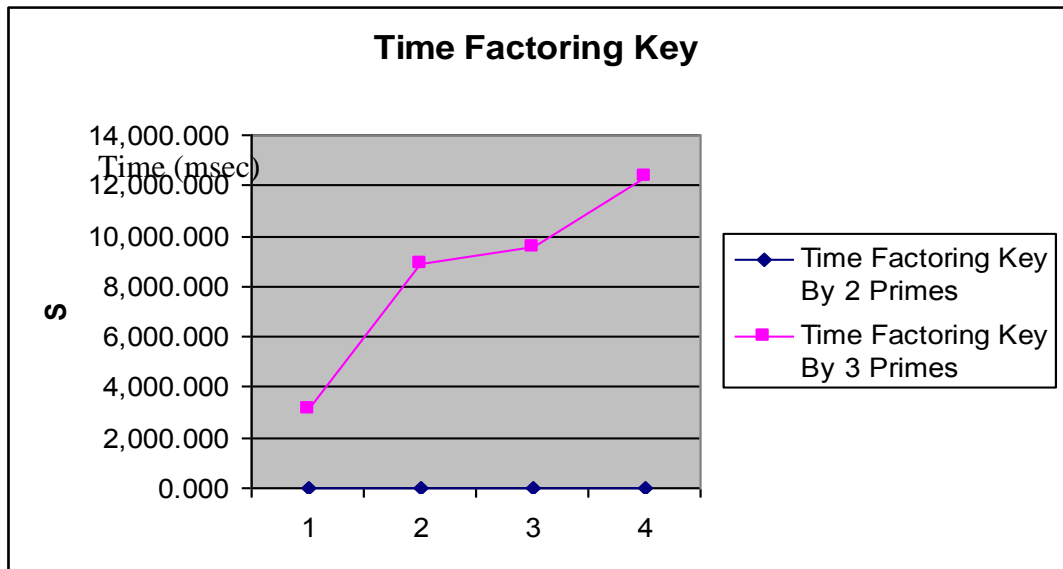


Figure 4.29: Time Generate Key

Note from the Figure (4.28) that the process of generating the key to be faster in the proposed method because the primes numbers consisting of the variable N be of little value either in the original method The primes numbers are much larger than the size of the primes numbers in the proposed method, therefore, the multiplication in the proposed method is easier and faster either in the original method is more complex and require more time, and this explains the speed of generating the key in the proposed method about the original method.

Summary of Programs

This paragraph displays a table comparison between the original method and the proposed method

Table 4.5: Comparison Between The Original Method And The Proposed Method

	Original method	Proposed method
speed Generation Of Keys	Slower	Faster
Speed Encryption	Slower	Faster
Speed Decryption	Slower	Faster
Speed Factorization	Faster	Slower
Size Message	No Different	No Different

Summary

Now we can summarize the results of the proposed method that the process of encryption and decryption is faster than the original method by the apparent results.

Also, we can summarize the results of the second program that the process of generating the key in the proposed method is faster than generating the key in the original method. As well as the process of factorizing of the variable n in the first table is faster than the process of factorizing of the variable n in the second table, and this shows that security in the proposed method is more stronger than the security in the origin method.

Chapter Five

Conclusion and Future Works

5.1. Formalistic Appraising

With this proposed method, the goal is to fulfill an efficient way to find a way to increase the protection and safety, speed and accuracy of the RSA algorithm to protect data and user-specific information. We provide the speed and protection at the same time by the proposed method and we have proven that in theory and practice.

Initially, the proposed method have been implemented with lengths 12 - 19 digits of n and with an unpretentious specifications of the computer, the results are Encryption and decryption of data in more speed and greater protection with the same precision.

It's important to specify the used personal computer which has been used during the study; its specification as the following:

CPU =E7400 (2.80 GHz, 3 MB)

RAM: 3 GB (System memory).

For that - as it is earlier mentioned - everybody has to know the operations of breaking an overall security system needs special tools and devices that luxuriate with high level of performance, specialized and ability of tolerance. So if somebody likes to break a security system; using a normal tools and devices such as personal computer

that is absolutely not enough and no way to get a good results using these tools and device. Also, there is basic difference between somebody who tries to make a gap in security system and that who tries to break the overall system.

5.2. Conclusion

The encryption system of the most important operations of the protection of information and data for the user was proposed coding systems currently used during the thesis. We have studied one of the encryption systems that are very important and used in our daily lives, RSA System. We have studied the strengths and weakness in the RSA algorithm. We have focused in this thesis on the analysis process of the variable n and we've developed more complex analysis for the variable n .

We can sum up the conclusions in the following points:

The proposed method has succeeded in increasing the difficulty of analysis the variable N .

Speed in the process of generating keys. As it previously appeared in chapter 4, especially when talking about an assumed machine with currently times, and where it's a personal computer in overall. The main specifications of machines that will actuate the proposed methods have to be apposite – at least – such as the CPU speed and RAM capacity, because the main specifications – if we want to forget an applications and utilities that runs within an operating system - plays an important role of manipulations and calculations speed, where everybody know what significance of the time.

Note through the process of generating the keys which is faster, it increases the speed of the process of encryption and decryption, and this means the provision of time on the user.

3.5. Future Works

We would like to say - after all have been mentioned - that the process of encryption and decryption of data must be highly confidential and precise so as not to break the data in any way to become more superior and efficient than the troublemakers we have to work to keep up with their ideas and simulate their ways of thinking. This way is one of the main stages that leads to wards building a solid algorithm to protect a system.

After all that, there are many and many ideas that could be applied and get best results, such as the following:

Working to find method for generating the variable n without doing any calculation or this calculations will be through a complex equations so it is not easy to analyze the variable n and this is considered to be one of the weakest points in RSA algorithm.

With beamy development of networks and the speed of the networks, it is possible to exploit the smart neural network to engage it with the proposed thesis to produce an integral protection system from any threat.

Also, the genetic algorithm (GA) could be used to support the protection functions of variant security algorithm and to make these systems is harder to break or attack, because as it is known, GA is a search technique used in computing to find exact or approximate solutions to optimization and search problems. At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for

the next generation. Over successive generations, the population "evolves" toward an optimal solution. You can apply the genetic algorithm to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, no differentiable, stochastic, or highly nonlinear. So, this technique is wonderful idea to exhort the experts to produce new generations of developed RSA algorithms.

Additionally, the development of technologies such as tools and machines make the challenges nearest to solution and any problem got solvable and nothing impossible, because as we say, what is available in these days of tools and technologies, it is not available in last days, also it's true if we say about the next days. As we know, the ideas have been breed seriatim with technology. But the main thing is exploit every thing towards enhancing and making world better.

References

- "Cryptography and network security" , Stallings, William, fourth edition, ISBN: 0-13-187316-4, 2005.
- "Applied Cryptography And Data Security", Paar, Christof, version 2.5, January 2005.
- "Introduction algorithms", Cromen, Thomas and others, second edition, ISBN: 0-262-03293-7, 2003.
- "Technology of information security", Al-Hamami Alaa and al ani saad, first edition, dar wa'ael for publishing, 2007.
- "Applied Cryptography; Second Edition", John Wiley & Sons, ISBN 0-471-11709-9, 1996.
- <http://www.rsa.com/> access date "17-10-2010 , 20:30 pm"
- "Efficient method for breaking RSA scheme", Sattar j Abboud and Mohammad a AL-Fayoumi, Ubiquitous Computing and Communication journal, 2008.
- "Factorization of RSA-170", Dominik Bonenberger and Martin Krone.; Revision 2010-03-08.
- "Factorization of a 512(bit RSA Modulus)", Stefania Cavallar and Bruce Dodson,1999.
- <http://x5.net/faqs/crypto/q10.html> access date "18-9-2010 , 16:15 pm"
- "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Rivest, R.; A. Shamir; L. Adleman (1978).
- "Greek-English Lexicon", Liddell and Scott's. Oxford University Press,1984.

"Body of Secrets: Anatomy of the Ultra-Secret National Security Agency from the Cold War through the Dawn of a New Century"
Bamford, James, New York, Doubleday, 2001.

<http://www.bga.org/~lessem/psyc5112/usail/tasks/security/security.html#denial> "19-9-2010 , 18:07 pm".

"cryptography overview ",W. M. Farmer , 2003.

"Cryptography: A guide to protecting your files for consultants, educators and researchers", Kirk Job-Sluder Indiana University.
"IST_Conf_2002_sluder.pdf" , 2002.

"A "fob" is defined as a ,A short chain or ribbon attached to a pocket watch and worn hanging in front of the vest or waist"
(<http://dictionary.reference.com/search?q=fob>). access date "22-9-2010 , 19:05 pm".

http://www.ridex.co.uk/cryptology/#_Toc439908857 access date "11-01-2011 , 18:50 pm".

"National computer conference .Advanced encryption standard",
Khan, Mansoor , Zafar Dawood and Abdul Raouf, 2002.

"Cryptography and Network Security, s,l : McGraw-Hill Forouzan",
Behrouz A, ISBN:0073327530 , 2007.

"Strength Assessment Of Encryption Algorithms", Limor Elbaz &
Hagai Bar-El , Discretix Technologies Ltd, October ,2000.

"Attacks On the RSA Cryptosystem", Dr. Kartik Krishnan, Xiao-lei Cui,2005.

"Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems" ,P. KOCHER, CRYPTO '96, Lecture Notes in Computer Science, pp. 104–113, 1996,.

"Twenty Years of Attacks on the RSA Cryptosystem", Dan Boneh,1999.

"Handbook of Applied Cryptography" A. Menezes, P. van Oorschot, and S. Vanstone, CRC, 1996.

Cryptanalysis of short RSA secret exponents, M. WIENER, IEEE Trans. Inform. Theory 36,PP. 553–558, 1990.

"Solving simultaneous modular equations of low degree" J. HASTAD, SIAM J. Compute, PP. 336–341,1988.

"In CRYPTO '96, volume 1109 of Lecture Notes in Computer Science", P. e - Hellman, RSA, DSS, and other systems, PP. 104-113, 1996.

"On the importance of checking cryptographic protocols for faults. In EUROCRYPT '97, volume 1233 of Lecture Notes in Computer Science", D. Boneh, R. DeMillo and R. Lipton, PP. 37-51, 1997.

"Mathematical Attacks on RSA Cryptosystem" , Imad Khaled Salah and Abdullah Darwish , Amman Arab University for Graduate Studies, , 2006.

"Donald Knuth. The Art of Computer Programming, Volume 2: Semi numerical Algorithms", Addison-Wesley, Third Edition, ISBN 0-201-89684-2, pp. 379–417,1997.

"Prime Numbers: A Computational Perspective", Richard Crandall and Carl Pomerance ,first edition, Springer. ISBN 0-387-94777-9, pp. 191–226, 2001.

"Experience in Factoring Large Integers Using Quadratic Sieve", D. J. Guan, National Sun Yat-Sen University, April 19, 2005.

"A Tale of Two Sieves" , Carl Pomerance , December 1996.

"On quadratic polynomials for the number field sieve", B. Murphy and R. P. Brent. Australian Computer Science Communications , pp. 199–213,1998.

"Introduction to analytic and probabilistic number theory", G. Tenenbaum, ISBN 0-521-41261-7, 1995.

"On Using RSA with low exponent in a public key Network", Håstad, J, Advances in Cryptology –CRYPTO '85, Springer-Verlag LNCS 218, pp. 403-408, 1986.

"Attack on the Cryptographic Scheme", Coppersmith , D. Advances in Cryptology–CRYPTO ,Springer, pp.294-307,1994.

"Key Distribution Protocol for Digital Mobile Communication Systems", Tatebayashi, m. Matsuzaki and Newman Jr. Advances in Cryptology –CRYPTO '89, Springer-Verlag LNCS 435, PP. 324-334, 1990.

"Joint Encryption and Message Efficient Secure Computation", Frankin, M.Haber, S.Advances in Cryptology–CRYPTO '93, Springer-Verlag LNCS 773, pp. 266-277, 1994.

"Another Birthday Attack", Coppersmith, D. Advances in Cryptology–CRYPTO '85, Springer-Verlag, LNCS 218, PP 14-17, 1986.

"RSA Encryption", Tom Davis , October 10, 2003.

"Combinatorial Proof of Fermat's "Little" Theorem" , S. W. Golomb ,
The American Mathematical Monthly, Vol. 63, No. 10, pp. 718, Dec,
1956.

"A Mechanical Proof of the Chinese Remainder Theorem" , David M.
Russino, 1997.

"RSA THEORY" , avid Ireland, Last updated: 5 May 2007. This Latex
version first published: 2 October 2006.

"Riemann's Hypothesis and Tests for Primality". Gary L.
Miller, Journal of Computer and System Sciences , 1976.

"Step-wise calculation of Performance and Complexity Analysis of
Safer with RSA Algorithm" , Amar Mohapatra , Dr. Nupur Prakash,
University School of Information Technology.

"CRYPTANALYSIS OF RSA USING ALGEBRAIC AND LATTICE
METHODS" , Glenn Durfee, stanford university , 2002.

"C# Language Specification" (4th ed.). Ecma International. June
2006. Retrieved June 18, 2009.

