جـامـعـة الـشـرق الأوسـط
MIDDLE EAST UNIVERSITY
Amman - Jordan

A Signcryption Approach Based on Rabin Digital Signature Schemes

التوقيع والتشفير القائم على التوقيع الرقمى رابين

Prepared By

Ali Mohammad Alzeinat

Supervisor

Dr. Ahmad Abu-Shareha

Thesis Submitted In Partial Fulfillment of the Requirements for the Degree of Master of Computer Science

Department of Computer Science

Faculty of Information Technology

Middle East University

June- 2017

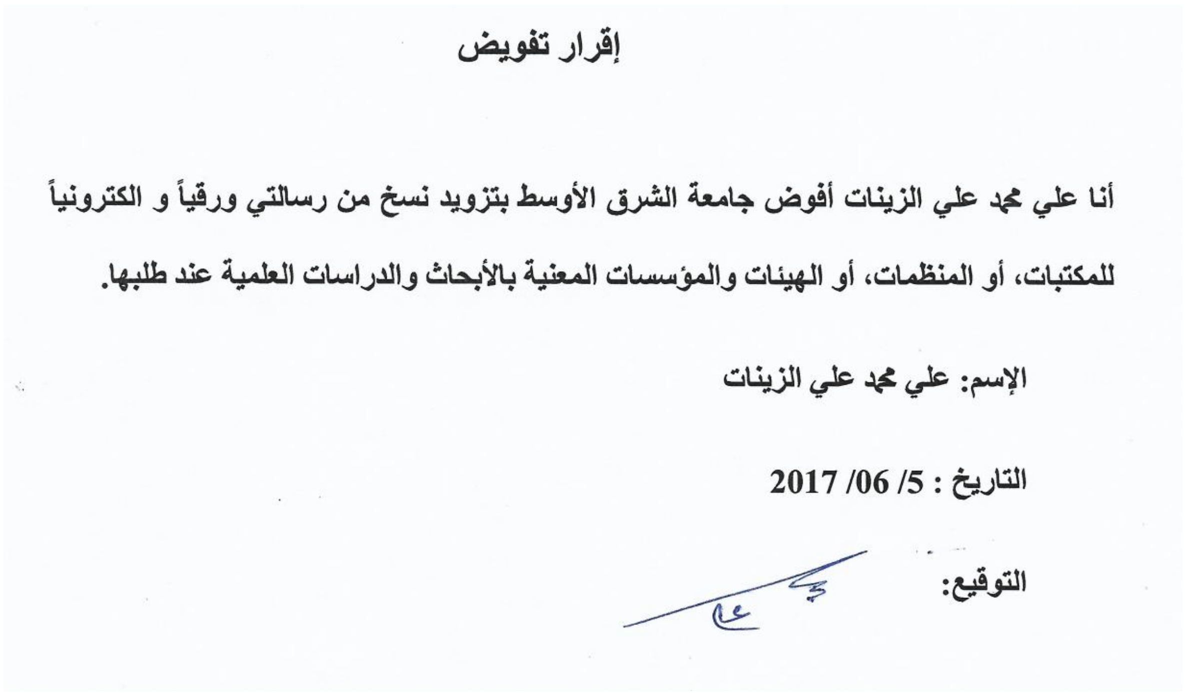
Authorization Statement

I, Ali Mohammad Ali Alzeinat, authorize the Middle East University to provide hard copies or soft copies of my Thesis to libraries, institutions or individuals upon their request.

Name: Ali Mohammad Ali Alzeinat

Date: 5/ 06/ 2017

Signature: [illegible]



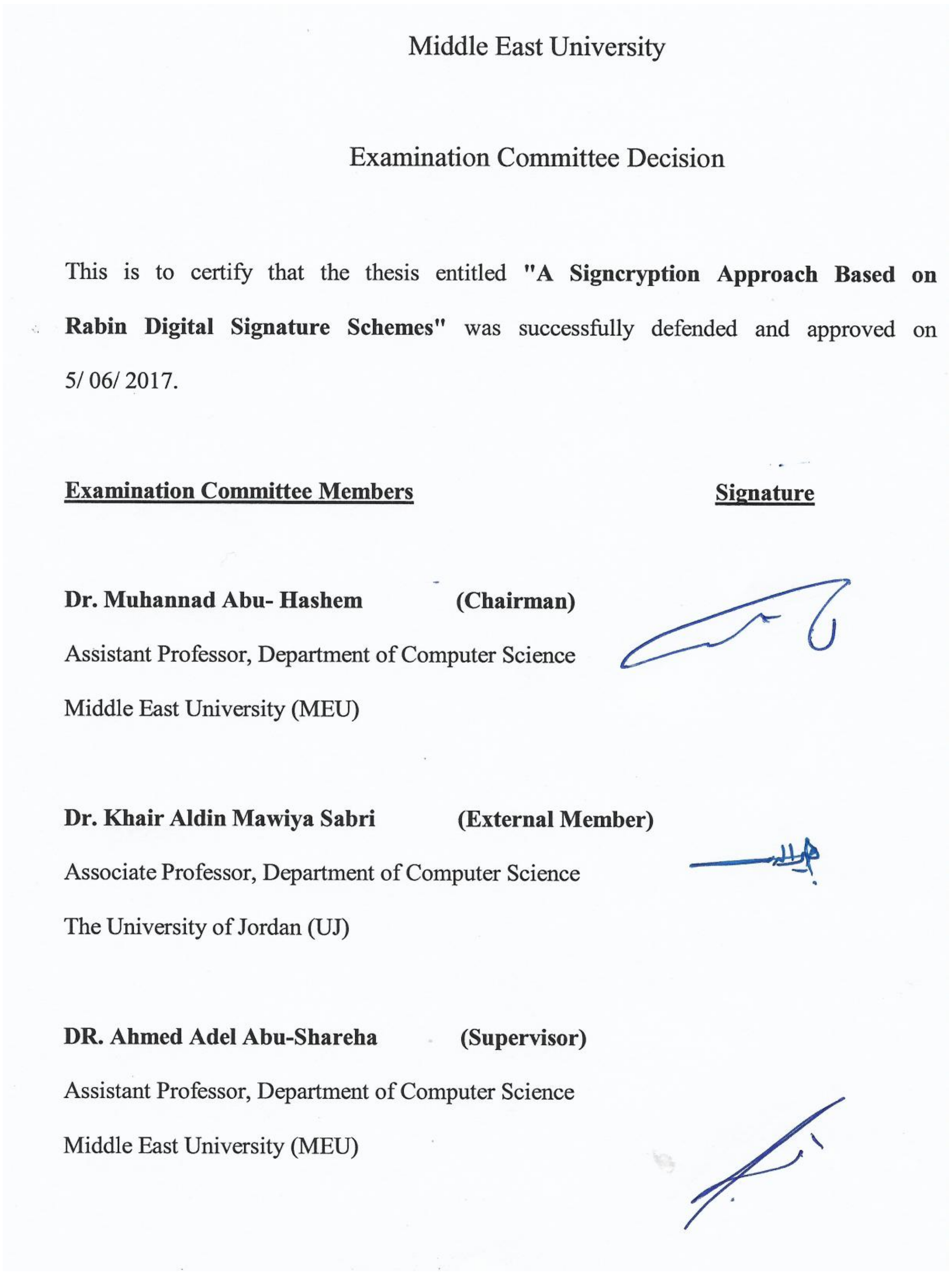
إقرار تفويض

أنا علي محمد علي الزينات أفوض جامعة الشرق الأوسط بتزويد نسخ من رسالتي ورقياً و الكترونياً للمكتبات، أو المنظمات، أو الهيئات والمؤسسات المعنية بالأبحاث والدراسات العلمية عند طلبها.

الإسم: علي محمد علي الزينات

التاريخ : 5/ 06/ 2017

التوقيع: [illegible]



Middle East University

Examination Committee Decision

This is to certify that the thesis entitled **"A Signcryption Approach Based on Rabin Digital Signature Schemes"** was successfully defended and approved on 5/ 06/ 2017.

Examination Committee Members **Signature**

Dr. Muhannad Abu- Hashem **(Chairman)** [illegible]

Assistant Professor, Department of Computer Science

Middle East University (MEU)

Dr. Khair Aldin Mawiya Sabri **(External Member)** [illegible]

Associate Professor, Department of Computer Science

The University of Jordan (UJ)

DR. Ahmed Adel Abu-Shareha **(Supervisor)** [illegible]

Assistant Professor, Department of Computer Science

Middle East University (MEU)

Acknowledgments

First, I give thanks, praise to Allah for his mercy, and reconcile and for granting me knowledge, confidence, patience to pass this Master thesis successfully.

Also, I would like to express my thank to my thesis advisor. **DR. Ahmed Adel Abu-Shareha** for the complete guidance throughout the thesis stages, and for the critical assistance in designing and proceeding the methodology of my research.

Finally, I would like to acknowledge my friends **Mustafa Awad Farhan** and **Qasem Abu Jabal** who have supported me through entire process. I will be grateful forever for your love.

Dedication

Table of Content

List of Tables

List of Figures

List of Symbols

Symbol	Description
ADD	The Number of Additions or Subtractions
C	Cipher-Text
D	Decryption Exponent
DEC	The Number of Decryption
DIV	The Number of Division Operations
E	Encryption Exponent
ENG	The Number of Encryption Operations
EXP	The number of Exponentiation Operations
HASH	The Number of Times the One Way or Keyed Hash Function Is Implemented
M	Plain-Text
MUL	The Number of Multiplication Operations
N	Public Key
P,Q	Private Keys
S	Signature

List of Abbreviations

Abbreviations	Meaning
AES	Advanced Encryption Standard
ATM	Automated Teller Machine
CDH	Computation Diffie-Hellman
CRT	Chinese Reminder Theorem
DEM	Data Encapsulation Mechanism
DES	Data Encryption Standard
DS	Digital Signature
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
EF-IDSCACMA	Unforgeability Against Adaptive Chosen Messages Attacks
GDH	Gap Diffie Hellman
H(.)	Hash Function
ID	Identity
K	Key
KEM	Key Encapsulation Mechanism
M	Message

MAC	Message Authentication Code
PKC	Public Key Cryptography
PrK	private key
PSS	Probabilistic Signature Scheme
PuK	public key
RSA	Rivest, Shamir, and Adleman
RSA-DS	RSA Digital Signature
SDSS	Shortened Digital Signature Scheme
WSN	Wireless Sensor Network

A Signcryption Approach Based on Rabin Digital Signature Schemes

Prepared By

Ali Mohammad Alzeinat

Supervisor

Dr. Ahmad Abu-Shareha

Abstract

Signcryption, which was invented by Yelling Zheng in 1997, is a recent security approach that is used to secure the communication between two or more communicated parties with low computational cost compared to sign-then-encrypt. Signcryption is a public key cryptography, which depends on performed digital signature and public key encryption in a single logical step. Signcryption has been implemented using various signature schemes such as ElGamal's shortened digital signature schemes, Proxy signature scheme, RSA digital signature, elliptic curve digital signature and Schnorr's signature scheme. Researchers are still working to find out the most effective method for signcryption. In this thesis, a new efficient signcryption approach based on Rabin cryptosystem is proposed. Rabin public cryptography is one of the earlier mechanisms, which is characterized with a smaller computational cost that depends on factorization. Rabin has the property of being difficult to find the private keys. This work represents a new signcryption approach based on Rabin digital signature, and the proposed approach is verified against the security concerns in addition to measuring the running time.

Rabin Cryptosystem has a problem of decryption as it gives four results instead of one, which means that only one of them represents the original message. Chinese Reminder Theorem (CRT) is used to get the solution in Rabin cryptosystem decryption. This process is critical in the proposed Rabin signcryption.

In this thesis, the Proposed Signcryption based on Rabin cryptosystem is implemented and the execution time for signcryption and unsigncryption is measured. Then, there is a comparison between the Proposed Signcryption and the original signcryption in terms of effectiveness. The Proposed Signcryption satisfied the security concerns: confidentially, unforgeability, integrity, non-repudiation and forward secrecy. The security concerns have been proven based on the factorization problem.

Compared to the original signcryption (SDSS1 & SDSS2) the Proposed Signcryption is achieve more effectiveness for signcryption process in the sender side at a rate of 24.1%. But in the unsigncryption process the Proposed Signcryption is achieve less effectiveness at a rate of 18.3%. In full execution time the Proposed Signcryption is achieve more effectiveness at a rate of 3%. By taking advantage of the signcryption speed on sender side the Proposed Signcryption is suitable for Smart Cards and Wireless Sensor Network (WSN).

Key Words: Cryptography, Signcryption, Public Key Cryptography, Digital Signature, Signature-then-Encryption, Discrete Logarithm problem, Factorization problem, Rabin Cryptosystem, Authentication, Forward Secrecy.

التوقيع والتشفير القائم على التوقيع الرقمى رابين

إعداد

علي محمد الزينات

إشراف

الدكتور أحمد أبو شريح

المُلخص

التوقيع والتشفير، الذي أُخترع من قِبل يلنج تشنغ في عام 1997، هو نهج أمني حديث يستخدم لتأمين الإتصال بين إثنين أو أكثر من الأطراف المتصلة مع تكلفة حسابية منخفضة بالمقارنة مع التوقيع ثم التشفير. التوقيع والتشفير هي من نوع تشفير المفتاح العام التي تعتمد على تنفيذ التوقيع الرقمي والتشفير بالمفتاح العمومي في خطوة منطقية واحدة. التوقيع والتشفير تم تنفيذه باستخدام أساليب التوقيع المختلفة مثل مخطط الجمال للتوقيع الرقمي المختصر, مخطط توقيع الوكيل, التوقيع الرقمي (RSA), التوقيع الرقمي منحنى الإهليلجي ومخطط التوقيع الرقمي شنور. الباحثون لا يزالون يعملون على إيجاد الطريقة الأكثر فعالية لعملية التوقيع والتشفير. في هذه الأطروحة، تم اقتراح نهج جديد وفعّال للتوقيع والتشفير على أساس رابين. التشفير العام رابين هو أحد الآليات السابقة التي تتميز بتكلفة حسابية صغيرة التي تعتمد على التحليل الى عوامل. رابين لديه خاصية صعوبة العثور على

المفاتيح الخاصة. ويمثل هذا العمل نهجا جديدا للشفرات يعتمد على التوقيع الرقمي رابين، النهج المقترح تم التحقق منه ضد المخاوف الأمنية بالاضافة الى قياس وقت التشغيل.

التشفير " رابين" لديها مشكلة في عملية فك التشفيرلانها تعطي اربع نتائج بدلا من نتيجة واحدة, مما يعني أن واحدة منهم فقط تمثل الرسالة الأصلية نظرية التذكير الصينية (CRT) تستخدم للحصول على الحل في فك تشفير خوازمية رابين. هذه العملية حرجة في التوقيع والتشفير المقترح رابين.

في هذه الأطروحة، التوقيع والتشفير المقترح على أساس خوارزمية رابين تم تنفيذه، ووقت التنفيذ للتشفير وفك التشفير تم قياسه. ثم، هناك مقارنة بين التوقيع والتشفير المقترح والتوقيع والتشفير الأصلي من حيث الفعالية. التوقيع والتشفير المقترح حقق الإهتمامات الأمنية: السرية، وعدم التزوير، والنزاهة، وعدم التنصل، والسرية إلى الأمام. الاهتمامات الامنية تم اثباتها بناء على مشكلة العوامل.

بالمقارنة مع التوقيع والتشفير الأصلي (SDSS1 و SDSS2) التوقيع والتشفير المقترح يحقق فعالية أكبر في عملية التوقيع والتشفير في جانب المرسل بمعدل 24.1%. لكن في عملية فك التوقيع والتشفير, التوقيع والتشفير المقترح يحقق فعالية أقل بمعدل 18.3%. وفي وقت التنفيذ كاملا،

التوقيع والتشفير المقترح يحقق فعالية أكبر بمعدل 3٪. من خلال الاستفادة من سرعة التوقيع والتشفير في جانب المرسل التوقيع والتشفير المقترح مناسب للبطاقات الذكية وشبكة الاستشعار اللاسلكية (WSN).

الكلمات المفتاحية: علم التشفير، التوقيع والتشفير، التشفير المفتاح العام، التوقيع الرقمي، التوقيع ثم التشفير، مشكلة لوغاريتم منفصلة، مشكلة التحليل الى العوامل، تشفير رابين، المصادقة، السرية إلى الأمام.

Chapter one

Introduction

1.1 Introduction

With the ever growing use of communication technology and its applications in everyday life, maintaining security of this communication is becoming an increasing demand. Cryptography, which means hide information to prevent any unauthorized access, is based on mathematical functions transforming information to random symbols and figures of incomprehensible meaning (cipher-text) (Petitcolas, Anderson & Kuhn, 1999).

The process that produces cipher-text is termed encryption, while retrieving the plain-text from the cipher-text is termed decryption. Cryptography uses keys for encryption and decryption. If the same key is used for encryption and decryption; a specific cryptography family of techniques should be used, which is symmetric cryptography (Agrawal & Mishra, 2012).

The other type is termed asymmetric cryptography, also known as public key cryptography (PKC), and uses two different keys; the first is public and the second is private, for encryption and decryption, as follows:

- Cipher-text = encrypt (plain-text, PuK)
- Plain-text = decrypt (cipher-text, PrK)

Where PuK is the public key, PrK is the private key (Bellare, Boldyreva, Desai & Pointcheval, 2001).

Signcryption is an extension of these cryptography techniques, which is built based on these techniques. The original signcryption created by used digital signature and public key encryption together not independently as it is in the signature- then- encryption. This combination resulting in reduced computational cost compared to the traditional approach signature- then- encryption. (Zheng, 1997a).

Since the invention of the signcryption and try to find a new generation of signcryption is a difficult challenge. The main problem in building a new signcryption is how to reduce computational cost and how to overcome security loopholes in the previous signcryption. Rabin cryptosystem can help in this filed with its feature in efficient encryption because it's have only one a square. (Nayak, 2015)

1.2 Cryptography

Maintaining the information transmitted between people and/or machine through online networks, which is used on daily basis in various applications, such as ATM transactions, lead to the emergence of cryptography algorithm (ciphers). Cryptography resists attacks to cover information from gain unauthorized access. Cryptography transforms the original information (plain-text) to unreadable information (cipher-text) before sending it to the receiver, in a process called encryption. The process of information retrieving of the plain-text is called decryption. Cryptography uses the same key in symmetric algorithm such as the block cipher algorithm, Data Encryption Standard (DES) and the stream cipher RC4. The other types of cryptography uses two different keys, one public and another is private, in what is called asymmetric cryptography (public key cryptography). The keys are used differently; one key for encryption and the other for decryption (Sreenivasarao, 2013). The process of encryption and decryption are shown in Figure 1.1 and Figure 1.2, respectively.

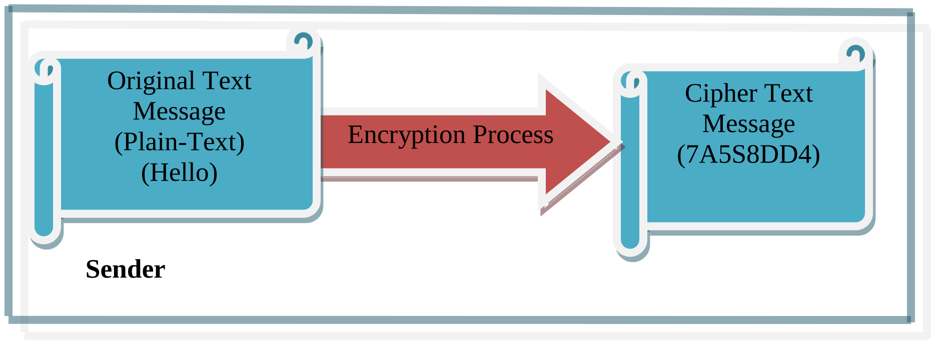

Original Text Message (Plain-Text) (Hello)
Encryption Process
Cipher Text Message (7A5S8DD4)
Sender

Figure 1.1: The Encryption Process (Ayele, Sreenivasarao, 2013)

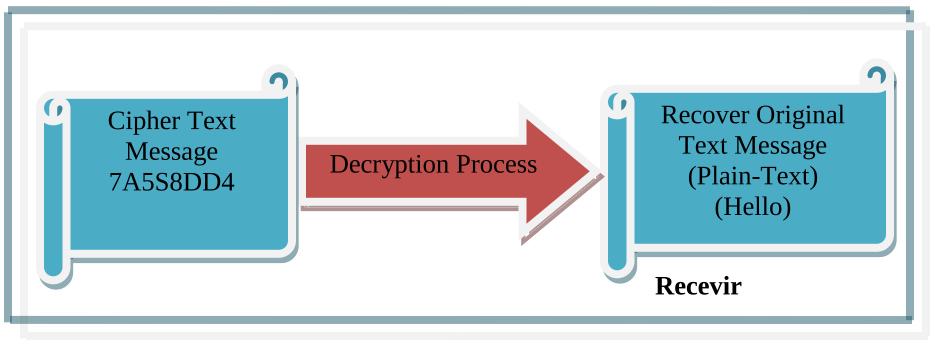

Cipher Text Message 7A5S8DD4
Decryption Process
Recover Original Text Message (Plain-Text) (Hello)
Recevir

Figure 1.2: The Decryption Process (Ayele, Sreenivasarao, 2013)

1.3 Digital Signature (DS)

It is a method of authentication, which is a proof that the message originates from the claimed sender not from another party. In digital signature the message is encrypted using the private key of the sender and decrypted by corresponding public key (Noorouzi1 et.al, 2011). Digital signature is achieved by the following steps:

1. The sender encrypts message by his private key, known as Digital Signature.
2. Digital Signature is sent to the receiver.
3. The receiver uses the sender's public key to recover the original message in decryption process.

Digital Signature is used in applications needing to verify the identity of the sender such as E-banking and E-Commerce Services. There are many digital signatures that have been invented like RSA-DS and Rabin-DS.

1.3.1 RSA Digital Signature

RSA Digital Signature was invented by Rivest, Shamir and Adleman (1977). RSA is the most popular public key algorithm and is implemented by generating two large prime numbers **p** and **q**, which kept secret. RSA is simple and easy to apply. The equations that are used to implement encryption, decryption, signing process and verification in RSA-DS are listed in Table 1.1 (Nicolas.Courtois, 2006).

Table 1.1: RSA Cryptosystem

Process	**Equations**
Encryption	$c=m^e \bmod n$
Decryption	$m=c^d \bmod n$
Signature	$s=m^d \bmod n$
Verification	$m \mathrel{?=} s^e \bmod n$
* C is cipher-text and M is plain-text	

There are some weaknesses in RSA such as the existence of similar secret key that can decrypts the cipher-text (Singh, 2013).

RSA is exposed to many types of attacks such as forward search attack, timing attack, small encryption exponent **"e"** and small decryption exponent **"d"** attacks (Bakhtiari & Maarof, 2012).

1.3.2 Rabin Digital Signature

It is a security technique formed based on the factorization problem and is similar to RSA-DS but differs by using the square root in the decryption process. Compared to RSA-DS, Rabin has faster encryption process. Rabin Digital Signature is more secure compared to RSA-DS (Srivastava & Mathur, 2013). Subsequently, Rabin is attractive for various applications. Encryption, decryption, signature and verification processes using Rabin cryptosystem are listed in Table 1.2.

Table 1.2: Rabin Cryptosystem (Galbraith, 2012).

Process	**Equations**
Encrypt(m,N)	Compute $c = m^2 \bmod N$
Decrypt(c, (p, q))	Compute $\sqrt{c} \bmod P$, $\sqrt{c} \bmod q$
Sign(m, (p, q))	Compute $s^2 = \pm m^{(p+1)/4} \pmod{p}$, $\pm m^{(q+1)/4} \pmod{q}$
Verify(m, s,N)	Check whether $m \equiv \pm s^2 \pmod{N}$.
* C is cipher-text and M is plain-text	

1.3.3 Hash Function H(.)

Also called message digest, Hash Function H(.) is a mathematical function that is used to convert variable length input string to a fixed length binary sequence (Preneel, 1999). Example of the one way hash function output using SHA-1 algorithm is given in Table 1.3.

Table 1.3: Hash Algorithm SHA-1 (Gupta, Kumar, 2014)

Test Strings	SHA-1
123456	7C4A8D09CA3762AF61E59520943DC26494F8941B
Abcdefgh	21B7C280D13AA4B59E583029F70136DEE7441F6A
Abcdefgh123456	0323661BEF7ABEB4045A981F4F3BB862D485CF8A

This table show how SHA-1 algorithm works. In a comparison between the second and third input strings it can be noted that even though there is a great similarity between these strings, a significant difference in the output is found. Because of the difficulty of reversing its process it is known as one way hash function. Hashing is considered as important in the cryptography because any change in the input even if in a single bit causes significant changes in the output. Moreover, it is impossible to find two messages that produce the same hash value (Gupta & Kumar, 2014).

In public key cryptography, instead of encrypt the whole message for authentication purpose, which can be very slow the hash value of the message is calculated using known algorithms, such as MD4, MD5, SHA or SHA256. Then, the hash value is encrypted using the sender’s private key, which is considered as the digital signature. On the receiver side, the received message is retrieved and the hash value is calculated for the received message using the same algorithm that is used by the sender. Moreover, the digital signature is decrypted by the sender's public key. After that, the receiver compares the two hash values;

if they are identical, then the sender is authenticated (Bakhtiari, Naini & Pieprzyk, 1995). Public key cryptography and digital signing using hash function are integrated for the security and authentication purposes, as illustrated in Figure 1.3.

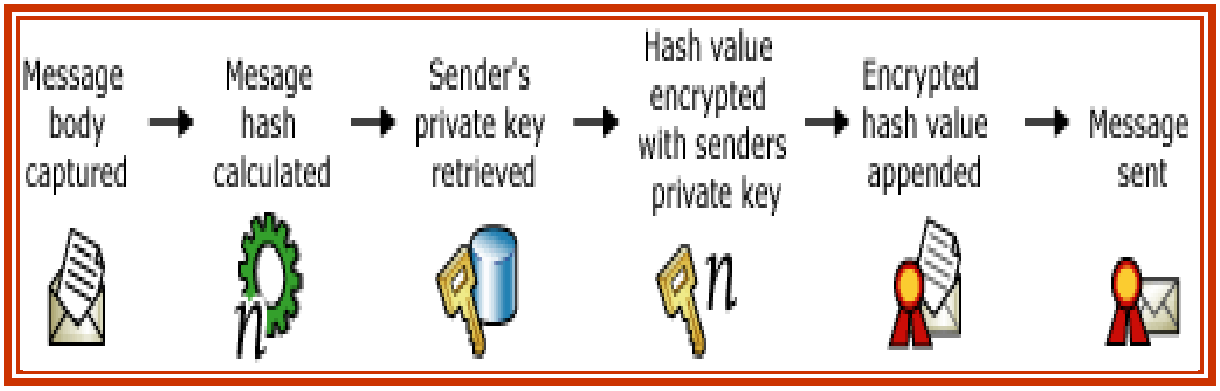

Message body captured
Mesage hash calculated
Sender's private key retrieved
Hash value encrypted with senders private key
Encrypted hash value appended
Message sent

Figure 1.3: Public Key Cryptography and Digital Signing

Using the hash function H(.) makes the encryption process faster and more secure because it is difficult to reverse this process and it's impossible to find two message produce the same hash value. Figure 1.4 shows the sequence of verifying and decryption process of the public key cryptography.

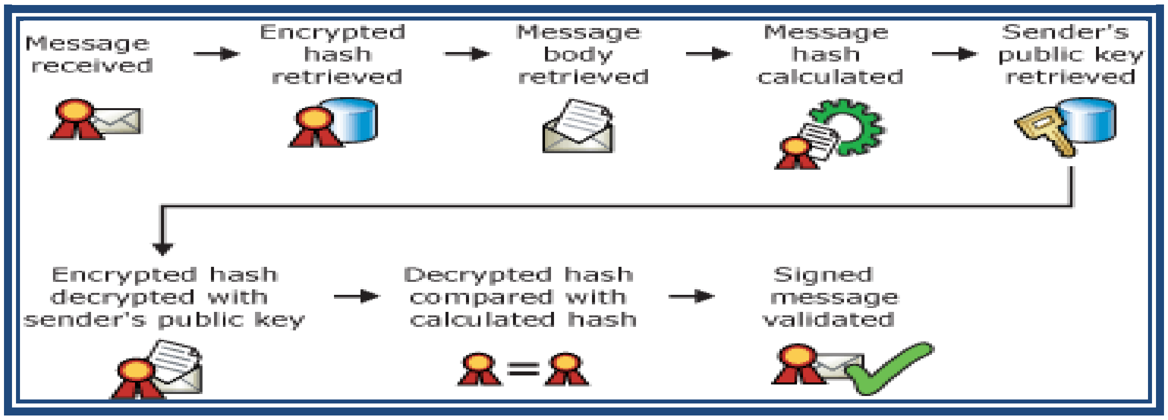

Message received
Encrypted hash retrieved
Message body retrieved
Message hash calculated
Sender's public key retrieved
Encrypted hash decrypted with sender's public key
Decrypted hash compared with calculated hash
Signed message validated

Figure 1.4: Public Key Cryptography and Verifying A digital Signature

1.4 Signcryption

Signcryption consists of two operations, public key encryption and digital signature. These operations are integrated into a single logical process in order to make the communication between the sender and receiver secure with less computation cost and communication overhead compared to the old approach of signature-then-encryption. Moreover, signcryption also involves symmetric key encryption and hashing; with and without keys (Zheng, 1997a).

1.4.1 Signature-Then-Encryption

It is the traditional approach for security and authentication in communication. This approach signs the message by digital signature scheme first and then encrypts it. At the receiver side, there are also two steps, these are: message decryption followed by verification process. Signature-then-encryption consumes high communication overhead and high computational time, which is equal to the computational cost of both operations.

Moreover, decryption process and verification both need similar time (Kumar, 2014). Signature-then-Encryption is shown in Figure 1.5

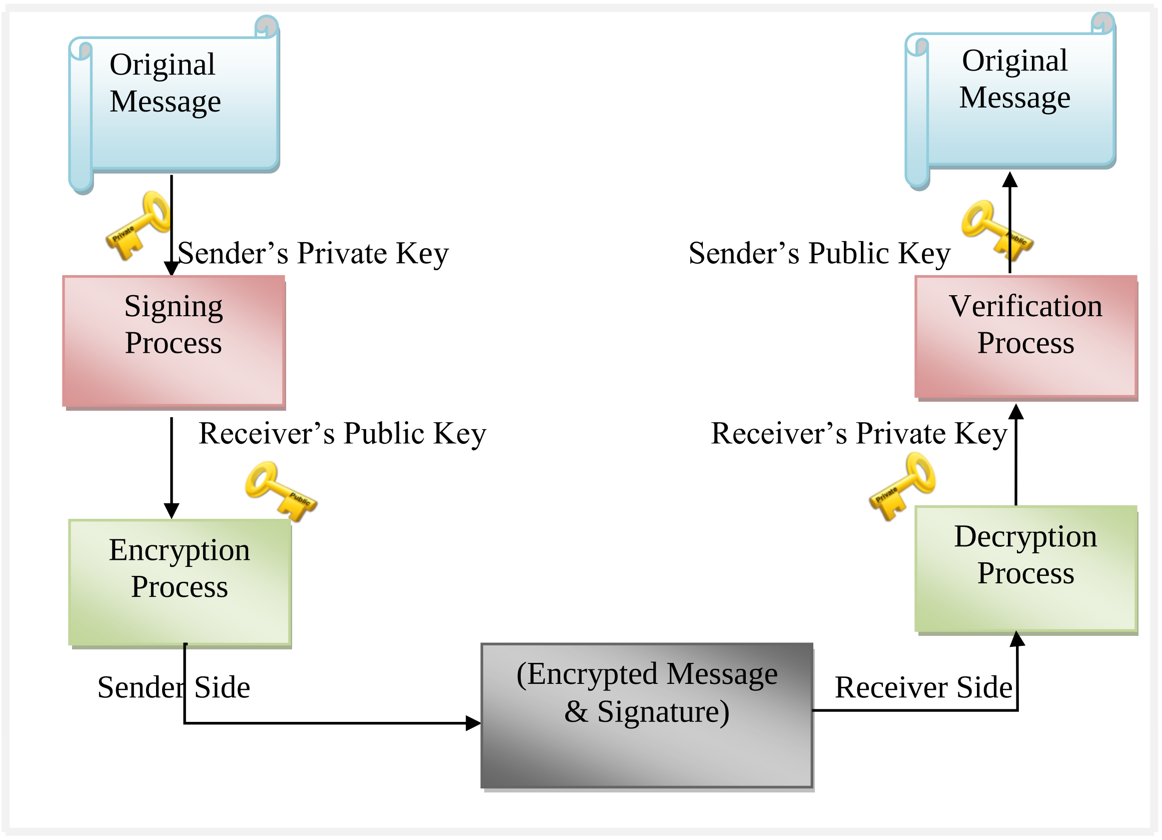


Figure 1.5: Signature-Then-Encryption (Kumar, 2014).

There are several schemes of signature-then-encryption like signature-then-encryption based on RSA digital signature and signature-then-encryption based on Digital Signature Standard (DSS) and ElGamal encryption. Signature-then-encryption methods and comparison between them are summarized in Table 1.4.

Table 1.4: Signature-then-Encryption Approaches (Zheng, 1997a).

Schemes		EXP	MUL	DIV	ADD	HASH	ENG	DEC
Signature-then-Encryption Based on RSA	**Sender**	2	-	-	-	1	1	-
	Receiver	2	-	-	-	1	-	1
Signature-then-Encryption Based on DSS and ElGamal Encryption	**Sender**	3	1	1	1	1	1	-
	Receiver	3	1	2	-	1	-	1
Signature- then-Encryption Based on Schnorr Signature and ElGamal Encryption	**Sender**	3	1	-	1	1	1	-
	Receiver	3	1	-	-	1	-	1

Where, EXP is the number of exponentiation operations, MUL is the number of multiplication operations, DIV is the number of division operations, ADD is the number of additions or subtractions, HASH is the number of times the one way or keyed hash function is implemented, ENG is the number of encryption operations using private key cipher and DEC is the number of decryption operations using private key cipher.

Through the table 1.4 notes that signature-then-encryption based on RSA is the efficient approach because it has the computational cost is the least especially in the number of exponentiation operations compared to other approaches on the table.

1.4.2 Signcryption Technique

Signcryption consists of five stages, and these are: Setup, Key generation for the sender, Key generation for the receiver, Signcryption at the sender side and Unsigncryption at the receiver side (Elshobaky, Elkabbany, Rasslan & Gurguis, 2014). The signcryption process sequence is shown in Figure 1.6.

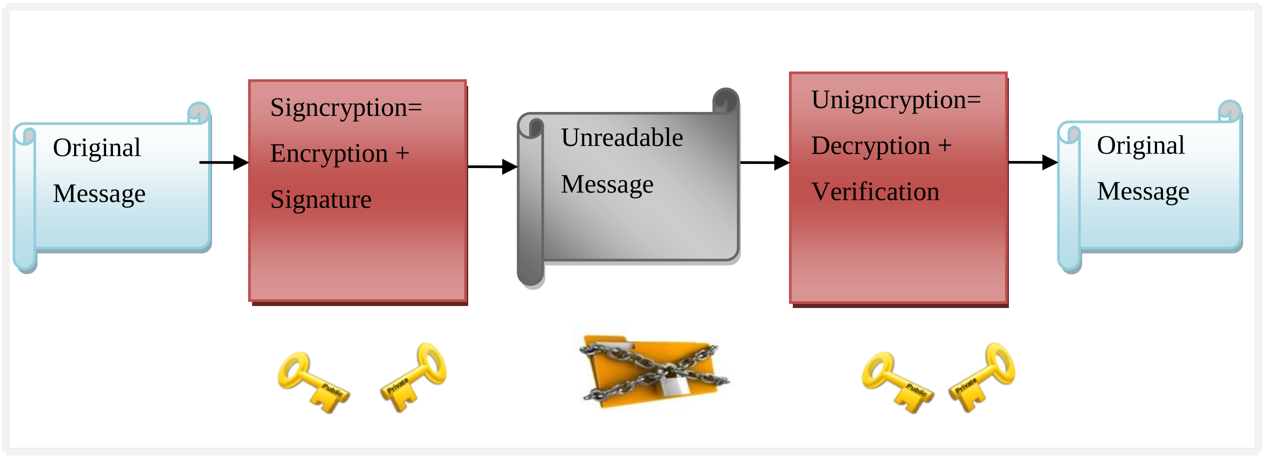

Original Message
Signcryption= Encryption + Signature
Unreadable Message
Unigncryption= Decryption + Verification
Original Message
Public
Private
Public
Private

Figure 1.6: The Signcryption Process (Elshobaky, Elkabbany, Rasslan& Gurguis, 2014).

Signcryption is more secure and efficient compared to signature-then-encryption scheme. Signcryption and Unsigncryption processes (S, U) in most existing signcryption schemes satisfied the security concerns, which are: unique unsigncryptability, security and efficiency. A comparison between the original signcryption and signature-then-encryption is given in Table 1.5.

Table 1.5: Compression between Original Signcryption and Signature-then-Encryption (Zheng, 1997b).

Various Dimensions	**Signcryption**	**Signature then Encryption**
Computation and Communication Cost	≈ Cost(signature)	Cost(signature) + Cost(encryption)
Forward Secrecy	X	√
Past Recovery	√	X
Repudiation Settlement	Interactive	No- Interactive

Signcryption based on RSA digital signature known as TBOS (Two Braids One Stone) which is one of the most well-known signcryption schemes, satisfies important security concerns which are: unforgeability and non-repudiation. RSA based signcryption has some limitations such as it is weak against any internal attack. Subsequently, new signcryption approach is required in order to resist internal attacks (Malone-Lee, 2004).

1.4.3 Signcryption and Security Concerns

Signcryption combines the security concerns of digital signature and public key encryption, these concerns are essentially confidentiality, unforgeability, integrity, and non-repudiation. In addition to some additional concerns such as public verifiability and forward secrecy which are achieved by most signcryption schemes (Toorani, & Beheshti, 2010):

1. Confidentiality: The receiver's private key must remain computationally invisible. So that no one can reach it. The process of unsigncryption is only done by the receiver. The attacker cannot get any information about signcrypted text. Therefore the unsigncryption process is invisible.
2. Unforgeability: The attacker cannot make a fake message because the sender's private key is computationally invisible. The real sender of the message he is the only person who can signcrypt the plain-text. Therefore the signcryption process is invisible.
3. Integrity: Making sure that the received message not exposed to the attack. The details of the message did not change, but as written by the sender. So, the message is original one. The sender signature helps in fulfillment the integrity.
4. Non-repudiation: Using trusted third party to help the communication parties to prove the identity of the sender. So, it becomes impossible that the sender deny the message. The trusted third parity used sender's public key. So that, the communication becomes in non-repudiation.
5. Forward secrecy: Previous messages exchanged between the communication parties must be protected if the sender's private key compromised. This is achieved by change sender's public and private keys continuously. So, if one message is compromised the previous messages will not be affected. (Toorani, & Beheshti, 2010)

1.5 Problem Statement

Different signcryption approaches were built based on various existing digital signature algorithms. The motivating goal in every new signcryption scheme is to reduce the computational cost and achieve the highest degree of security. Hackers are always trying to find out loop holes in signcryption algorithms. Thus, year after year, new signcryption approaches are built to stay strong against these enemies. The main question in this thesis is **How to implement new signcryption scheme based on Rabin digital signature to satisfy main security goals?**

1.6 Research Questions

This problem, addressed in this thesis, can be further clarified in the following questions:

1. How to develop a new signcryption scheme that is efficient in cost, secure and which can compete original Signcryption?
2. How to prove the security concerns (Correctness, Efficiency, and Security) of the Proposed Signcryption?
3. How to measure and prove the performance of the Proposed Signcryption scheme?

1.7 Goal and Objectives

The goal of this thesis is to propose a new signcryption approach based on Rabin cryptosystem. Rabin cryptosystem has been proved to be a factorization based problem. Thus, it would give more secure signcryption scheme compared to original signcryption. The Proposed Signcryption achieves the main security requirements to overcome the weakness in the signcryption based on ElGamal shortened digital signature depending on fast encryption in signcryption side and solving the problem in decryption process by recovering original plain-text through four plain-texts in the receiver side. The objectives of this research are as follows:

1. To use the features of Rabin cryptosystem to develop a new signcryption scheme.

2. To analyze the security concerns of the Proposed Signcryption scheme.

3. To implement the Proposed Signcryption approach and to check the performance of it by using real time as compared to original signcryption.

1.8 Motivation

Cryptology continuously seeks to fill versatilities in data communication against attacker, who always try to break the signcryption algorithm. So, it is important to stay ahead of those enemies. Hackers always try to penetrate the security barriers and discovery of security flaws and they succeed in their sabotage work. To stop any attempt to access hidden information, it is necessary to continuously update and modernize the signcryption mechanisms. Building new signcryption scheme based on Rabin provides security goals like confidentiality, authenticity, non-repudiation. In confidentiality the receiver's private keys computationally invisible in Rabin cryptosystem depending on the factorization problem, authenticity satisfied by digital signature in Rabin cryptosystem. Non-repudiation satisfied by using third parity. Rabin cryptosystem has been utilized for the first time in signcryption to effectively operate and to add a motive and a great challenge in the signcryption field of work.

1.9 Contribution and Significance of the Research

Building new signcryption based on Rabin cryptosystem to improve signcryption security and find the solution to Rabin cryptosystem problem in how to retrieve the original text through the four plain-texts in the unsigncryption side. The continuity of maintaining a safe distance from the enemies who are constantly trying to attack data and preventing them from reaching their goal highlights significance of the research in.

1.10 Scope of the Study

Signcryption scheme assumes importance roles in various applications, which are: saving time and to provide a secure channel for different parties in the communication. The proposed work focuses on signcryption based on Rabin equations to try actively preventing certain attacks. To complement the signcryption approach, it is important to study the old signcryption models; particularly original signcryption and make use of the features wrapped by it. Hashing and symmetric key encryption are important parts of the signcryption process, however, the details of these processes are outside the scope of this study. The presence of a third party is important to make sure the identity of the sender to achieve Non-repudiation, the security of this party is outside of the scope of this study.

1.11 Thesis Outlines

The order of the thesis is as follows:

Chapter One: Addressed cryptography in general, and provided an overview of the signcryption invention, its importance and role in the preservation of information during the communication between two parties, and presented the research problem and objective.

Chapter Two: Reviewed previous studies on signcryption which relied on different algorithms, and reviewed Rabin cryptosystem discussed in a literature review.

Chapter Three: Showed the Proposed Signcryption, which is used in the development of signcryption. The work of the new signcryption is presented in detail.

Chapter Four: Provided implementation of the Proposed Signcryption, measuring its effectiveness and compared it with original signcryption. The Proposed Signcryption strength in terms of security is shown in this chapter.

Chapter Five: General summary of the thesis and supported it by suggestions about future work are presented in this chapter to develop the Proposed Signcryption in the signcryption filed.

Chapter Two

Theoretical Background and Related Work

2.1 Overview

This chapter presents the factors, motivations and approaches related to the signcryption and the evolution of signcryption algorithms since 1997 until now. Section 2.2 presents an introduction about the invention of signcryption and how the public key cryptography helps in this filed. Section 2.3 explains how messages can be exchanged by using cryptography algorithm and how the verification of the sender is performed. Also, the importance of public key encryption and digital signature, and how they help in protecting data from any unauthorized disclosure are also explained. Section 2.4 shows different signcryption schemes, how they operate, security goals achieved and a comparison between these schemes in terms of advantages and disadvantages. Section 2.5 explains Rabin cryptosystem, how it works and its advantages. Section 2.6 provides a summary for all previous signcryption algorithms.

2.2 Introduction

Cryptography aims at supporting security communication depending on encryption and digital signature. The asymmetric cryptography, in which no keys are shared, uses public key and private key for each party, regardless of being a sender or receiver. Digital signature provides the authenticity of the message. Many digital signatures were invented such as RSA digital signature scheme, ElGamal digital signature schemes, Schnorr digital signature scheme and Elliptic Curve digital signature scheme. Signcryption, which is a public key cryptography, is invented by combining the principles of public key encryption and digital signature in order to gain the advantages of both. There are a number of signcryption schemes, such as, Zheng's Signcryption, which allows two parties to exchange information securely, more efficiently compared to signature-then-encryption. Deng’s Signcryption, Zheng-Imai Elliptic Curve Signcryption Scheme and Schnorr Signcryption are examples of the existing signcryption schemes (Chennuri & Manchala, 2015).

2.3 Background

The appearance of the signcryption came as a result of the previous efforts on the public key principles, public key algorithm, digital signature and digital signature algorithm.

2.3.1 Public Key Algorithms

Public Key refers to using a key that is not a secret, but known to all. This key is used for encryption in case of security. While this key is used for decryption in the authentication process. Public key has a corresponding private key, and is used in the opposite process of the public key. If public key is used for encryption then the corresponding private key is utilized for decryption and vice versa in the authentication (Pointcheval, 2001).

Public Key algorithm- also called asymmetric algorithm- has been discovered in 1976 by Whitfield Diffe and Martin Hellman (Molale, 2005). This discovery led to significant changes and evolution in the field of security. Public key is used after that in new discoveries such as signcryption. In public key algorithm, there are two main processes, and these are:

1. Encryption process: The sender uses mathematical function to convert text message to scrambled text.

2. Decryption process: The receiver recovers the plain-text from the scrambled text.

Asymmetric algorithm uses two different keys that are mathematically related, public and private; one of them is used for encryption and the other for decryption. In a sense, anyone can use public key to encrypt a message but there is only one person who can decrypted it; and that is the owner of the corresponding private key. In a traditional way, to send a message encrypted using public key, the following steps are followed:

1. The sender and receiver agree on specific public key algorithm.
2. The receiver's public key is sent to the sender.
3. The message is encrypted by the receiver's public key.
4. The sender sends the message to the receiver.
5. The receiver decrypts the message by his private key.

The inability to obtain the private key from the public key property of public key makes it highly secure against enemies. However, public key cryptography is relatively slow. To overcome this problem, distributed symmetric key (session key) is generated by the sender, and encrypted by the receiver’s public key, and this encrypted session key is sent to the receiver. Then, the session key is used with the fast symmetric key encryption to encrypt the communicate data. This technique uses symmetric and asymmetric algorithms called hybrid algorithm to gain the advantages of rapid processing in symmetric and high security in asymmetric algorithms. A summary of public key process is shown in Figure 2.1 (Ayele & Sreenivasarao, 2013).

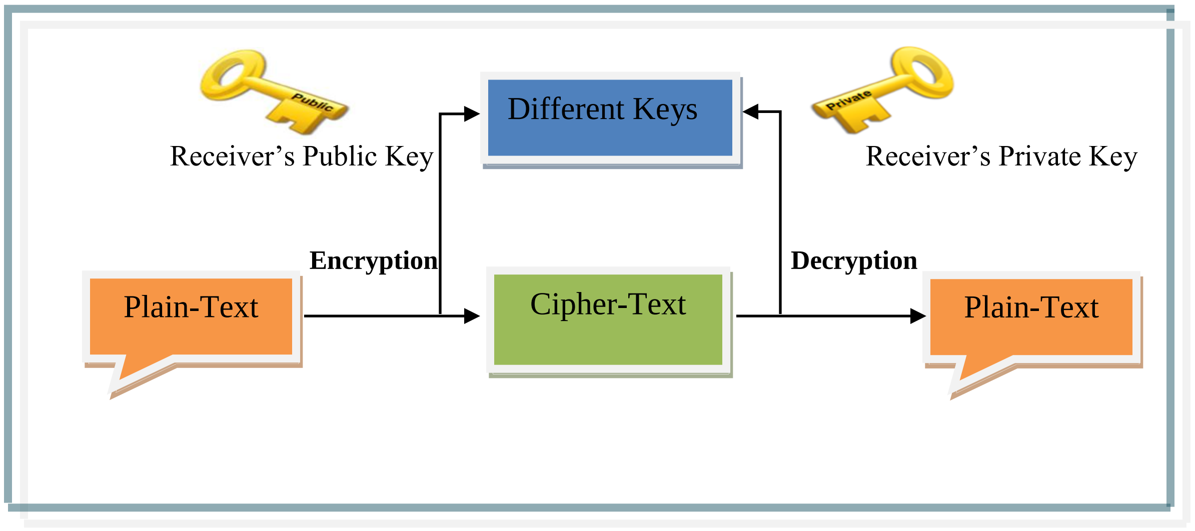


Figure 2.1: Public Key Algorithm (Ayele, Sreenivasarao, 2013)

2.3.2 Digital Signature Algorithms (DSA)

Digital Signature uses the signing process to authenticate the person who sent the message. The sender signs a message to show that he created and approved the message. Thus, signature is a way for authentication and integrity. Documents and files become electronic and they are signed electronically by the digital signature. Digital signature has been used on various applications, such as: Email, E-banking and E-Shopping (Zhu & Li, 2008). Digital signature is shown in Figure 2.2.

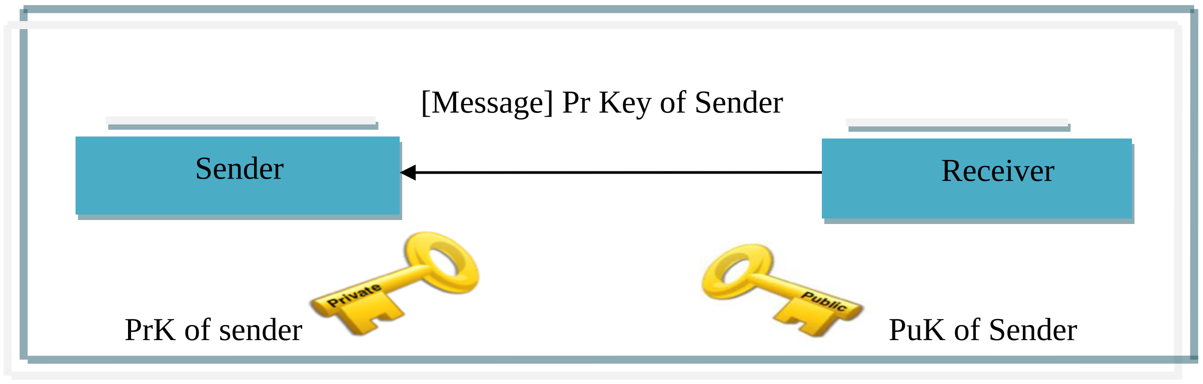


Figure 2.2: Digital Signature (Zhu & Li, 2008)

In Figure 2.2, before sending the message, the sender, signs the message to prove that the message is written by him. Then, the sender encrypts the message by his own private key. On the other side, the receiver verifies the identity of the sender by using the sender's public key to decrypt the message before accepting it.

Digital Signature Algorithm (DSA) looks like a hand sign, as it is difficult to be replicated. DSA consists of three stages:

1. Generating keys process.
2. Signing process.
3. Signature verification process.

There are various digital signature algorithms such as El-Gamal digital signature algorithm, RSA digital signature algorithm and Elliptic Curve digital signature algorithm (ECDSA). Each of them varies in the effectiveness and the resistance of the attacks like forgery attack (Roy & Karforma, 2012).

The following steps are followed in implementing digital signature:

1. In the beginning, a one-way hash of the original message is calculated.
2. The sender encrypts the hash value with his private key.

3. The sender sends the message and the encrypted hash value (digital signature) to the receiver.
4. The receiver decrypts the hash value using sender's public key, and then generates a one-way hash for the received message using the same algorithm used by the sender.
5. The receiver makes a comparison between the two hash values; if equal, then the message is authenticated, and if not, then the message is rejected.

Operation of the Digital Signature Algorithm is illustrated in Figure 2.3.

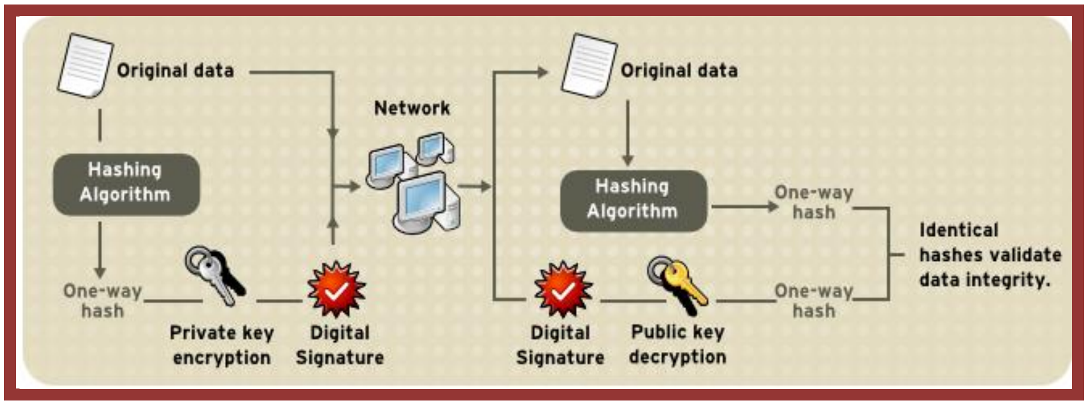


Figure 2.3: Summery of Digital Signature (A New Digital Signature Algorithm, 2011).

2.4 Related Work

There are several signcryption proposed in the literature. All of the previous signcryption schemes implement the same principle, but using different algorithms in order to achieve the main goals such as reducing computation cost and communication overhead, increasing security power by integrating encryption and digital signatures in one step.

The significant of reviewing the existing signcryption scheme is to investigate the theory of building these signcryption schemes, and to highlight the disadvantages which have to be addressed in the proposed work.

After the invention of signcryption by Zheng (1997), several modification and enhancement over this original signcryption, which depends on ElGamal digital signature algorithm, were proposed. Subsequently, new signcryption algorithms such as signcryption based on Schnorr digital signature algorithm, (Bao and Deng, 1998) signcryption scheme, Elliptic Curve-based Signcryption Scheme (ECSES1), signcryption scheme based on diffie-hellman (Libert & Quisquater, 2001), RSA-based signcryption scheme and ID-Based Proxy Signcryption were introduced.

Signcryption approaches were proposed in the literature by adapting the existing digital signature and public encryption algorithms. This section reviews the existing signcryption.

2.4.1 El-Gamal-Based Signcryption

El-Gamal cryptosystem is a public key cryptosystem proposed by Taher ElGamal in 1984, which is based on discrete Logarithm Problem. ElGamal cryptosystem gives deferent cipher-text in every encryption process. ElGamal cryptosystem is used to build first signcryption scheme (Hwang, Chang, & Hwang, 2002)

Zheng (1997) the father of signcryption, proposed the first signcryption approach based on the discrete logarithm problem in two forms (SDSS1, SDSS2). The proposed scheme used ElGamal shortened digital signature scheme combined with public key encryption. These signcryption schemes were simple, but they achieved security objectives of public key cryptography which are unforgeability, non-repudiation and confidentiality. More importantly, SDSS1 and SDSS2 succeeded in achieving the goal of signcryption, which is minimizing the computational time and communicational overhead compared with traditional approach signature-then-encryption. This goal is represented by Equation 2.1.

$$\text{Cost (Signcryption)} \ll \text{Cost (Signature)} + \text{Cost (Encryption)} \qquad \textbf{(2.1)}$$

Subsequently, signcryption is proven to give more efficient and better results in many cases. SDSS opened new horizons towards more efficient security solutions. The parameters, signcryption process and unsigncryption process in SDSS1 and SDSS2 are clarified in Table 2.1.

Table 2.1: Summary of SDSS1 and SDSS2 Parameters and Equations (Zheng, 97).

Setup (SDSS1 and SDSS2 Parameters)	
p: a large prime number.	
q: a large prime factor of p -1.	
g: integer with order q modulo p chosen randomly from [1, .., q-1]	
hash: a one-way hash function whose output has, say, at least 128 bits	
KH: a keyed one-way hash function	
(E: D): Encryption and Decryption algorithms of a private key cipher	
KeyGen Sender:	
x_a: Sender's private key, chosen uniformly at random from [1, .., q-1]	
y_a: Sender's public key ($y_a = g^{xa} \bmod p$)	
KeyGen Receiver:	
x_b: Receiver's private key, chosen uniformly at random from [1, .., q-1]	
y_b: Receiver's public key ($y_b = g^{xb} \bmod p$)	
Signcryption by the Sender	**Unsigncryption by the Receiver**
• $x \in R$ [1, .., q-1]	• $(k1; k2) = hash(ya \times g^r)^{s \times xb} \bmod p)$ if SDSS1 is used, or
• $(k_1; k_2) = hash(y_b^x \bmod p)$	• $(k1; k2) = hash\ (g \times ya^r)^{s \times xb} \bmod p)$ if SDSS2 is used.
• $c = E \times k_1 (m)$	• $m = D \times k_1 (c)$
• $r = KH \times k_2 (m)$	• Accept m only if $KH \times k_2 (m) = r$
• $s = x/(r + x_a) \bmod q$, if SDSS1 is used, or	
• $s = x/ (1 + x_a \times r) \bmod q$, if SDSS2 is used.	
• Send c, r, s to the receiver	

In SDSS1 and SDSS2, the value of **x** is chosen randomly. Then, **x** is used with the receiver's' public key **(y_b)**, to compute a hash value **K**, with 128-bits. **K** is divided to **(K_1 & K_2)** with equal 64-bits value each. Then, the cipher-text is produced by using the public key encryption **(E)** with **(k_1)** on message **(M)** and **(r)** is computed by applying one way keyed hash function **KH** with **(K_2)** on **(M)**. Finally, **(S)** is computed which differs in the two forms SDSS1 and SDSS1, to help in recovering the original text at the receiver side. Signcryption process is shown in Figure 2.4.

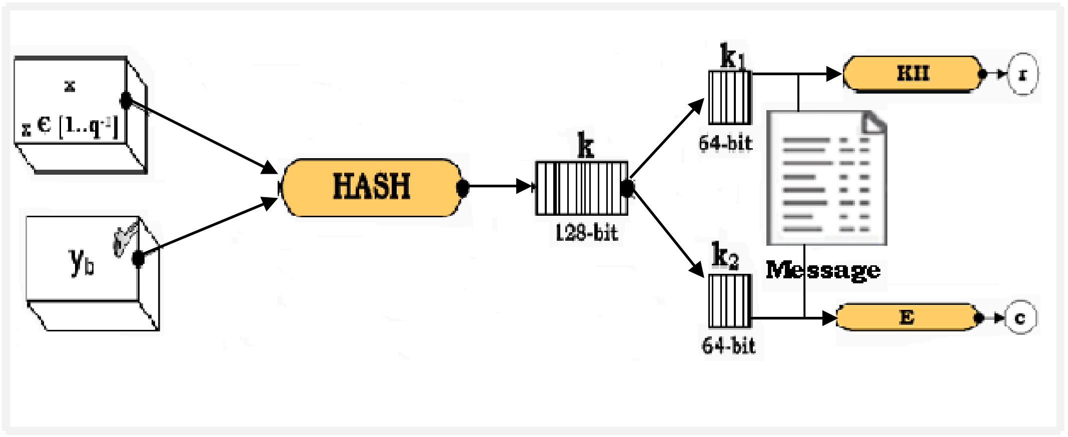


Figure 2.4: Signcryption Process

In unsigncryption process, the values of $\mathbf{y_a}$, **g**, **r**, **s** and **xb** are used to recover **(k)**, and then split it into two 64-bits keys, $\mathbf{k_1}$ and $\mathbf{k_2}$. after that, $\mathbf{(k_1)}$ is used to recover **(m)** by decrypting the cipher-text and $\mathbf{(k_2)}$ is used with one way hash function on **m** to compute **r**. Finally, the value of **r** is compared to **r** in the received envelop and the message is accepted only if the two values are equal. Unsigncryption process is shown in Figure 2.5.

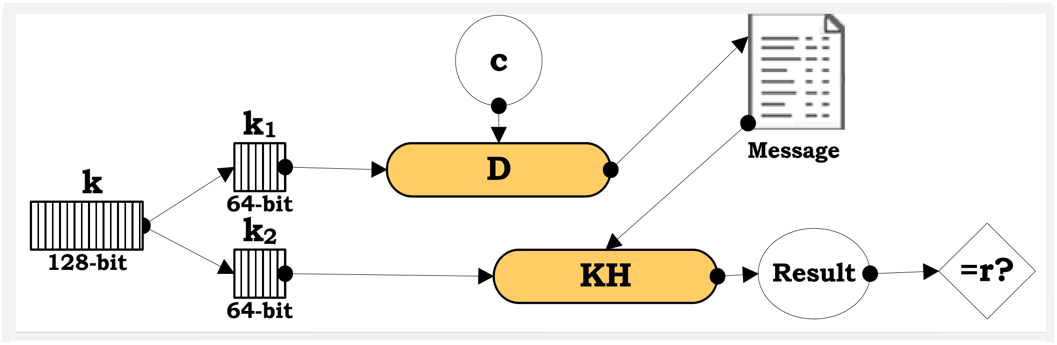


Figure 2.5: Unsigncryption Process

2.4.2 Bao and Deng Signcryption Scheme

Bao & Deng (1998) Attempted to improve the original signcryption (SDSS) based on ElGamal digital signature with computational cost lower than the original signcryption, as it contains more powers, which needs more time to execute. However, this modified signcryption failed in reducing time and cost. The goal, which is achieved by this signcryption scheme, as similar to the signature-then-encryption, is to eliminate the need of the receiver’s private key in the verification process. The modified signcryption satisfied security goals unforgeability, as it is mathematically difficult for the attacker to create sincrypted text and this makes the signcryption process only done by the sender. Non-repudiation is done by a trusted third party. Unforgeability and Non-repudiation in this signcryption based on its hard to forge **m** or **r** or **s** without knowing the secret key $\mathbf{x_a}$. Confidentiality, mathematically it is hard to attack and theft data cause in SDSS1 it is enough to resist brute force attack through the equation **r = KH (k, m)** were **k= hash ((ya ×** $\mathbf{g^r)^s}$ **mod p)**. Bao and Deng signcryption scheme operations are shown in Table 2.2.

Table 2.2: Bao and Deng Signcryption Scheme Operations

Setup (Bao and Deng Signcryption Parameters)	
p: A large prime number chosen randomly. q: A large prime factor of p -1 g: An integer with order q modulo p chosen randomly from [1, .., q-1] Hash: A one-way hash function whose output has, say, at least 128 bits KH: A keyed one-way hash function (E: D): The encryption and decryption algorithms of a private key cipher	
KeyGen Sender:	
x_a: Sender's private key, chosen uniformly at random from [1, .., q-1] y_a: Sender's public key ($y_a = g^{xa}$ mod p)	
KeyGen Receiver:	
x_b: Receiver's private key, chosen uniformly at random from [1, .., q-1] y_b: Receiver's public key ($y_b = g^{xb}$ mod p)	
Signcryption at Sender	**Unsigncryption at receiver**
• Sender randomly chooses $x \in R\ Z^*q$ then sets • K_1= hash (yb^x mod p) • K_2= hash (g^x mod p) • $c = E \times k_1(m)$ • $r = KH \times k_2(m)$ • $s = x/(r + x_a)$ mod q • Sender sends (r, s , c) to Receiver	• t_1= $(ya \times g^r)^s$ mod p • t_2= $t1^{xb}$ mod p • k_1= hash (t_2) • k_2= hash(t_1) • $m = D \times k_1(c)$, Decrypt the cipher-text to return message. • $KH \times k_2$ (m)? = r for signature verification. • Later when necessary, Receiver may forward (m, r, s) to others, who can be convinced that it came originally from Sender by verifying • k= hash(($y_a \times g^r)^s$ mod p) and $r = KH \times k(m)$

The difference here compared with the Zheng equations is the number of the utilized exponential operation. Subsequently, the computational cost is higher compared with Zheng signcryption.

2.4.3 Elliptic Curve-based Signcryption Scheme

Niel Koblitz and Victor Miller proposed a new cryptosystem based on curves. In 1985, Elliptic Curve cryptography is public key cryptography depending on curves, used in encryption and sign message with smaller key size. Elliptic Curve cryptography is characterized as fast cryptography. It is used in some application such smart cards and phones (Koblitz, Koblitz, & Menezes, 2011).

Zheng & Imai (1998) proposed new signcryption schemes, ECSCS1 and ECSCS2, which was the first signcryption schemes based on elliptic curved with smaller key size compared to RSA and DSA. Reducing the size causes reductions in the power consumption and saves time and bandwidth. ECSCS1 and ECSCS2 achieved the security goals and reduced the computational and communication cost compared to signature-then-encryption. Signcryption based on elliptic curve digital signature compared with elliptic curve signature-then-encryption saved 58% in computational cost and 40% in communication overhead. So, ECSCS1 and ECSCS2 achieved effectiveness. However, security goals archived by this signcryption is the unsigncryptability that the process of retrieving the original message text is performed, and the receiver is the only one able to retrieve the message. So, this process is unique.

Singh (2014) proposed and analyzed various signcryption schemes based on Elliptic Curves Discrete Logarithmic Problem (ECDLP) and clarified the benefits and drawbacks of each scheme. ECDLP can be used appropriately in resource constrained applications, which offered the best solutions in security and it is one of the best signcryption schemes achieving less memory requirements and required less computational costs compared to signature-then-encryption. Comparisons of some signcryption schemes based on ECDLP based on their computational cost are shown in Table 2.3.

Table 2.3: Comparison of ECDLP Signcryption Schemes on Basis of Computational Cost (Singh & Patro, 2017).

Schemes	ENC	DEC	DIV	P.M	P.A	MUL	ADD	HASH	Summation of CC
Zheng & Imai (1998)	1	1	1	3	1	3	1	4	15
Han et al (2004)	1	1	2	5	1	4	1	4	19
Hwang et al (2005)	1	1	0	5	1	1	1	2	12
Toorani (2008)	1	1	0	6	1	1	2	4	16
Mohamed (2009)	1	2	1	5	2	0	1	6	17
Amounas (2013)	1	1	0	4	2	1	0	2	11

Where, ENG is the number of encryption operations, DEC is the number of decryption operations, DIV is the number of division operations, P.M is number of Point multiplication operations, P.A is the number of point addition operations, MUL is the number of scalar multiplication operations, ADD is the number of additions or subtractions, HASH is the number of times the one way or keyed hash function is implemented and Summation of CC is computational cost of all operations.

As shown in Table 2.3 by comparison between the computational costs of Elliptic Curve based signcryption schemes. Amounas signcryption is the most efficient than the other signcryption in the table; this is because the number of calculations is the least in this Amounas signcryption.

Most signcryption schemes based on elliptic curves provide all security goals, which are: Confidentiality as the attacker needs to know receiver private key $\mathbf{v_b}$ to perform the unsigncryption process. Non-repudiation by trusted third party who can ensure that message sent from the sender. Unforgeability because it is not possible mathematically to create a fake text (**c**, **r**, and **s**) and send it to the sender without knowing sender's private key $\mathbf{v_a}$. Finally, in forward secrecy the attacker cannot access the past message of the attacker who owns $\mathbf{v_a}$ needs to know the value **r** which requires solving ECDLP on **r**, and this process is mathematically difficult (Das & Samal, 2013). An exception for this scheme is the scheme that was proposed by Han (2006), and did not support any security goals. The operation of the Zheng-Imai signcryption is listed in Table 2.4.

Table 2.4: Zheng-Imai Signcryption, (Zheng, Imai, 1998)

Setup (Zheng-Imai Signcryption Parameters)	
C: Consider C as an elliptic curve over a finite field GF (pm), either with p ¸2160 and m=1 or p=2 and m¸160. q: a large prime whose size is approximately of orderpm-1. G: a point with order q. Chosen randomly from the points on C. Hash (.): a one way hash function whose output has at least 160 bits. KH (.): a keyed one-way hash function. (E, D): the encryption and decryption algorithms of a private key cipher.	
KeyGen Sender:	
v_a: Sender's private key chosen uniformly at random from [1... q-1]. p_a: Sender's public key. ($p_a = v_a$*G).	
KeyGen Receiver:	
v_b: Receiver's private key chosen uniformly at random from [1... q-1]. p_b: Receiver's public key. ($p_b= v_b$*G).	
Zheng and Imai Signcryption	**Zheng and Imai Unsigncryption**
• v ∈R [1,...,q-1].A random number chosen by Sender. • (k_1; K_2)=hash (v × p_b). • c=E × k_1 (m). • r=KH (k_2, m). • s= v/(r+v_a) mod q. if SECDSS1 is used. • s= v/(r+v_a × r) mod q. if SECDSS2 is used. • Send c, r and s to the Receiver**.**	• u=s × v_b mod q. • (k_1; K_2)=hash (u × p_a + u × r × G).if SECDSS1 is used, • Or (k_1; K_2)=hash (u × G + u × r × p_a).if SECDSS2 is used. • m=D × k_1(c). • Accept m only if KH (k_2, m) =r.

The sender generates the sincrypted message (**c**, **r**, and **s**) by encrypting the original text to produce the cipher-text. Then, **r** and **s** are computed. In the other side. In the unsigncryption process, the receiver receives **c**, **r** and **s** from the sender and uses them to recover the message using the decryption algorithm with **k_1** Finally, verifying the digital signature which means accepting m only if **KH (k_2, m)= r**.

2.4.4 RSA-based Signcryption

RSA cryptosystem was invented by Rivest, Shamir, and Adleman in 1978. RSA cryptosystem is secure, and its security is derived from the factorization problem difficulty, and this is the basis of the algorithm. It's have storing encryption and high security (Jamgekar, & Joshi, 2013)

Lee & Mao (2002) proposed a new signcryption scheme based on RSA digital signature algorithm and proved to satisfy the security attributes, privacy and unforgeability. This model did not use symmetric encryption and supported non-repudiation in a simple way. Lee and Mao (2002) used RSA with a padding scheme such as Probabilistic Signature Scheme (PSS), which helped in building strong encryption because PSS design for secure digital signature with RSA. The use of PSS padding is the optimal use for RSA signcryption. RSA signcryption has the effectiveness to cause the message encrypted and signed by RSA its size is halved. For this reason, this signcryption is called Two Birds One Stone (TBOS).

Kirtane & Rangan (2008) proposed a new signcryption scheme based on RSA-TBOS signcryption of Mao and Lee (2002). This scheme was called RSA-TBOS-PRE, and it was the first proxy-based signcryption with re-encryption step without using bilinear maps. RSA-TBOS-PRE proxy re-encryption user converts the cipher-text to another by using calculations without knowing any details about the clear text, which facilitates proxy utilization with high security. The proxy, who is third party, becomes a semi-trusted because; through transformation process he tries to identify the original message text or keys, by delegating the work of the sender or receiver, some properties are achieved such as unidirectional, non-interactive, non-transitive and single-use proxy re-encryption scheme. Many applications used proxy re-encryption such as Digital Rights Management (DRM) of apple's iTunes, secure email forwarding and distributed secure storage systems.

2.4.5 Diffie-Hellman Based Signcryption Scheme

Diffie-Hellman cryptography or key exchange is asymmetric public key cryptography used to make a common key between the sender and the receiver, to exchange information securely (Boyko, MacKenzie & Patel, 2000)

Libert & Quisquater (2004) proposed a new signcryption scheme based on diffie-hellman cryptosystem and Boneh, Lynn and Shacham (BLS) signature. This signcryption is more effective than ElGamal-based signcryption, as it needs lesser time to execute the signcryption and has strong security and unforgeability, supports key invisibility, which is considered an important security notion that the cipher-text produced after encryption cannot be distinguished of a uniform distribution on the encrypted text space when you run this algorithm with random public Key. Besides, Diffie-Hellman signcryption. Libert and Quisquater (2004) is considered as the first signcryption based on discrete logarithm that proves unforgeability related to the diffie-hellman problem

2.4.6 Proxy Signcryption

Proxy cryptosystem is public key cryptosystem works to allow a party other than the receiver called a proxy. The third party does not have the ability to know the details of the message and the sender’s private key. The sender authorizes the proxy by using his new proxy key to encrypt the message before sending it to the receiver (Mambo & Okamoto, 1997).

Li & Chen (2004) proposed a new signcryption scheme based on proxy signature. The main idea of this scheme is to involve other party to sign the message instead of the original person, who is known as proxy signer. A proxy signature is suitable to be used in many applications such as E-cash system, E-commerce and mobile communications. Proxy-signcryption provides security nations such as indistinguishability against adaptive chosen cipher-text attacks (IND-IDSC-CCA) and unforgeability against adaptive chosen messages attacks (EF-IDSCACMA). Unforgeability is derived from his identity based on the signature scheme under the Devi-Helman Computation Diffie-Hellman (CDH) assumption. The proxy signcrypter receives the public parameters in public channel and the proxy key in secret channel. This process is implemented one time only; if it is completed successfully. Then, the proxy signcrypter can signcrypt any message and sends it to the receiver.

Jun-Bao & Guo-Zhen (2005) proposed a signcryption based on a multi-proxy multi-signature scheme. In this signcryption, the original signcrypter group gives authority to the proxy signcrypter to sign the message. Multi-Proxy Multi-signcryption can be implemented only by the cooperation of all original signers to give the authority to the proxy signer. This signcryption satisfied security attributes unforgeability depending on that a multi-proxy multi-signature has a private key and without it, a valid multi proxy multi-signcryption cannot be created, Strong identifiability that the receiver can distinguish between the normal signcryption and multi-proxy multi-signature and this key differs from its own key.

Swapna et.al. (2012) proposed a new Proxy-signcryption scheme. In addition to Proxy signcryption, several algorithms have emerged as an extension to the proxy signcryption such as ring signcryption, multi signcryption**.** Proxy signcryption scheme differs from normal models and it is suitable for applications, in which the delegate rights writes commonly dramatically such as e-cash systems, global distribution networks, grid computing, mobile agent applications and mobile communications. The main idea in this primitive form of the original entity is to delegate another entity to work on the signing the message.

2.4.7 ID-Based Signcryption

ID-Based cryptosystem was created by Shamir (1984) and allows the sender and receiver to exchange messages without exchange their keys depending on the identity of the user. In ID-Based cryptosystem, a third party is used to generate the private keys by private key generator. By a trusted third party, the identity of the sender is ascertained. This cryptosystem achieved success in reducing complexity (Youngblood, 2005).

Hua, Li & Aimin (2011) proposed an identity based ring signcryption scheme without random oracle by using bilinear pairings. Ring signcryption provides confidentiality provided by encryption and non-repudiation provided by digital signature. Cipher-text length is constant and independent of the size of the ring. The mathematical relationship between cipher-text length and the ring size is linear. Thus, using constant cipher-text size is more practical in ring signcryption design, so it is more efficient than other ring signcryption, it consume less time.

2.4.8 Hybrid Signcryption Scheme

Hybrid cryptosystem combines between symmetric and asymmetric public cryptography, Hybrid cryptosystem can be built by any two cryptosystems, the first is a key of encapsulation scheme (asymmetric cryptosystem) and the second is a data encapsulation scheme (a symmetric cryptosystem) (Gupta & Singh, 2013)

Dent (2005), proposed a new signcryption scheme based on hybrid digital signature, which is secure against any forgery attacks done by any party, except the sender, and this is known as insider security. The main idea of the new scheme is to use asymmetric key encapsulation mechanism (KEM) and a symmetric data encapsulation mechanism (DEM) together. This union supports security concerns and gains more benefits compared to using one of these mechanisms individually. In particular, hybrid scheme helps the asymmetric cryptography to solve the time-consuming problem required to process long messages and to overcome the symmetric cryptography problem of insider attacks. In KEM_DEM encryption scheme, KEM is used to produce **K** (a symmetric key) and to encrypt the key $\mathbf{C_1}$. Then, DEM is used with the generated key, **K** to encrypt the message symmetrically to produce cipher-text $\mathbf{C_2}$. KEM is also used on the other side to decrypted $\mathbf{(C_1{:}C_2)}$ to recover the symmetric key **K**. DEM is used with the symmetric key **K** to retrieve the message m from the cipher-text $\mathbf{C_2}$. KEM_DEM signcryption provides security goals, authentication, integrity, confidentiality service and non-repudiation.

Dent (2005) Proposed Signcryption scheme based on hybrid digital signature, which combines asymmetric cryptography and a symmetric encryption as a part of asymmetric encryption. This signcryption provides security against any attack by a third party, which is known outsider security. A signcryption KEM and DEM work independently to satisfy security features, which is confidentiality, integrity and provides efficient processes comparable with the traditional approach. In this signcryption, the decapsulation and verification algorithms are separated and work independently.

2.4.9 Schnoor Based Syncryption

Savu (2012) presented a signcryption approach based on schnoor digital signature. This signcryption scheme works better than the original scheme that depended on El-gamal digital signature, because it consumes lesser computational resources. The proposed scheme was applied on two-users and multi-user. In both types, the data protection is not only achieved from an outsider party, but also from an insider party. The insider party can be the sender or the receiver themselves or another party who has the secret key for one of them. If the attacker is the sender himself, the Schnorr signcryption prevents him from creating a fake message and this helps achieve non-repudiation. But, if the sender is the attacker himself, the Schnorr signcryption prevents him from unsigncryptd the message. So, Schnorr signcryption provides confidentiality even if the sender's key is known.

The attacker can be more dangerous in multi-user model, in which a bigger opportunity for successful attack is presented. Schnorr signcryption processes are given in Table 2.5.

Table 2.5: Schnorr Signcryption

Setup (Schnorr Signcryption parameters)	
p: a large prime number, public to all q: a large prime factor of p-1, public to all g: an integer with order q modulo p, in [1,… , p-1], public to all hash: a one-way hash function KH: a keyed one-way hash function = KH × k(m) = hash(k, m) (E, D): the algorithms which are used for encryption and decryption of a private key Cipher.	
Sender's keys:	
X_a: Sender's private key, chosen randomly from [1, .., q-1] Y_a: Sender's public key g^{-xa} mod p.	
Receiver's keys:	
X_b: Bob's private key, chosen randomly from [1, .., q-1] Y_b: Bob's public key = g^{-xb} mod p.	
Signcryption of message by the Sender	**Unsigncryption of (c; r; s) by the receiver**
• Split k in k_1 and k_2 of appropriate length. • Calculate r = KH × k_2(m) = hash(k_2, m) • Calculate s = x + (r × x_a) mod q • Calculate c = E × k_1(m) = the encryption of the message m with the key k_1. • Sends to the receiver the values (r, s and c).	• k = hash (($g^s \times ya^r)^{-xb}$ mod p) • Calculate k using r, s, g, p, Ya and x_b • Split k in k_1 and k_2 of appropriate length. • Calculate m using the decryption algorithm m = D × k_1(c). • Accept m as a valid message only KH × k_2(m) = r.

Schnorr has less computational cost compared with Zheng's signcryption. This is clear from the equation used to calculate the value of **s** which is **s = x + (r × x_a) mod q**, which is lesser in computational cost, to the calculation of the S value in Zheng's signcryption, as the following, **s= x/(r+x_a) mod q**. In unsigncryption side, calculating the value of **K** is different and is implemented in Schnorr scheme as follows, **k = hash ($g^s \times ya^r)^{-xb}$ mod p** while in Zheng's scheme, it is calculated as follows, **k = hash ($ya \times gr)^{s \times xb}$ mod p**.

2.4.10 Comparison

After reviewing previous studies related to the existing signcryption scheme, a comparison between these signcryption algorithms is conducted and summarized as given in Table 2.6.

Table 2.6: Comparison between Different Signcryption Schemes.

Author (Year)	**Digital Signature**	**Advantages**	**Disadvantages**
Zheng (1997)	El-gamal shortened digital signature scheme (SDSS1 , SDSS2)	Confidentiality, Integrity, Non-repudiation and Unforgeability	Did not achieved forward secrecy of message confidentiality
Bao & Deng (1998)	El-gamal digital signature	Confidentiality, Integrity ,Unforgeability and Public verification	Not effective as Zheng signcryption, Did not achieve forward secrecy.
Zheng and Imai (1998)	Elliptic curve discrete logarithm problem (ECDLP)	Confidentiality, Integrity and Unforgeability	Did not achieve forward secrecy and public verification.
Lee & Mao (2002)	RSA digital signature algorithm	Supports non-repudiation in a simple way.	Missing security aspects, high computation power requirement.

Li & Chen (2004)	**Proxy signature**	**Suitable for applications e-cash systems, global distribution networks.**	**The computation of the pairing still remains time-consuming**
Libert &Quisquater (2004)	Diffie-hellman problem	Supports key invisibility, strong security and unforgeability	Not secure against non-adaptive chosen cipher-text attacks
Dent (2005)	Based on hybrid digital signature KEM and DEM	Authentication, integrity, confidentiality service and non-repudiation	No satisfactory model for multi-user security.
Savu (2012)	Schnoor digital signature	Provide efficient security solutions in: E-payment, authenticated.	Dose not suitable for large groups

Overall, security level provided by these signcryption schemes varied depending on the security attributes that have been considered in developing them. Table 2.7 summarizes the achieved security concerns by the existing signcryption scheme.

Table 2.7: Security Concerns for Different Signcryption Schemes

Signcryption Schemes	**Confidentiality**	**Integrity**	**Unforgeability**	**Non-Repudiation**	**Forward Secrecy**
El-Gamal Signcryption	Yes	Yes	Yes	Yes	No
Bao & Deng Signcryption	Yes	Yes	Yes	Yes	No
Elliptic Curves Signcryption	Yes	Yes	Yes	Yes	No
RSA Signcryption	Yes	Yes	Yes	Yes	Yes

ID-Based Signcryption	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
A Diffie-Hellman Based Signcryption	Yes	Yes	Yes	Yes	Yes
Hybrid Signcryption Schemes	Yes	Yes	Yes	Yes	Yes
Schnoor Syncryption	Yes	Yes	Yes	Yes	Yes

2.5 Rabin Cryptosystem

Public key cryptography was invented by Michael O. Rabin in 1979, and Rabin depends on the difficult factorization with security near to RSA cryptosystem. Various studies have investigated the Rabin cryptosystem, and below is a review of some of these studies.

Eliay & Schipani (2011) presented several Rabin blind signatures and explained the way to defend the weakness in the forgery. The presented approach was based on the use of two primes $\boldsymbol{p \equiv 3 \pmod 8}$ *and* $\boldsymbol{q \equiv 7 \pmod 8}$*. The blind signature, distinguish type of digital signature, are important for e-commerce, where the message writer and the sender are different. In this digital signature, the message writer produces disguised message for the signer, then it is signed by the signer. While this digital signature is sample digital signature, it may be vulnerable to forgery and other attacks such as RSA blinding attack.*

Srivastava & Mathur (2013) presented Rabin cryptosystem with the Chinese reminder theorem (CRT) and explained how the decryption process was done in different scheme. The purpose of this work was to ease the problem of the extra complexity, which required finding out the true plaintext from the four possible outputs in decryption process. Because Rabin used public key **n** in the encryption and two private factor **p** and **q** whereas **$n= p \times q$**, in the decryption, the cipher-text can be obtained using Equation 2.2.

$$\mathbf{c = m^2 \bmod n.} \qquad \mathbf{(2.2)}$$

Chinese reminder can be used to recover the plaintext using Equation 2.3.

$$\mathbf{mp = \sqrt{c} \bmod p, \quad mq = \sqrt{c} \bmod q} \qquad \mathbf{(2.3)}$$

One of the four possible results in decryption process is the plaintext. To find unique decryption, redundancy in the message and added extra bits of information will help in resolving this problem.

Ciss & Youssef (2013) proposed a new cryptosystem based on two hard problems, factorization and discrete logarithm problem. This cryptosystem combines these two hard problems in the process of key generation, encryption and decryption. A solution to the ambiguity of Rabin cryptosystem was found, which helped in conducting efficient cryptosystem. This result came from the new encryption function **$E(x) =x^3 \bmod n$**, where **$n= p \times q$**, and **p, q** are two large prime number. The security of new cryptosystem based on cube root extraction. The proposed approach was proven to be secured against many types of attacks, which are direct attack, factoring attack and discrete logarithm attack.

Hashim (2014) suggested an update for Rabin cryptosystem to be more secure by using three prime numbers **p, q** and **r**, which became private and **n= p × q × r** which became public. This cryptosystem was called H-Rabin cryptosystem and the cryptosystem based on factorization. In the encryption process, only the public key is needed, where in decryption process, we need the three prime numbers **p**, **q** and **r** by using square root. Compared to RSA cryptosystem, Rabin is more secure which is hard as integer factorization, and H-Rabin cryptosystem is more secure than Rabin cryptosystem because there are eight square roots. So, there are eight possible messages in encryption process. Using three prime numbers makes the H-Rabin cryptosystem much harder than figuring two keys. So, H-Rabin is more secure and efficient than Rabin. The disadvantage of H-Rabin cryptosystem is that there are eight plain-text recovered in decryption process one of them is true.

Chakraborty, Biswas & Mandal (2014) modified Rabin cryptosystem by building a hybrid Rabin cryptosystem at containing both symmetric and asymmetric cryptosystem. In the symmetric cryptosystem process, add to the plain-text a shard key between the sender and the receiver, which becomes private and the plain-text will change after this process. In the asymmetric cryptosystem process, encryption process is conducted for the new plain-text by Rabin cryptosystem. After that, the cipher-text is sent to the receiver with two values. Chinese Remainder Theorem used in the decryption process, four plain-texts in the result from square roots modulo, the two values used to return the plain-text, only that have the two private values will return the original plain-text. If the Hacker decrypted the cipher-text, he gets the modified plain-text, not the original one.

Sidorov & Yandex (2015) presented Rabin-Williams (RW) digital signature, which is based on factorization problem. RW is similar to RSA, but, with some advantages such as using small exponent which accelerates verification process. This paper explained how to deal with the digital signature implantation problem with blinding technique designed to prevent the timing of the attacks. In RW, the public key **n= p × q**, where **p** and **q** two large prime number. This paper attempted to increase security of RW digital signature.

2.6 Summary

In this chapter, various studies related to signcryption were reviewed, analyzed and summarized. This summary covered the advantages, disadvantages, goals and cost of the existing signcryption schemes. Overall, Most the existing signcryption schemes achieved the original goal, which was to reduce the computational cost as compared to the traditional approach of signature-then-encryption. However, these schemes differed between them, to a certain degree, in achieving the security concerns. Zheng (1997) started the revolution in the invention of signcryption which has achieved reduced the computational cost and communication overhead but this scheme has not been achieved forward secrecy. Bao & Deng (1998) try to improve original signcryption but he failed. Zheng and Imai (1998) achieved a good result compared to the original scheme signature-then-encryption but this scheme did not achieve forward secrecy and public verification. RSA based signcryption by Lee & Mao (2002) Supports non-repudiation in a simple way. Libert and Quisquater (2004) produced signcryption based on Diffie-Hellman that is better than the original signcryption and Supports key invisibility, Savu (2012) proposed Schnoor signcryption which is more efficient signcryption in computational cost compared with the original signcryption. Attempting to improve the previous signcryption to save time and overcome the weakness in the security aspects is an increasing requirement

Rabin cryptosystem with two private keys and one public key used to implement new signcryption. Rabin has fast encryption help in build efficient signcryption.

Chapter Three

Proposed Work

3.1 Overview

This chapter presents the proposed work, and that is developing a new signcryption scheme based on Rabin cryptosystem. The organization of this chapter is as follows: Section 3.2 is an introduction, and gives the reason for choosing Rabin cryptosystem to build new signcryption scheme. Section 3.3 presents a solution for Rabin decryption problem. Section 3.4 analyses Rabin cryptosystem and how to generate signcryption using it. Section 3.5 presents the Proposed Signcryption algorithm and its equations. Section 3.6 gives a summary for the proposed signcryption.

3.2 Introduction

This chapter proposed a new signcryption approach based on Rabin cryptosystem, which is based on the factorization problem. Rabin is similar to RSA, but is more secure and efficient (Srivastava & Mathur, 2013). The Proposed Signcryption tried to overcome the problem of Rabin cryptosystem in decryption process to recover one plain-text instead of recovering four plain-texts. This unique decryption needs lower cost than the decryption process that resulting in four plain-texts.

The need to build a new system of signcryption is to keep abreast of any new attack. The new system must be more powerful in repelling any attack. Hackers have the ability to know how the system works. So, depriving the attacker from accessing any weak points of the system is the critical point here. Messages are not penetrated to maintain the confidentiality of communication and to keep it safe. Achieving this goal requires continuous development and modernization, whether, through the development of the previous signcryption, or by building new signcryption approach.

Rabin cryptosystem is characterized by several properties that help in building a powerful new signcryption approach. This algorithm derives its strength from the hardness of factoring problem. Rabin cryptosystem adds to the signcryption more security to maintain the message details. This warranty comes through the difficulty of retrieving the original message of the message in the decryption process. Even if the decryption process is done, there is four results. No one knows which one represents the original text of the message. Rabin cryptosystem chose the original message by convert the message to binary and preset bits ”000”. This process helps in determining the original message in decryption process.

3.3 Solving Rabin cryptosystem Problem

The problem of Rabin cryptosystem is in the decryption process and it needs solving to build the Proposed Signcryption scheme based on Rabin. Initially, the plain-text is encrypted and then, the cipher-text is sent to the receiver. The process of encryption includes:

1. Generates two large prime numbers randomly **p** and **q**.

2. Computes public key **n** by using the private keys **p** and **q** in Equation 3.1.

$$\mathbf{n=p \times q} \tag{3.1}$$

3. **3.** Represents the message as an integer **m** in the range **{0, 1, … n-1}**. Convert it to binary and preset bits**"000"** and used the new value as integer. It can also repeat the binary number before retrieving the integer value.

4. **4.** Computes the cipher-text as shown in Equation 3.2.

$$\mathbf{c = m^2 \bmod n} \qquad \mathbf{(3.2)}$$

In the receiver's side, the unsigncryption process begins, and here's the problem as there are four values for the original message, Equation 3.3, Equation 3.4, Equation 3.5, Equation 3.6, Equation 3.7 and Equation 3.8 are implemented as shown in the following steps:

1. Take $\mathbf{x_1}$ and $\mathbf{x_2}$ as in following equations:
   - $\mathbf{x_1 = \sqrt{c} \bmod p}$ **(3.3)**
   - $\mathbf{y1 = \sqrt{c} \bmod q}$ **(3.4)**
2. Get 4 square roots of **c (mod n)** using Chinese Remainder Theorem (CRT). $\mathbf{x_1}$ and $\mathbf{x_2}$ in the Equations **3.3** and **3.4** consecutively are used as in the following equations:
   - $\mathbf{m_1 = x_1 \times q \times (q^{-1} \bmod p) + y_1 \times p \times (p^{-1} \bmod q) \pmod n}$ **(3.5)**
   - $\mathbf{m_2 = n - m_1}$ **(3.6)**
   - $\mathbf{m_3 = x_1 \times q \times (q^{-1} \bmod p) - y_1 \times p \times (p^{-1} \bmod q) \pmod n}$ **(3.7)**
   - $\mathbf{m_4 = n - m_3}$ **(3.8)**

Only one **m** of them has required redundancy, it's the original message.

In the decryption process there are four plain-texts, only one of them is the original. So, how this original text is defined? Presets bits help to resolve this issue by adding a set of bits to the message before encrypting it. After that, when the decryption process is done, four plain-text are returned but one has required redundancy it's the original message.

3.4 Rabin Cryptosystem Analysis for Signcryption

1. **Formulas for parameters:** parameters in the setup of the Proposed Signcryption as follows:
   - $\mathbf{k_1}$: Secret key is chosen randomly in the order {0, 1.....n_2-1}.
   - **H**: One Way Hash function (SHA-1 hash function).
   - **KH**: A keyed one-way hash function (SHA-1 keyed hash function).
   - **E**: Encryption process by (Rijndael Algorithm).
   - **D**: Decryption process by (Rijndael Algorithm).
2. **Formulas for Key generation:** The Proposed Signcryption uses three keys for the sender and other three keys for the receiver. For the sender, two private keys $\mathbf{p_1}$ and $\mathbf{q_1}$ are generated, which are large prime numbers chosen randomly and there is one public key $\mathbf{n_1}$**,** which is produced through the process of multiplying the two private keys, as represented by Equation 3.9. $\mathbf{n_1 = p_1 \times q_1}$ **(3.9)**

   On the other hand, the receiver similarly calculates his public key $\mathbf{n_2}$ as represented by Equation 3.10.

   $$\mathbf{n_2 = p_2 \times q_2} \qquad \mathbf{(3.10)}$$

3. **Formulas for Signcryption:** In the sender's side, the process of signcryption takes place in the Equation 3.11, Equation 3.12, Equation 3.13 and Equation 3.14 as shown in the following steps:

- **k** is integer random number known only by the sender, **k** is converted to binary and preset with “000” and then return to decimal number to produce $\mathbf{k_1}$. Then $\mathbf{k_1}$ is used to calculate **v** as in the Equation 3.11.

  $\mathbf{V= k_1^2 \bmod n_2}$ **(3.11)**

- The hash of $\mathbf{k_1}$ is computed using one way hashing function SHA-1 as in the Equation 3.12.

  $\mathbf{k_2= H (k_1)}$ **(3.12)**

- The cipher-text is computed by encryption process of the message with the key $\mathbf{k_2}$. Rijndael Algorithm is used in message encrypting as in the Equation 3.13.

  Calculate $\mathbf{c= E \times k_2 (m)}$ **(3.13)**

KH is the one-way keyed hash function, and is computed for the message with $\mathbf{k_2}$ by the SHA-1 keyed hash algorithm. After the signcryptd text is produced, the sender of the message sends **v, c** and **r** to the receiver. **r** is calculated as in the Equation 3.14.

$\mathbf{r= KH (m, k_2)}$ **(3.14)**

After the process of signcryption is completed the value **v**, **c** and **r** are sent to the receiver.

4. **Formulas for unsigncryption:** In the receiver's side, the process of unsigncryption takes place in the Equation 3.15, Equation 3.16 Equation 3.17, Equation3.18, Equation 3.19, Equation 3.20, Equation 3.21, Equation 3.22, and Equation 3.23 as shown in the following steps:

- The value **v** comes from the sender and $\mathbf{v = k_1^2 \bmod n_2}$ as is evident through the Equation 3.11. This represents Rabin encryption. Now Rabin is used to find $\mathbf{k_1}$:

- Take $\mathbf{x_1}$ **and** $\mathbf{x_2}$ as in following equations:
  - $\mathbf{x_1 = \sqrt{v} \bmod p_2}$ **(3.15)**
  - $\mathbf{y_1 = \sqrt{v} \bmod q_2}$ **(3.16)**

- Get 4 square roots of **v (mod** $\mathbf{n_2}$**)** using Chinese Remainder Theorem (CRT) as shown in the Equation 3.17 Equation, 3.18, Equation 3.19 and Equation 3.20:
  - $\mathbf{s_{r1} = x_1 \times q_2 \times (q_2^{-1} \bmod p_2) + y_1 \times p_2 \times (p_2^{-1} \bmod q_2) \pmod{n_2}}$ **(3.17)**
  - $\mathbf{s_{r2} = n_2 - s_{r1}}$ **(3.18)**
  - $\mathbf{s_{r3} = x_1 \times q_2 \times (q_2^{-1} \bmod p_2) - y_1 \times p_2 \times (p_2^{-1} \bmod q_2) \pmod{n_2}}$ **(3.19)**
  - $\mathbf{s_{r4} = n_2 - s_{r3}}$ **(3.20)**

Convert the four results to binary; only one square of them has required redundancy it's the original $\mathbf{k_1}$.

- The hash of $\mathbf{k_1}$ is computed using SHA-1 one way hash function as in the Equation 3.21.

  Calculate $\mathbf{k_2 = H\ (k_1)}$ **(3.21)**

- The message is computed by a decryption process for the cipher-text with the key $\mathbf{k_2}$. Rijndael Algorithm being used for decrypts the cipher-text as in the Equation 3.22.

  Calculate $\mathbf{m = D \times k_2\ (c)}$ **(3.22)**

KH is the one-way keyed hash function, and is computed for the message with $\mathbf{k_2}$ by the SHA-1 keyed hash algorithm. After $\mathbf{r'}$ is calculated, it is compared with $\mathbf{r}$ that comes from the sender. If it is identical, then; the message is accepted, if not; the message is rejected, $\mathbf{r'}$ is calculated as in the Equation 3.23.

$\mathbf{r' = KH\ (m, k_2)}$ **(3.23)**

3.5 The Proposed Signcryption based on Rabin Cryptosystem

The Proposed Signcryption based on Rabin cryptosystem after solving Rabin problem in the decryption process is illustrated with its parameters and equations details are given in Table 3.1.

Table 3.1: Parameters and Equation of the Proposed Signcryption

Setup (Signcryption parameters)	
• **Hash:** One-way hash function. • **KH:** A keyed one-way hash function. • **E:** Encryption algorithm. • **D:** Decryption algorithm.	
Key Generation	
Sender's keys: • **p_1**: Sender's private key, chosen uniformly at random. • **q_1**: Sender's private key, chosen uniformly at random. • **n_1**: Sender's public key, $\mathbf{n_1= p_1\times q_1}$.	**Receiver's keys:** • **p_2**: Receiver's private key, chosen uniformly at random. • **q_2**: Receiver's private key, chosen uniformly at random. • **n_2**: Receiver's public key, $\mathbf{n_2 =p_2\times q_2}$.
Signcryption (By The Sender)	
V: Calculate key (V) to use it in encryption and authentication	$\mathbf{V= k_1^2 \bmod n_2}$ Integer **k** in the range {0, 1… n_2-1}. **k** is converted to binary and preset bits: “000” and returned to decimal to produce $\mathbf{k_1}$.
k_2: Hashed of the key $\mathbf{k_1}$ using SHA-1.	$\mathbf{K_2 = H (k_1)}$.
c: Cipher-text after encrypted the message	$\mathbf{c=E \times k_2(m)}$
r: Using hashing keyed function SHA-1 to hash the message.	$\mathbf{r =KH(m,k_2)}$
The sender sends v, r, and c to the receiver	

Unsigncryption (By The Receiver)	
V: Retrieving the onetime key used to encrypt and authenticate a message.	**$V= k_1^2 \bmod n_2$ (Rabin)** To recover k_1, calculates 4 square roots : • Take **$x_1= \sqrt{v} \bmod p_2$** and **$y_1= \sqrt{v} \bmod q_2$**. • Get 4 square roots of **$V(\bmod n_2)$** using Chinese Remainder Theorem (CRT): **1.** **$s_{r1}= x_1\times q_2\times (q_2^{-1} \bmod p_2)+y_1\times p_2\times (p_2^{-1} \bmod q_2) (\bmod n_2)$** **2.** **$s_{r2}= n_2 – s_{r2}$** **3.** **$s_{r3}=x_1\times q_2\times (q_2^{-1} \bmod p_2)-y_1\times p_2\times (p_2^{-1} \bmod q_2) (\bmod n_2)$** **4.** **$s_{r4}= n_2 – s_{r3}$** Convert the four results to binary; only one square of them has required redundancy, it’s the original **k_1**.
k_2: Hashed of the key **k_1** using SHA-1.	**$K_2 = H (k_1)$.**
m: Decryption of the c to recover the original text.	**$m=D \times k_2 (c)$**
r´: Using hashing keyed function SHA-1 to hash the message.	**$r´= KH(m, k_2)$**
Chick if the two hashing value is equal to accept the message or not.	**$r==r´$**

From the above table, the new signcryption focuses on how to use the Rabin equations to build a strong sincrypted text by generating **k_2** and encrypting it in the signcryption side; sending the cipher-text with two values **v** and **r** that used the sender private keys, these values give more security to the Proposed Signcryption compared with original signcryption SDSS1 and SDSS2 which uses one value. From the unsigncryption side, the $\mathbf{k_2}$ is recalculated and that is the basis for retrieving the original message, the decryption process based on Rabin makes the message safe although it takes more time compared with SDSS1 and SDSS2 signcryption.

3.6 Summary

In this chapter, the Proposed Signcryption is built to take advantage of Rabin cryptosystem properties in encryption speed and its security which is different from others in the decryption processes that depend on factorization problem. Rabin decryption process has four results; and only one is true and this means additional cost.

Chinese Remainder Theorem (CRT) is used to get the solution in the unsigncryption process. The Equation 3.17, Equation 3.18, Equation 3.19 and Equation 3.20 show the four possible rustles for the original onetime $\mathbf{k_1}$. This process is critical in the Proposed Signcryption.

The Proposed Signcryption equations include encryption and decryption process in the Equations 3.13 and 3.22 consecutively; symmetric cryptography is used in this process because it's very fast. One-way keyed hash function used to create digital signature in the Equation 3.14. The verification process in the Equation 3.23 take place and the message is rejected if the two hash values are not equal.

Chapter Four

Implementation and Results

4.1 Overview

This chapter presents implementation steps of the Proposed Signcryption. C Sharp **(C#)** is used to implement the Proposed Signcryption, to obtain and compare results. Section 4.2 explains dataset that used in the implementation. Section 4.3 shows how C# is used in details to implement the Proposed Signcryption. Section 4.4 shows the computational cost for the Proposed Signcryption. Section 4.5 presents the results of the implementation and shows the performance of the Proposed Signcryption. Section 4.6 explains the security concerns that are satisfied by the Proposed Signcryption. Section 4.7 gives summary for this chapter.

4.2 Dataset

Dataset is a collection of texts files used in implementing process for the Proposed Signcryption and the original signcryption (SDSS1 and SDSS2) to see the execution time for the Rabin process consumes compared with the execution time for the original signcryption (SDSS1 and SDSS2). Dataset used in the execution process are really tens of text files selected from the website TEXTFILES.COM. There are 40 categories, the selected data of 4 categories: computer, hacking, internet and programming. Dataset is chosen with different sizes start from 4830 bytes to 96802 bytes.

4.3 Implementation Steps

The Proposed Signcryption is implemented using C# language:

4.3.1 Dataset Uploaded

In this proposed signcryption, the dataset is uploaded at the beginning of the implementation to show the Proposed Signcryption performance. Text files used from TEXTFILES.COM are shown in Table 4.1.

Table 4.1: Sample of Text Files from Dataset

file_no	File name	file size (Byte)
1	webwar.txt	4911
2	batch.txt	51183
3	tr823.txt	106655
4	zeninternet.txt	158659
5	cihac009.txt	208948
6	interest.txt	261074
7	orange.txt	312213
8	vendors1.txt	362624
9	cguide_3.txt	414402
10	netguide.txt	465662

4.3.2 Hashing SHA-1

Using hashing SHA-1 and Keyed SHA-1 in the implementation process. Hashing SHA-1 is a secure cryptography hash function used for authentication by producing digital signature identifying the sender's identity. The results from applying hash function are a string of characters with fixed length, regardless of input data. Keyed SHA-1 is a way for authentication and its one type of a message authentication code (MAC) which contains hash function and secret key.

4.3.3 Rijndael Algorithm (Advanced Encryption Standard) AES

Rijndael algorithm is symmetric cryptography that supports different key sizes of 128, 192 and 256 bits. This algorithm has been widely accepted in the world as it is a secure encryption method because of the length of the encryption key. It is used to convert message to unreadable text (Cipher-Text) in encryption process, and to recover the original message in decryption process in the proposed signcryption. Rijndael used in order to achieve confidentiality and security during the data transfer process.

4.3.4 Hardware Specification

To install and execute the program, the personal computer (PC) is used with the following features:

- Operating system: Windows 10 home 64-bit.
- Processor: Intel (R) Core (TM) i3-2350 m CPU @ 2.3 GHz.
- Memory: 4096 MB RAM.

4.4 Computational Cost for the Proposed Signcryption

One of the most important goals of the signcryption process is reducing computational cost. The Proposed Signcryption satisfied low computational cost in the sender's side by the signcryption process compared to the original signcryption. But, the unsigncryption process did not achieve that goal.

4.4.1 Low Computational Cost for the Proposed Signcryption in the Signcryption process

The Proposed Signcryption process has high effectiveness in the signcryption process performed by the sender. By comparing the computational cost of the signcryption process for the Proposed Signcryption and the computational cost for signcryption based on ElGamal shortened digital signature scheme (SDSS1 and SDSS2) the efficiency of the signcryption process is determined. Comparisons between signcryption computational cost for the Proposed Signcryption and the original signcryption are shown on Table 4.2.

Table 4.2: Computational Cost of the Signcryption Process in the Proposed Signcryption

Schemes	ENC	DEC	HASH	MOD.ADD	MOD.MUL	MOD.EXP
SDSS1	1	0	2	1	1	1
SDSS2	1	0	2	1	2	1
Proposed Signcryption	1	0	2	0	0	1

Where ENC is the number of encryption operations, DEC is the number of decryption operations, HASH is the number of times the one way or keyed hash function is implemented, MOD.ADD is the number of modular addition, MOD.MUL is the number of modular multiplication and MOD.EXP is the number of exponentiation operations.

As shown in the table 4.2 the Proposed Signcryption consumes less computational cost than the original schemes (SDSS1 and SDSS2). This means fast signcryption process in the sender side for the Proposed Signcryption.

4.4.2 High Computational Cost for Proposed Signcryption in the Unsigncryption process

The Proposed Signcryption process has low effectiveness in the unsigncryption process by receiver. By comparing the computational cost of the unsigncryption process in the Proposed Signcryption to the computational cost for unsigncryption process in the signcryption based on ElGamal shortened digital signature scheme (SDSS1 and SDSS2) the efficiency of the unsigncryption process for the Proposed Signcryption is determined. Comparisons between the unsigncryption computational costs of the Proposed Signcryption and the original signcryption are shown on Table 4.3.

Table 4.3: Computational Cost of the Unsigncryption Process in the Proposed Signcryption

Scheme	**ENC**	**DEC**	**Hash**	**Mod. Add**	**Mod. Mul**	**Mod. Exp**
SDSS1	0	1	2	0	2	2
SDSS2	0	1	2	0	2	2
Proposed Signcryption	0	1	2	2	6	4

Where ENC is the number of encryption operations, DEC is the number of decryption operations, HASH is the number of times the one way or keyed hash function is implemented, MOD.ADD is the number of modular addition, MOD.MUL is the number of modular multiplication and MOD.EXP is the number of exponentiation operations.

As shown in the table 4.3 the Proposed Signcryption consumes more computational cost than the original schemes (SDSS1 and SDSS2) in the unsigncryption process. This means slower signcryption process in the receiver side for the Proposed Signcryption, this is because the process of chose correct value from the four possible in the decryption process.

4.5 Analysis Result for the Purposed Signcryption

To determine the effectiveness of the Proposed Signcryption action, the speed process is measured for signcryption and unsigncryption process. Measurement process using selected dataset that contain text files of different sizes for the Proposed Signcryption, SDSS1 and SDSS2 algorithm is also used.

4.5.1 Signcryption Performance

In the signcryption process, the Proposed Signcryption is faster than SDSS1 and SDSS2, the Proposed Signcryption consumes lesser time. Signcryption time for the Proposed Signcryption and the original signcryption (SDSS1 & SDSS2) is shown in Figure 4.1.

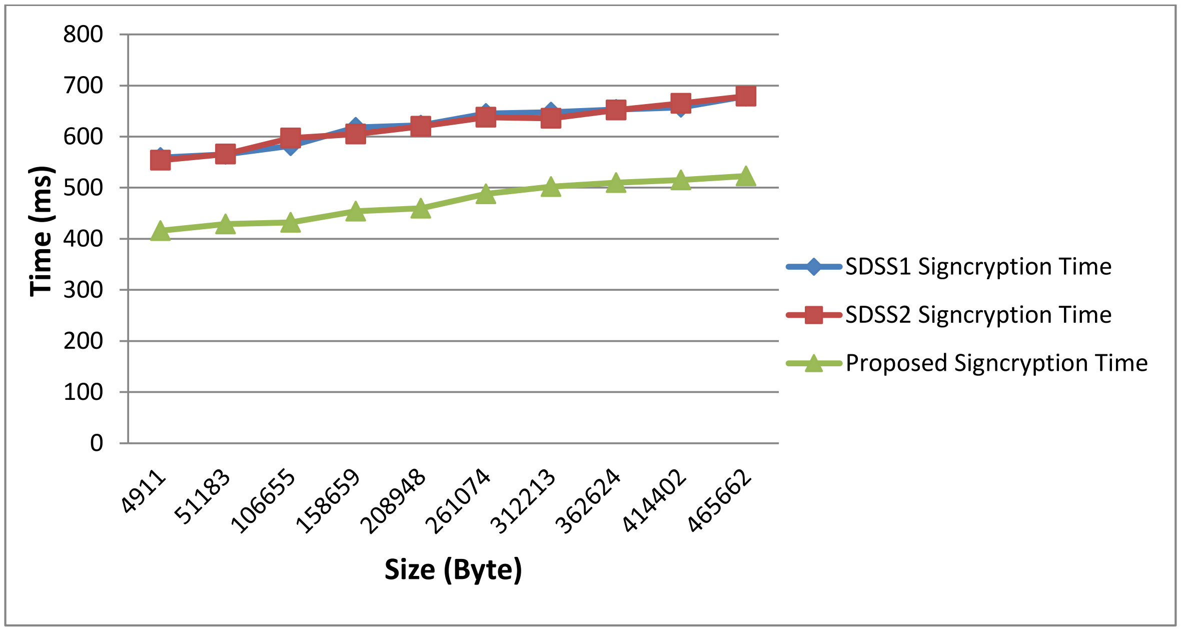


Figure 4.1: Analysis during Signcryption

As seen in Figure 4.1, the Proposed Signcryption achieved better results compared to original methods in the signcryption speed. The reasons for this result is due to the fact that it does not contain the number of exponents as they are in the SDSS1 and SDSS2 which are need more time. The best result in the Proposed Signcryption for text files with size 4911 byte which consume 416 milliseconds at a rate of 25.6% reduction compared to the original signcryption SDSS1 and at a rate of 24.9% reduction compared to the original signcryption SDSS2. The worst result in the Proposed Signcryption for text files with size 465662 byte which consume 523 milliseconds at a rate of 23% reduction compared to the original signcryption SDSS1 and SDSS2. Here it is concluded that the reduction rate decreases with increasing file size. Finally the average for signcryption execution time in the Proposed Signcryption is 472.9 milliseconds at a rate of 24.1% reduction compared to the original signcryption SDSS1 and at a rate of 23.9% reduction compared to the original signcryption SDSS2.

4.5.2 Unsigncryption Performance

In the unsigncryption process, the Proposed Signcryption is slower than SDSS1 and SDSS2. The Proposed Signcryption consumes more time. Unsigncryption time for the Proposed Signcryption and the original signcryption (SDSS1 & SDSS2) is show in Figure 4.2.

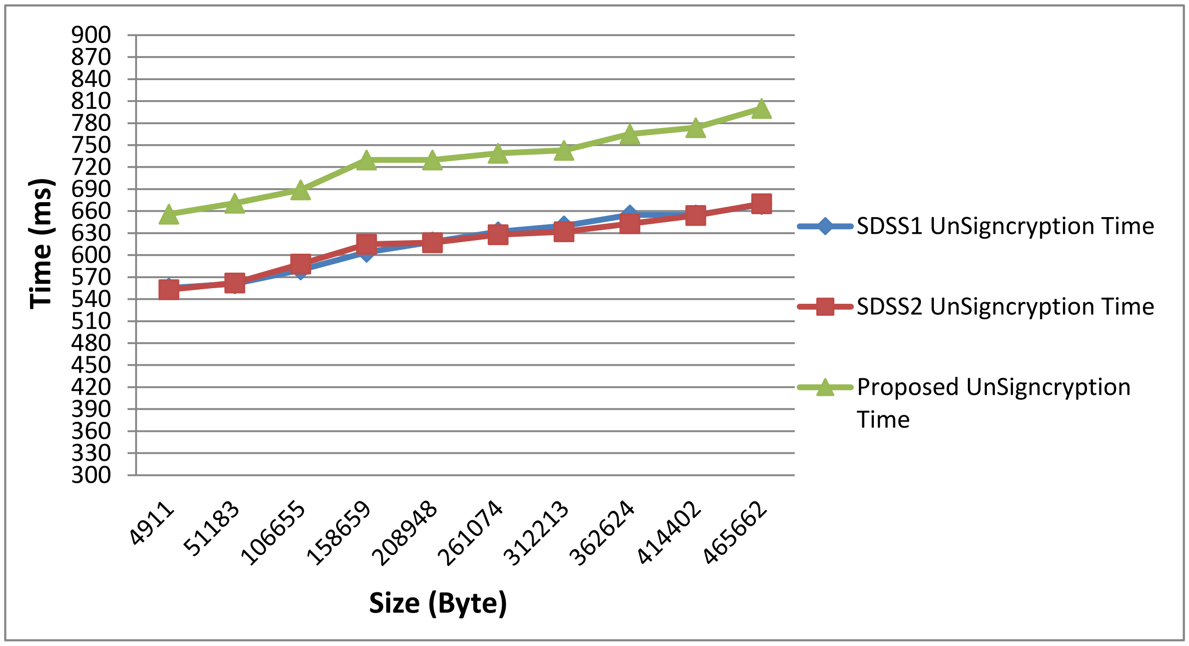


Figure 4.2: Analysis during Unsigncryption

As shown in Figure 4.2, the Proposed Signcryption did not achieve better results compared to original methods in the unsigncryption speed. The reasons for this result is explained by the fact that the decryption process consumes more time to find the correct result among the four results to recovering the original message. The best result in the Proposed Signcryption for text files with size 4911 byte which consume 656 milliseconds at an increase rate of 18.2% compared to the original signcryption SDSS1 and at an increase rate of 18.6% compared to the original signcryption SDSS2. The worst result in the Proposed Signcryption for text files with size 465662 byte which consume 800 milliseconds at an increase rate of 19.6% compared to the original signcryption SDSS1 and at an increase rate of 19.4% compared to the original signcryption SDSS2. Finally the average for unsigncryption execution time in the Proposed Signcryption is 730 milliseconds at an increase rate of 18.3% compared to the original signcryption SDSS1 and at an increase rate 18.5% compared to the original signcryption SDSS2.

4.5.3 Full Execution Time

The Proposed Signcryption takes less execution time than SDSS1 and SDSS2. Accordingly, the Proposed Signcryption is more effective than original signcryption. The full execution time for the Proposed Signcryption and the original signcryption (SDSS1 & SDSS2) is shown in Figure 4.3.

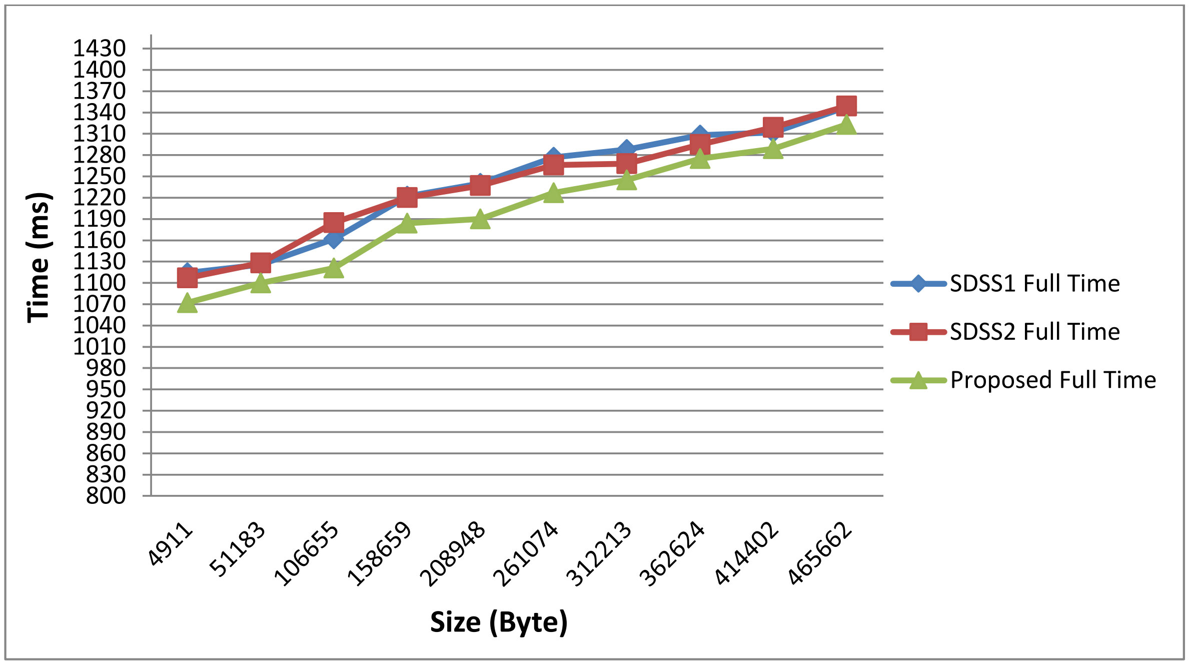


Figure 4.3: Full Execution Time for the Proposed Signcryption, SDSS1 and SDSS2

As seen in Figure 4.3, the proposed the Proposed Signcryption achieved better results compared to original signcryption in the full execution time. The best result in the Proposed Signcryption for text files with size 4911 byte which consume 1072 milliseconds at a rate of 3.8% reduction compared to the original signcryption SDSS1 and at a rate of 3.2% reduction compared to the original signcryption SDSS2. The worst result in the Proposed Signcryption for text files with size 465662 byte which consume 1323 milliseconds at a rate of 2% reduction compared to the original signcryption SDSS1 and SDSS2. Here it is concluded that the reduction rate decreases with increasing file size. Finally the average for signcryption execution time in the Proposed Signcryption is 1202.6

milliseconds at a rate of 3% reduction compared to the original signcryption SDSS1 and at a rate of 2.8% reduction compared to the original signcryption SDSS2.

4.6 Security Concerns of the Proposed Signcryption

1. **Confidentiality:** It is to ensure that the message reaches the specified receiver. Confidentiality is realized through preventing unwanted people from accessing the contents of the message. The attacker tries to access the message details in many ways; one of them is reincarnation of the personality of the receiver. If he succeeds and gets the receiver's private keys, he decodes the message. Confidentiality prevents this from happen and is done by keeping the receiver’s private key away from the hands of the attackers. So, no one can unsigncrypt the message unless it holds the receiver’s private key. That is, no one can play the role of the receiver. The unsigncryption process is done by the specified receiver who only can cover the message details. In the Proposed Signcryption confidentiality proved depending on the factorization problem. Through the equations in the receiver side, the process of retrieving $\mathbf{k_2}$ value is a factorization problem. Through Equation 3.17 , Equation 3.18, Equation 3.19 and Equation 3.20:

   - $\mathbf{s_{r1}= x_1\times q_2\times (q_2^{-1} \bmod p_2)+y_1\times p_2\times (p_2^{-1} \bmod q_2) \pmod{n_2}}$
   - $\mathbf{s_{r2}= n_2 - s_{r1}}$
   - $\mathbf{s_{r3}=x_1\times q_2\times (q_2^{-1} \bmod p_2)-y_1\times p_2\times (p_2^{-1} \bmod q_2) \pmod{n_2}}$
   - $\mathbf{s_{r4}= n_2 - s_{r3}}$

Remember that **q_2** and **p_2** are receiver's private keys. This means that no one can recover the message except the receiver. The confidentiality is illustrated in Figure 4.4.

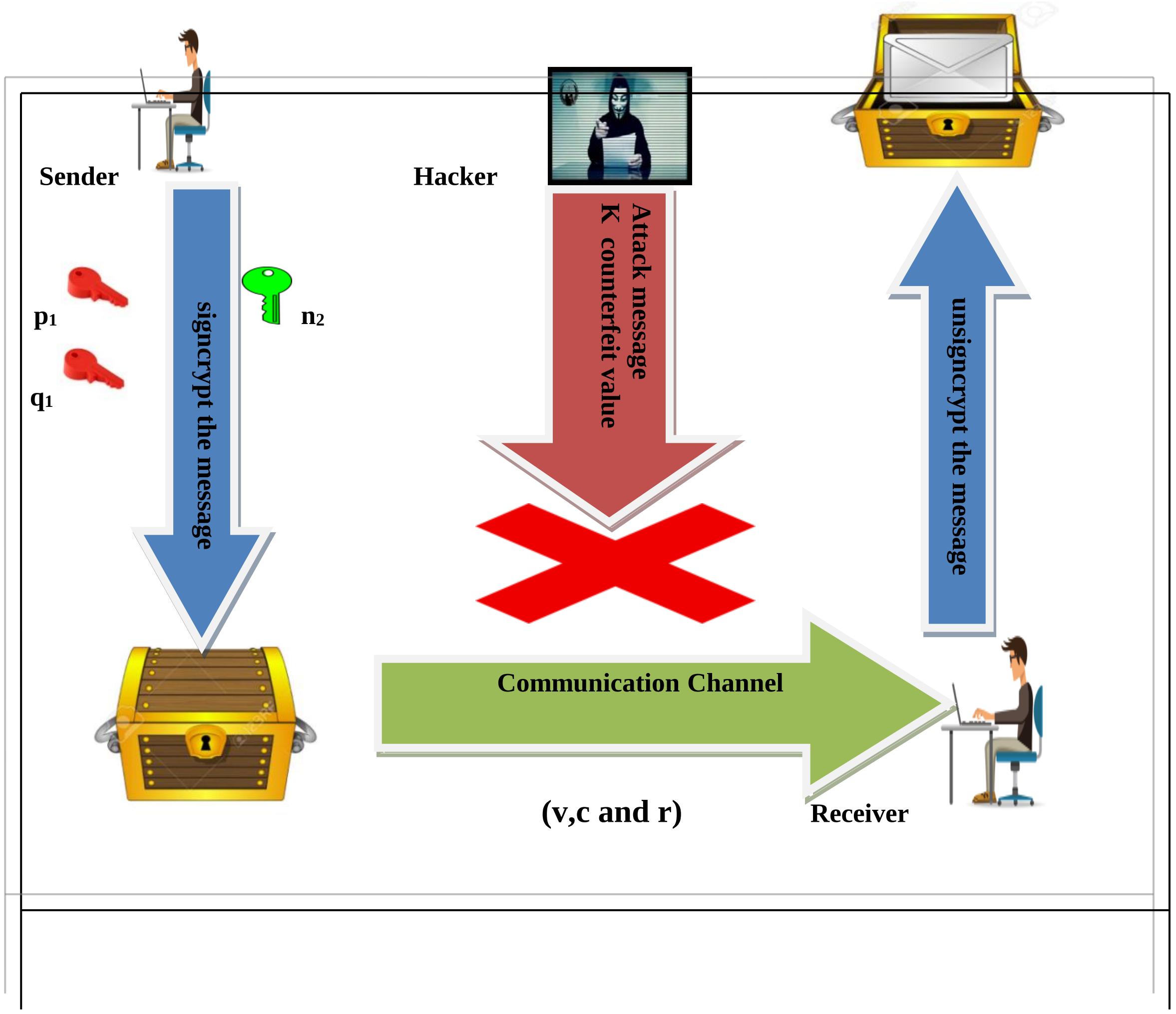


Figure 4.4: Confidentiality of the Proposed Signcryption

2. **Unforgeability:** One of the ways to penetrate communication between the sender and receiver is by trying the unauthorized person to create a forged message. Unforgeability is achieved by validate the sender of the message, and that is to verify the real sender and that the message is not written by an unreliable party.

When the attacker writes a forged message he must signcrypt it and sends it to the receiver. Here, how the unforgeability takes place? The unreliable party attempts to signcrypt the forged message, which must only be done by the real sender's private key. If that does not happen, the receiver rejects any forged message that will not be signcrypted by the sender's private key. The Proposed Signcryption cannot signcrypt the message without the sender's private key. This is done depending on factorization problem.

The receiver rejects any message that doesn't use the private keys $\mathbf{p_1}$ and $\mathbf{q_1}$, the unforgeability is illustrated in Figure 4.5.

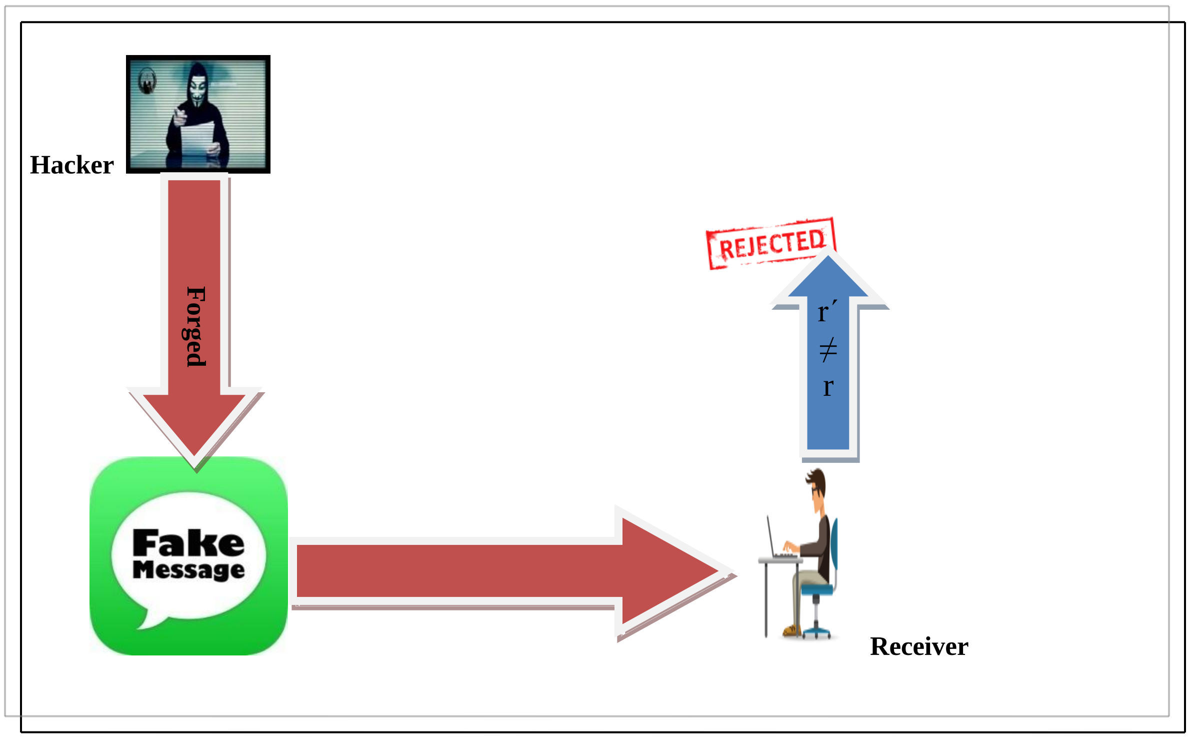


Figure 4.5: Unforgeability of the Proposed Signcryption

3. **Integrity:** The channel between the sender and the receiver not secured. To ensure that the message arrives to the receiver as it is without any change, if there is an unreliable party trying to manipulate the message and attempts changing its contents, the question is how to detect this change? When the message details are changed, the value of **r** changes, and when the message reaches the receiver, he makes the right decision to accept or reject the message by creating a value **r´** for the message that comes from the supposed sender and compare it with r coming from the sender. If the message is changed, it is impossible that **r´= r**; so, the message is rejected. The Proposed Signcryption ensures the integrity of the message as illustrated in Figure 4.6.

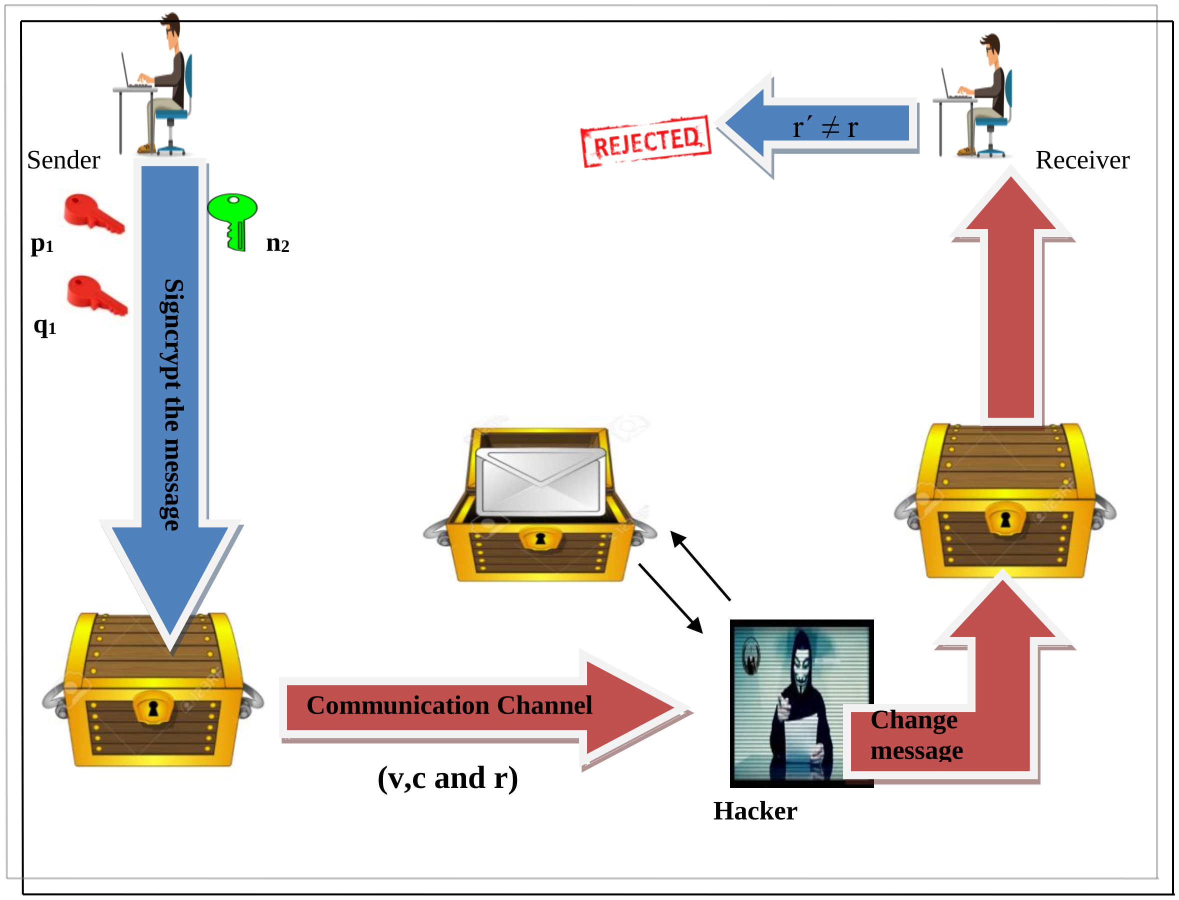


Figure 4.6: Integrity of the Proposed Signcryption

4. **Non-Repudiation:** Before delivering the message to the receiver, a trusted party other than the sender or the receiver is used to achieve a Non-repudiation state. The use of this party aims to the protection of the rights of the parties (sender and receiver) in the communication between the two. In the case where one of them denied the message, the trusted party checks the identity of the sender and ensures that the message is sent to the receiver.

The existence of a trusted third party ensures that the message reaches the desired receiver but it causes extra time to the Proposed Signcryption. The Non-repudiation is illustrated in Figure 4.7.

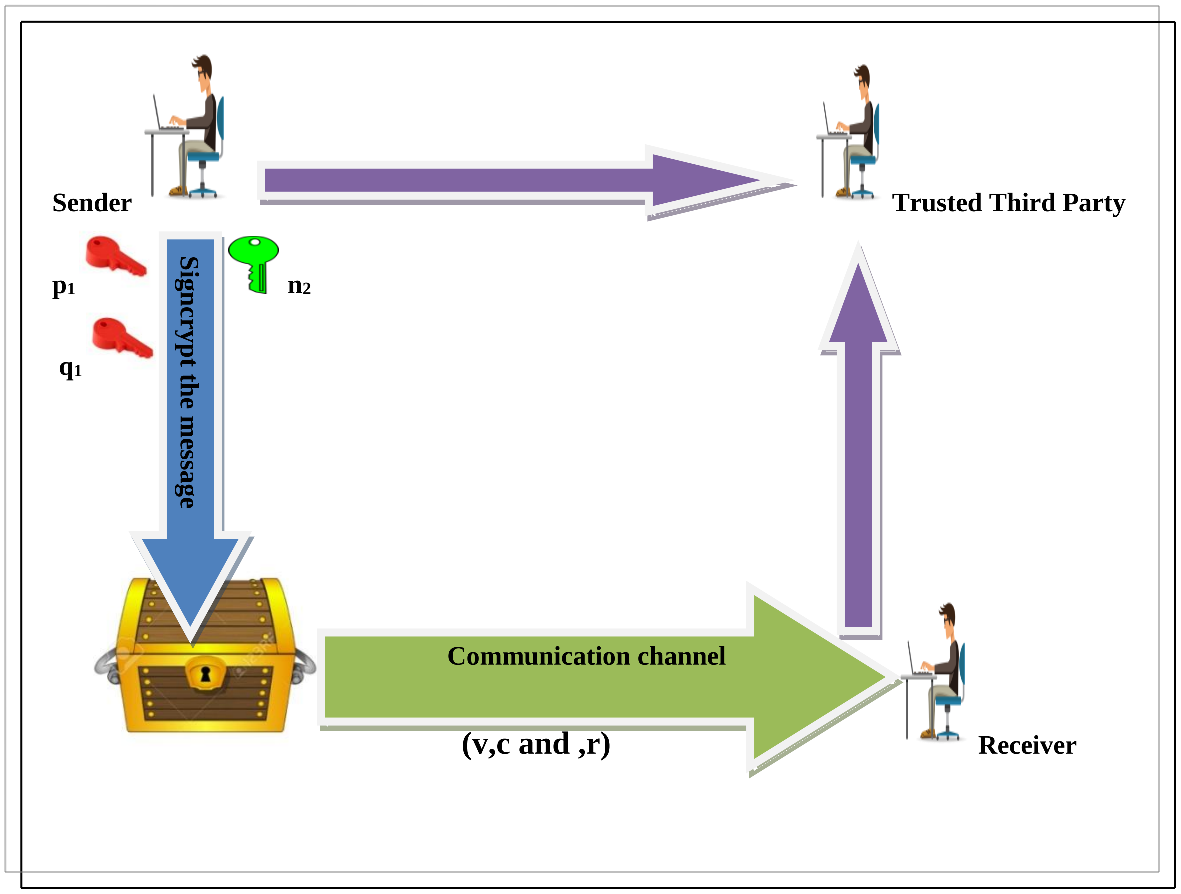


Figure 4.7: Non-repudiation of the Proposed Signcryption

5. **Forward Security**: If the message is hacked by knowing the sender's private key, the other previous messages must be maintained from any breach by the unauthorized access. This goal is achieved by changing the private and public keys continuously. Thus, if one of the keys for one message has been breached, other messages will not be affected. The Forward Security is illustrated in Figure 4.8.

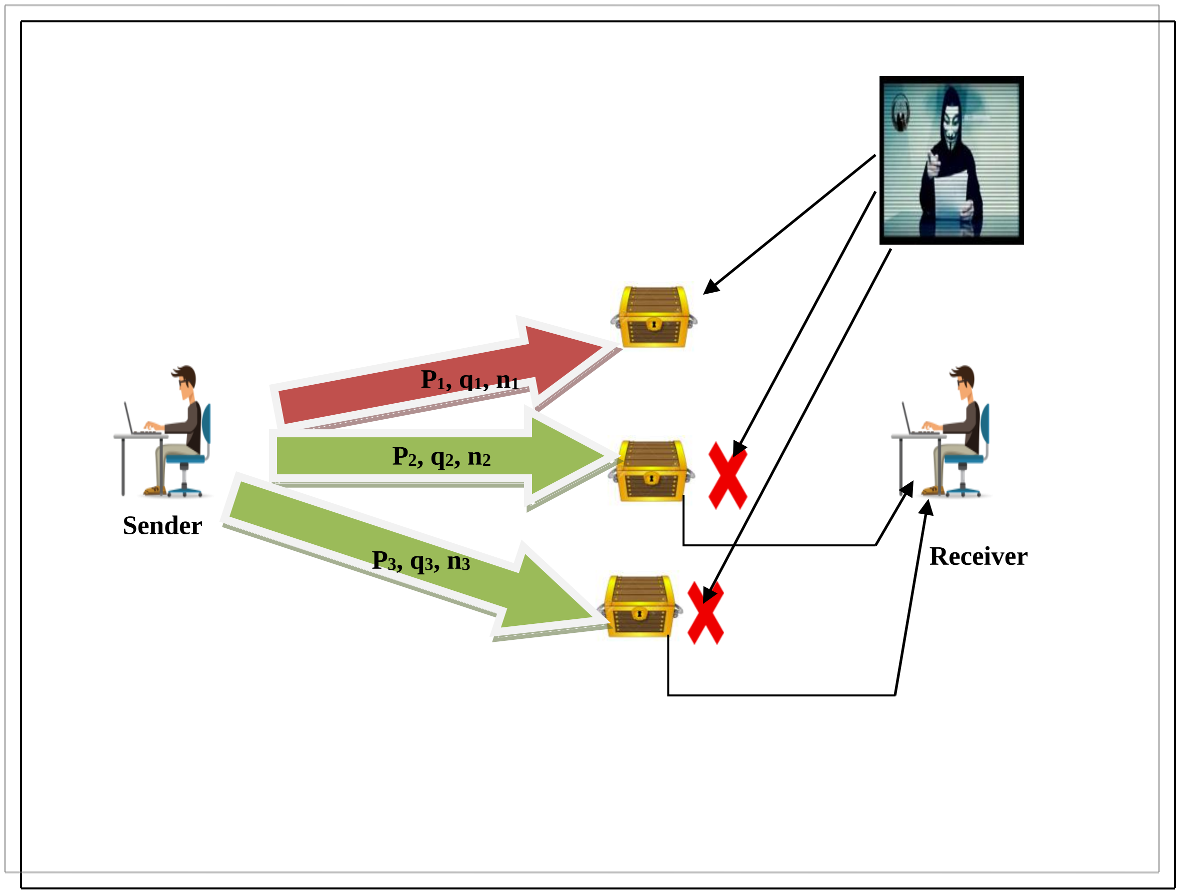


Figure 4.8: Forward Security of the Proposed Signcryption

4.7 Summary

In this chapter, the Proposed Signcryption is implemented in C# by using dataset with different size to show the time for the Proposed Signcryption. The Proposed Signcryption using hashing SHA-1 and Keyed SHA-1 to compute hash function. Rijndael algorithm which is symmetric cryptography used in signcryption and unsigncryption process its more speed than asymmetric cryptography.

The Proposed Signcryption satisfied security concerns confidentially, non-reputation, unforgeability, integrity and forward security. Computational cost for the Proposed Signcryption is calculated for signcryption and unsigncryption process and compared it with the original signcryption; the computational cost in signcryption process is lesser than the computational cost in the original signcryption (SDSS1 and SDSS2). In the unsigncryption process the computational cost for the Proposed Signcryption is higher than the computational cost in the original signcryption (SDSS1 and SDSS2).

In the Proposed Signcryption at sender side the best result for text files with size 4911 byte which consume 416 milliseconds. The worst result for text files with size 465662 byte which consume 523 milliseconds and the average for signcryption execution time are 472.9 milliseconds. In the receiver side the best result for text files with size 4911 byte which consume 656 milliseconds. The worst result for text files with size 465662 byte which consume 800 milliseconds and the average for signcryption execution time are 730milliseconds. In the full signcryption process the best result for text files with size 4911 byte which consume 1072 milliseconds. The worst result for text files with size 465662 byte which consume 1323 milliseconds and the average for signcryption execution time are 1202.6milliseconds. The Proposed Signcryption could be utilized in many applications such as which need fast encryption such as Smart Cards and Wireless Sensor Network (WSN).

Chapter Five

Conclusions and Future Work

The main objective of this thesis is to build and implement new type of signcryption based on Rabin cryptosystem, which has many security features adding to the signcryption and leads to better results.

5.1 Conclusion

Signcryption process combines between digital signature and public key encryption to achieve better results than their own work alone, Rabin cryptosystem is used to build new signcryption to get good results depending on Rabin properties in fast encryption building.

In this thesis, a new signcryption system based on Rabin cryptosystem is proposed; the effectiveness of the new signcryption has been measured and compared with two forms of E-Gamal-Based Signcryption (SDSS1 &SDSS2).

The new signcryption satisfied security concerns, and these are: confidentiality, unforgeability, non-repudiation, integrity, and forward secrecy. These security features of the signcryption have been proven on the basis of this new signcryption, and based on hardness of factorization problem, and so on.

The efficiency of the Proposed Signcryption is better than the efficiency of the original signcryption (SDSS1 and SDSS2) in the signcryption process. The performance of the Proposed Signcryption is very high in the sender side it saved 24.1% of the time compared to the original signcryption (SDSS1 & SDSS2). But, in the unsigncryption process by the receiver, the effectiveness of the Proposed Signcryption is low it takes 18.3% more time than the original signcryption (SDSS1 & SDSS2). In full execution time the Proposed Signcryption satisfied better efficiency it saved 3% of the time compared to the original signcryption (SDSS1 & SDSS2). The Proposed Signcryption suitable for many applications such as Smart Cards and Wireless Sensor Network (WSN).

5.2 Recommendations and Future Work

For this thesis, the researcher suggested the following recommendations:

- Build new signcryption that combines between discrete logarithm problem and factorization problem to achieve the best through them. This combination achieves enhancement in signcryption security and efficiency at the same time, which is not achieved in the proposed signcryption.
- Build new Rabin based signcryption, use it to encrypt images rather than abstract texts only.

References

Agrawal, M., & Mishra, P. (2012). A comparative survey on symmetric key encryption techniques. *International Journal on Computer Science and Engineering, 4*(5), 877.

Amounas, F., & Kinani, E. H. (2013). An Efficient Signcryption Scheme based on the Elliptic Curve Discrete Logarithm Problem. *International Journal of Information and Network Security*, *2(3), 253.*

Ayele, A. A., & Sreenivasarao, V. (2013). A Modified RSA Encryption Technique Based on Multiple public keys. *International Journal of Innovative Research in Computer and Communication Engineering*, *1*(4).

Bakhtiari, M., & Maarof, M. A. (2012). Serious Security Weakness in RSA Cryptosystem. *IJCSI International Journal of Computer Science*, *9*(3).

Bakhtiari, S., Safavi-Naini, R., & Pieprzyk, J. (1995). Cryptographic hash functions: A survey. *Centre for Computer Security Research, Department of Computer Science, University of Wollongong, Australia.*

Bao, F., & Deng, R. H. (1998). A signcryption scheme with signature directly verifiable by public key. In *International Workshop on Public Key Cryptography*, 55-59. Springer Berlin Heidelberg.

Bellare, M., Boldyreva, A., Desai, A., & Pointcheval, D. (2001). Key-privacy in public-key encryption. In *International Conference on the Theory and Application of Cryptology and Information Security*, 566-582. Springer Berlin Heidelberg.

Boyko, V., MacKenzie, P., & Patel, S. (2000). Provably secure password-authenticated key exchange using Diffie-Hellman. In *International Conference on the Theory and Applications of Cryptographic Techniques*, 156-171. Springer Berlin Heidelberg.

Chakraborty, R., Biswas, S., & Mandal, J. K. (2014). Modified Rabin Cryptosystem through Advanced Key Distribution System. *IOSR Journals (IOSR Journal of Computer Engineering), 1*(16), 1-7.

Chennuri, V. K, & Manchala, S. (2015). A Survey on New Approaches in Signcryption. *International Journal of Research and Applications.*

Ciss, A. A., Cheikh, A. Y., & Sow, D. (2013). A Factoring and Discrete Logarithm based Cryptosystem. *International Journal of Contemporary Mathematical Sciences, 8(11), 511-517.*

Das, S., & Samal, B. (2013). An Elliptic Curve based Signcryption Protocol using Java. *International Journal of Computer Applications*. *66*(4).

Dent, A. W. (2005). Hybrid signcryption schemes with insider security. In *Australasian Conference on Information Security and Privacy*, 253-266. Springer Berlin Heidelberg.

Dent, A. W. (2005). Hybrid signcryption schemes with outsider security. In *International Conference on Information Security*, 203-217. Springer Berlin Heidelberg.

Elia, M., & Schipani, D. (2011). On the Rabin signature. *Journal of Discrete Mathematical Sciences and Cryptography*, *16*(6), 367-378.

Elshobaky, A., Elkabbany, G., Rasslan, M., & Gurguis, S. (2014). Implementation, Comparison, and Enhancement of Secure Communication Designs. *Procedia Computer Science*, *(37)*, 363-369.

Galbraith, S. D. (2012). *Mathematics of public key cryptography*. United States of America, New York: Cambridge University Press.

Gupta, P., & Kumar, S. (2014). A Comparative Analysis of SHA and MD5 Algorithm. Architecture. *International Journal of Computer Science and Information Technologies, 5 (3), 4492-4495.*

Gupta, R. K., & Singh, P. (2013). A new way to design and implementation of hybrid crypto system for security of the information in public network. *International Journal of Emerging Technology and Advanced Engineering*, *3*(8), 108-115.

Hashim, H. R. (2014). H-Rabin cryptosystem. *Journal of Mathematics and Statistics*, *10*(3), 304.

Hua, S., Li, G., & Aimin, W. (2011). An efficient identity-based ring signcryption scheme without random oracles. In *International Conference on Computer and Electrical Engineering 4th (ICCEE 2011)*. ASME Press.

Hwang, M. S., Chang, C. C., & Hwang, K. F. (2002). An ElGamal-like cryptosystem for enciphering large messages. *IEEE Transactions on Knowledge and Data Engineering*, *14*(2), 445-446.

Hwang, R. J., Lai, C. H., & Su, F. F. (2005). An efficient signcryption scheme with forward secrecy based on elliptic curve. *Applied Mathematics and computation*, *167*(2), 870-881.

Jamgekar, R. S., & Joshi, G. S. (2013). File encryption and decryption using secure RSA. *International Journal of Emerging Science and Engineering (IJESE)*, *1*(4), 11-14.

Jun-Bao, L., & Guo-Zhen, X. (2005). Multi-proxy multi-signcryption scheme from pairings. *arXiv preprint cs/0509030*.

Kirtane, V., & Rangan, C. P. (2008). RSA-TBOS signcryption with proxy re-encryption. In *Proceedings of the 8th ACM workshop on Digital rights management* (pp. 59-66). ACM.

Koblitz, A. H., Koblitz, N., & Menezes, A. (2011). Elliptic curve cryptography: The serpentine course of a paradigm shift. *Journal of Number theory*, *131*(5), 781-814.

Kumar, V. (2014). **Efficient identity based signcryption scheme and solution of key-escrow problem** (Doctoral dissertation). National Institute of Technology, Odisha, India.

Libert, B., & Quisquater, J. J. (2004). Efficient signcryption with key privacy from gap Diffie-Hellman groups. In *International Workshop on Public Key Cryptography* (pp. 187-200). Springer Berlin Heidelberg.

Li, X., & Chen, K. (2004). Identity based proxy-signcryption scheme from pairings. In *Services Computing, 2004.(SCC 2004). Proceedings. 2004 IEEE International Conference on (pp. 494-497). IEEE.*

Mambo, M., & Okamoto, E. (1997). Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. *IEICE transactions on fundamentals of electronics, Communications and computer sciences*, *80*(1), 54-63.

Mao, W., & Malone-Lee, J. (2002). Two Birds One Stone: Signcryption using RSA. *Progress in Cryptology—CT-RSA . HP Laboratories Bristol.*

Mohamed, E., & Elkamchouchi, H. (2009). Elliptic curve signcryption with encrypted message authentication and forward secrecy. *International Journal of Computer Science and Network Security*, *9*(1), 395-398.

Malone-Lee, J. C. (2004). **On the Security of Signature Schemes and Signcryption Schemes** (Doctoral dissertation), University of Bristol, Bristol, England.

Molale, T. (2005). Mathematical Foundation of Public Key Cryptosystems. In *Lectures SE 4C03 – Computer Networks and Computer Security*.

Nayak, R. (2015). Analysis of Rabin and Rabin-P Cryptosystem for specifying correct plain text. *IJRCCT*, *4(5),* 354-357.

Nicolas T. Courtois. (2006). Public *Key Cryptography*. London, England: University College of London.

Noorouzi, E., Haghighi, A. R. E. A., Peyravi, F., & Zadeh, A. K. (2009). A New Digital Signature Algorithm. *International Conference on Machine Learning and Computing IPCSIT, 3 (2011).*

Petitcolas, F. A., Anderson, R. J., & Kuhn, M. G. (1999). Information hiding-a survey. *Proceedings of the IEEE, 87*(7), 1062-1078.

Pointcheval, D. (2001). Practical security in public-key cryptography. *In International Conference on Information Security and Cryptology. 1-17.* Springer Berlin Heidelberg.

Preneel, B. (1999). The state of cryptographic hash functions. In *Lectures on Data Security, 158-182.* Springer Berlin Heidelberg.

Roy, A., & Karforma, S. (2012). A Survey on digital signatures and its applications. *Journal of Computer and Information Technology, 3*(1), 45-69.

Savu, L. (2012). Signcryption scheme based on schnorr digital signature. *arXiv preprint arXiv:1202.1663.*

Singh, A. K. (2014). A Review of Elliptic Curve based Signcryption Schemes. *International Journal of Computer Applications, 102*(6).

Singh, A. K., & Patro, B. D. K. (2017) Performance Comparison of Signcryption Schemes–A Step towards Designing Lightweight Cryptographic Mechanism. *International Journal of Engineering and Technology (IJET)*

Singh, G. (2013). A study of encryption algorithms (RSA, DES, 3DES and AES) for information security. *International Journal of Computer Applications*, *67*(19).

Sidorov, E., & Yandex, L. L. C. (2015). Breaking the Rabin-Williams digital signature system implementation in the Crypto++ library. *IACR Cryptology ePrint Archive*, 368.

Srivastava, A. K., & Mathur, A. (2013). The Rabin Cryptosystem & analysis in measure of Chinese Reminder Theorem. International Journal of Scientific and Research Publications, 493.

Toorani, M., & Beheshti, A. (2008). Cryptanalysis of an efficient signcryption scheme with forward secrecy based on elliptic curve. In *Computer and Electrical Engineering, 2008. ICCEE 2008. International Conference on*. *IEEE, 428-432*.

Youngblood, C. (2005). An introduction to identity-based cryptography. *CSEP 590TU*.

Zheng, Y., & Imai, H. (1998). How to construct efficient signcryption schemes on elliptic curves. *Information Processing Letters*, 68(5), 227-233.

Zheng, Y. (1997). Digital signcryption or how to achieve cost (signature & encryption) ≪ cost (signature) + cost (encryption). In *Annual International Cryptology Conference, 165-179*. Springer Berlin Heidelberg.

Zheng, Y. (1997). Signcryption and its applications in efficient public key solutions. *In Proceedings of 1997 Information Security Workshop ISW, 291-312.*

Zhu, H., & Li, D. (2008). Research on Digital Signature in Electronic Commerce. In *proceedings of the International Multi Conference of Engineers and Computer Scientists* (Vol. 1).

Appendix A: Dataset

file_no	File Name	File Size	
1	at&arebl.txt	2965	Byte
2	ddn01.txt	2993	Byte
3	cheap144disk.txt	2996	Byte
4	arc.txt	3025	Byte
5	muf161.txt	3044	Byte
6	hackprod.txt	3110	Byte
7	zipwarn.txt	4096	Byte
8	govthlp.txt	4106	Byte
9	datatap.txt	4109	Byte
10	zappvowl.txt	4112	Byte
11	tec016.txt	4129	Byte
12	tec002.txt	4131	Byte
13	anywhere.txt	4142	Byte
14	honey.txt	4144	Byte
15	wal-mart.txt	4175	Byte
16	modbook2.txt	4182	Byte
17	equation.txt	4200	Byte
18	bindery.txt	4201	Byte
19	hackrsts.txt	4221	Byte
20	pcjrmem.txt	4224	Byte
21	chaos03.txt	4264	Byte
22	hackterm.txt	4314	Byte
23	uucpmap81.txt	4317	Byte
24	alterna2.txt	4339	Byte
25	hackholl.txt	4372	Byte
26	sharew.txt	4373	Byte
27	lca-1.txt	4422	Byte
28	thumb.txt	4432	Byte
29	2w93358a.txt	4445	Byte
30	modbook3.txt	4522	Byte
31	hackingc.txt	4524	Byte
32	acro01.txt	4538	Byte
33	gif_info.txt	4597	Byte
34	caution.txt	4613	Byte
35	janet.txt	4618	Byte
36	getbust.txt	4631	Byte
37	vmbhack.txt	4662	Byte

38	qsd.txt	4676	Byte
39	easylink.txt	4686	Byte
40	hdindex.txt	4686	Byte
41	hackl1.txt	4699	Byte
42	nha-app.txt	4706	Byte
43	sourcetelnet.txt	4828	Byte
44	dialplus.txt	4830	Byte
45	tec014.txt	4864	Byte
46	webwar.txt	4911	Byte
47	master.txt	4915	Byte
48	hacker03.txt	4931	Byte
49	atlas.txt	4938	Byte
50	31.txt	4986	Byte
51	22.txt	4999	Byte
52	drives.txt	5060	Byte
53	lca-5.txt	5082	Byte
54	ss-info2.txt	5120	Byte
55	mickeyd.txt	5145	Byte
56	modbook5.txt	5199	Byte
57	serial.txt	5207	Byte
58	contact.txt	5246	Byte
59	fbicompu.txt	5246	Byte
60	hackad.txt	5248	Byte
61	fbisys.txt	5281	Byte
62	fbisys.txt	5281	Byte
63	fast.txt	5301	Byte
64	tec022.txt	5341	Byte
65	amickpt.txt	5404	Byte
66	shrware.txt	5420	Byte
67	pw-hack.txt	5432	Byte
68	timenet.txt	5434	Byte
69	privacy.txt	5474	Byte
70	srbediff.txt	5476	Byte
71	autohack.txt	5504	Byte
72	mrdos4.txt	5527	Byte
73	acro02.txt	5532	Byte
74	scanprg.txt	5542	Byte
75	tec021.txt	5550	Byte
76	386486.txt	5589	Byte
77	response.txt	5623	Byte
78	hacktips.txt	5625	Byte
79	hacksong.txt	5632	Byte

80	zippass.txt	5637	Byte
81	c64topc.txt	5646	Byte
82	supdev.txt	5654	Byte
83	asn_temp.txt	5738	Byte
84	style.txt	5738	Byte
85	atmhacking.txt	5742	Byte
86	36.txt	5747	Byte
87	mrdos2.txt	5753	Byte
88	cbvhack.txt	5754	Byte
89	bios&mb.txt	5755	Byte
90	autonet3.txt	5764	Byte
91	javabugs.txt	5767	Byte
92	hacker.txt	5787	Byte
93	cache.txt	5795	Byte
94	hackinga.txt	5856	Byte
95	homebank.txt	5863	Byte
96	statemind.txt	5863	Byte
97	balli.txt	5888	Byte
98	trash.txt	5888	Byte
99	mactricks.txt	5942	Byte
100	hacethi.txt	5949	Byte
101	hackethic.txt	5953	Byte
102	autonet4.txt	5973	Byte
103	getinfo.txt	5978	Byte
104	noise_1.txt	6020	Byte
105	noise.txt	6022	Byte
106	iiu-001.txt	6073	Byte
107	xit.txt	6108	Byte
108	q88164.txt	6110	Byte
109	hashish.txt	6146	Byte
110	hacethic.txt	6161	Byte
111	144disk.txt	6178	Byte
112	risks.txt	6181	Byte
113	security.txt	6184	Byte
114	cisagain.txt	6192	Byte
115	citibank.txt	6226	Byte
116	demo3.txt	6288	Byte
117	compserv.txt	6337	Byte
118	m200ram.txt	6363	Byte
119	citibank2.txt	6430	Byte
120	teleinfo.txt	6496	Byte
121	amscsi.txt	6497	Byte

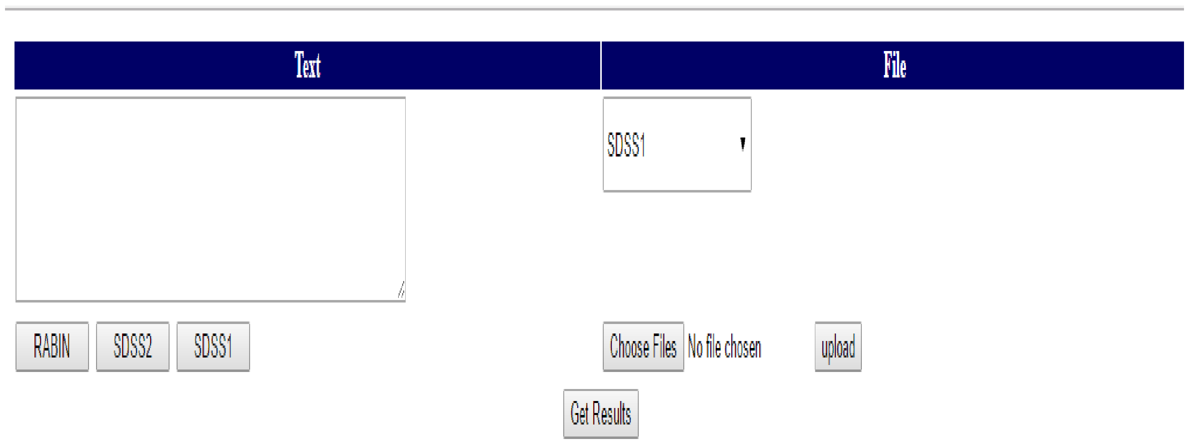
122	earlybst.txt	6529	Byte
123	altgroup.txt	6546	Byte
124	aio5b.txt	6588	Byte
125	basictip.txt	6588	Byte
126	may-bust.txt	6617	Byte
127	boahack.txt	6686	Byte
128	mnemonic.txt	6705	Byte
129	defaults.txt	6726	Byte
130	bootfromdf1.txt	6757	Byte
131	cascade.txt	6784	Byte
132	144_ctrl.txt	6819	Byte
133	photoscn.txt	6901	Byte
134	ftp-help.txt	6924	Byte
135	freeware.txt	6933	Byte
136	boces.txt	6957	Byte
137	Adventur	6975	Byte
138	500mm.txt	6983	Byte
139	force6.txt	7027	Byte
140	ansikode.txt	7033	Byte
141	maccrack.txt	7038	Byte
142	news_dw.txt	9186	Byte
143	aolhak.txt	9252	Byte
144	school.txt	9253	Byte
145	his-hp.txt	9320	Byte
146	28_8khst.txt	9335	Byte
147	Biblio	9338	Byte
148	perstest.txt	9361	Byte
149	vthack2.txt	9378	Byte
150	mrdos1.txt	9381	Byte
151	dpacbas.txt	9425	Byte
152	tec023.txt	9441	Byte
153	bd_ch5.txt	9481	Byte
154	biprint.txt	9485	Byte
155	whatsnew.txt	9526	Byte
156	pcphack.txt	9537	Byte
157	c-easy.txt	9573	Byte
158	exam1_65.txt	9588	Byte
159	ibmbios.txt	14384	Byte
160	teln0418.txt	14410	Byte
161	datapac.txt	14413	Byte
162	hdigest.txt	14413	Byte
163	dts.txt	14481	Byte

164	shell_history.txt	14566	Byte
165	tnet3.txt	14592	Byte
166	austpac0.txt	14674	Byte
167	dma_rti.txt	14835	Byte
168	us_domai.txt	14858	Byte
169	longpass.txt	14916	Byte
170	hung.txt	15203	Byte
171	ipg.txt	15204	Byte
172	ais.txt	15248	Byte
173	innerc.txt	15295	Byte
174	natural.txt	15511	Byte
175	jul93blt.txt	15600	Byte
176	datapac2.txt	19480	Byte
177	bot.txt	19481	Byte
178	genlock.txt	19649	Byte
179	scolicen.txt	19800	Byte
180	ssn-stuf.txt	19840	Byte
181	gte.txt	19883	Byte
182	hacker1.txt	19996	Byte
183	datapac4.txt	20024	Byte
184	atm-more.txt	20049	Byte
185	atm-92.txt	20096	Byte
186	handbook.txt	20162	Byte
187	pcl100.txt	20296	Byte
188	bigfun.txt	24960	Byte
189	mag_stripes.txt	25011	Byte
190	primos4.txt	25184	Byte
191	foregole.txt	25563	Byte
192	desdebug.txt	25598	Byte
193	buyguide.txt	25636	Byte
194	battery.txt	26055	Byte
195	bd_ch4.txt	29582	Byte
196	hackcrak.txt	29696	Byte
197	force3.txt	29702	Byte
198	sw56gu.txt	30052	Byte
199	ph.txt	30181	Byte
200	gfxhints.txt	30197	Byte
201	gitr01.txt	30264	Byte
202	acroynym.txt	30296	Byte
203	force2.txt	30310	Byte
204	how2mnp.txt	30321	Byte
205	arthayes.txt	30471	Byte

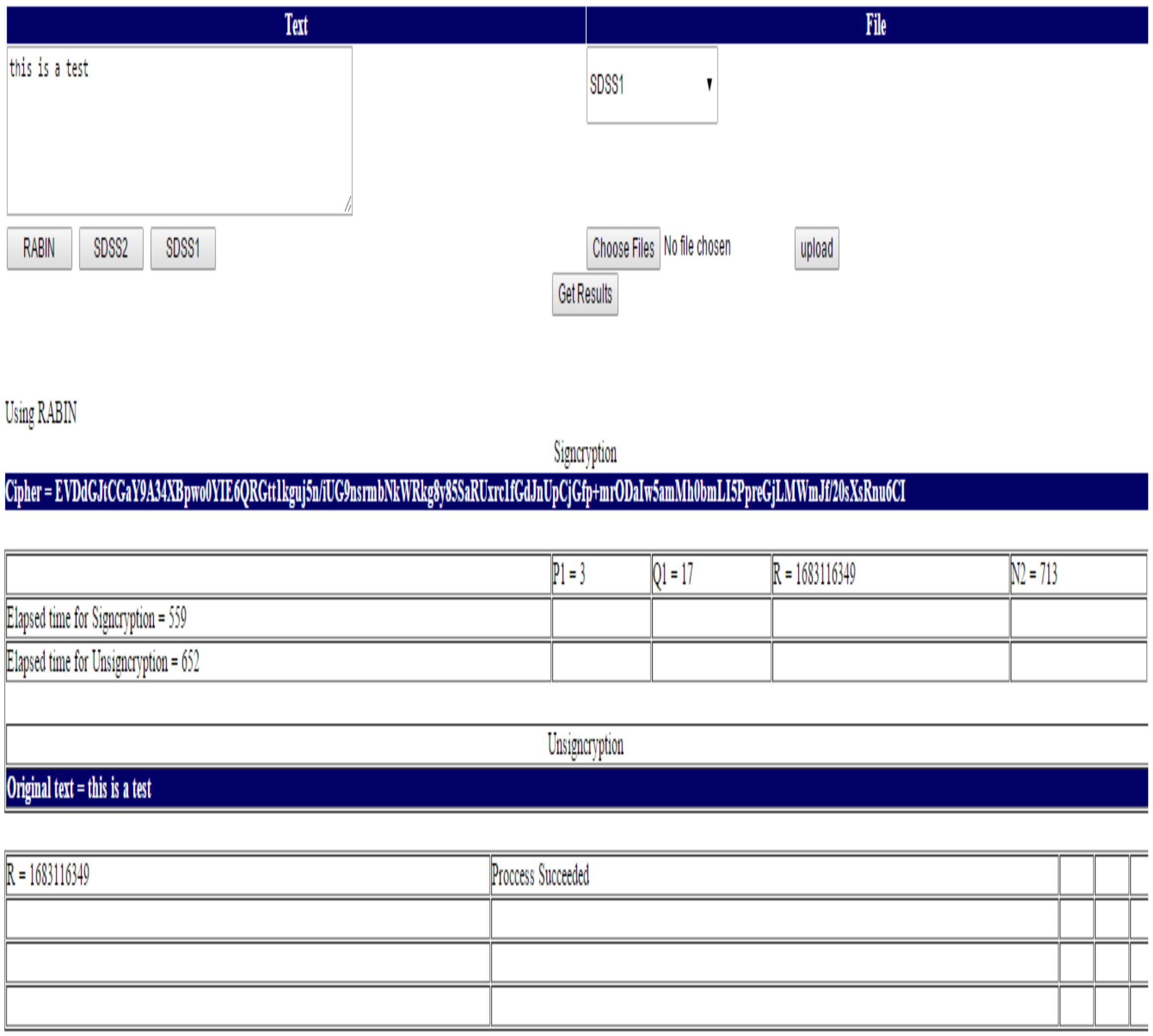
206	svgatrix.txt	30553	Byte
207	unp.txt	30663	Byte
208	hacking.txt	30670	Byte
209	tcp-ip.txt	35765	Byte
210	dosmnual.txt	42543	Byte
211	source12.txt	43303	Byte
212	english.txt	43375	Byte
213	usenet.txt	44160	Byte
214	equip.txt	45113	Byte
215	modem.txt	45650	Byte
216	sp4rpt.txt	45895	Byte
217	bd_ch3.txt	45939	Byte
218	500hacks.txt	46075	Byte
219	bd_ch6.txt	50298	Byte
220	active.txt	50738	Byte
221	batch.txt	51183	Byte
222	nixpub.txt	55687	Byte
223	arcsuit.txt	56430	Byte
224	iirgacr6.txt	62629	Byte
225	diac92.txt	62740	Byte
226	m100quic.txt	63135	Byte
227	act-13.txt	63155	Byte
228	softshop.txt	63378	Byte
229	crackam1.txt	63466	Byte
230	havok1.txt	64006	Byte
231	iirgacr7.txt	67165	Byte
232	dialout1.txt	67840	Byte
233	varian1.txt	71070	Byte
234	slipdoc.txt	80271	Byte
235	zine0494.txt	81657	Byte
236	6502.txt	84034	Byte
237	ncsc-tg-003.txt	86792	Byte
238	crc.txt	91339	Byte
239	weird2_1.txt	101705	Byte
240	asm.txt	101753	Byte
241	tr823.txt	106655	Byte
242	scsidefs.txt	113040	Byte
243	electrop.txt	115151	Byte
244	iah1.txt	142976	Byte
245	hacker01.txt	152594	Byte
246	ftpsites.txt	153042	Byte
247	cuthesis.txt	154635	Byte

248	mdic200.txt	155549	Byte
249	unixsec.txt	157190	Byte
250	zeninternet.txt	158659	Byte
251	cihac009.txt	218045	Byte
252	interest.txt	262144	Byte
253	essays.txt	263347	Byte
254	orange.txt	319321	Byte
255	hayes.txt	322645	Byte
256	begunix.txt	337256	Byte
257	asttechnical.txt	342270	Byte
258	courierv34man.txt	346655	Byte
259	vendors1.txt	389431	Byte
260	dummy20.txt	408942	Byte
261	cguide_3.txt	440893	Byte
262	netguide.txt	460283	Byte
263	wholegui.txt	499973	Byte

Appendix B: Interface



This is the main page which allows you to insert a text in the textbox and choose the button below to signcrypt the string using Rabin or SDSS1 or SDSS2 algorithm.



this is a test

Using RABIN

Signcryption

Cipher = EVDdGJtCGaY9A34XBpwo0YIE6QRGtt1kguj5n/iUG9nsrmbNkWRkg8y85SaRUxrc1fGdJnUpCjGfp+mrODaIw5amMh0bmLI5PpreGjLMWmJf/20sXsRnu6CI

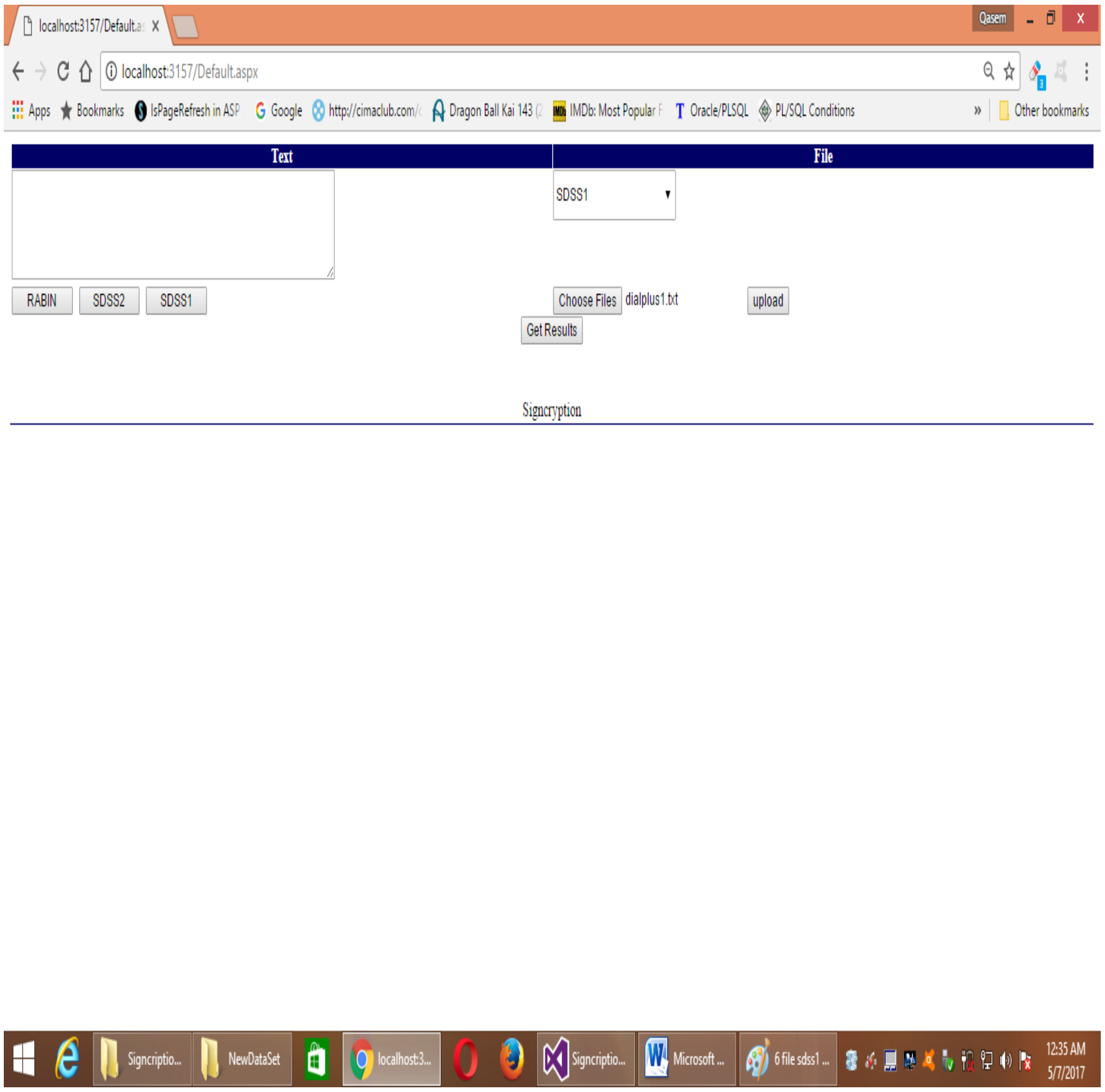
	P1 = 3	Q1 = 17	R = 1683116349	N2 = 713
Elapsed time for Signcryption = 559				
Elapsed time for Unsigncryption = 652				

Unsigncryption

Original text = this is a test

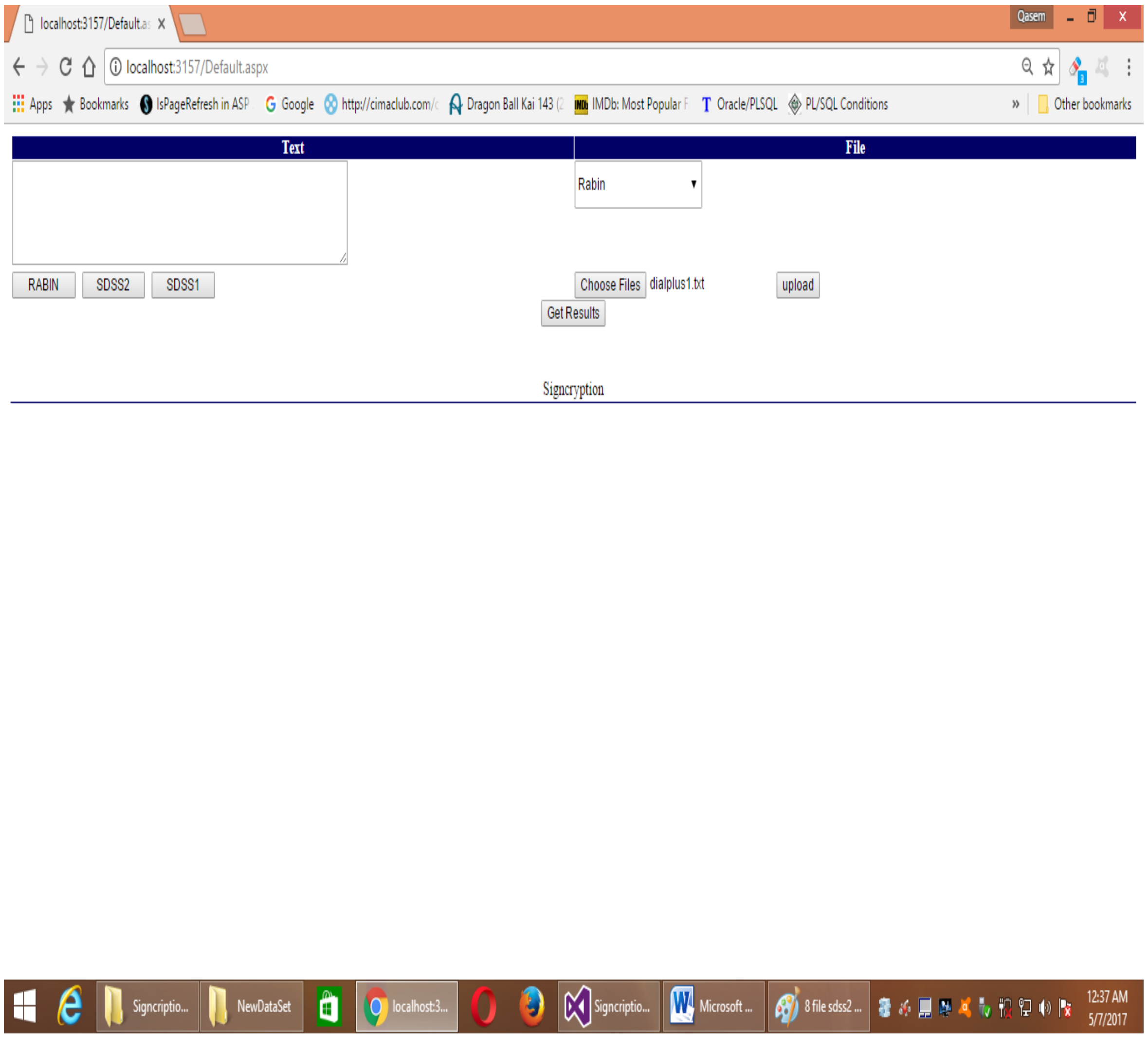
R = 1683116349	Process Succeeded			

This is the result of the signcryption and unsigncryption using Rabin

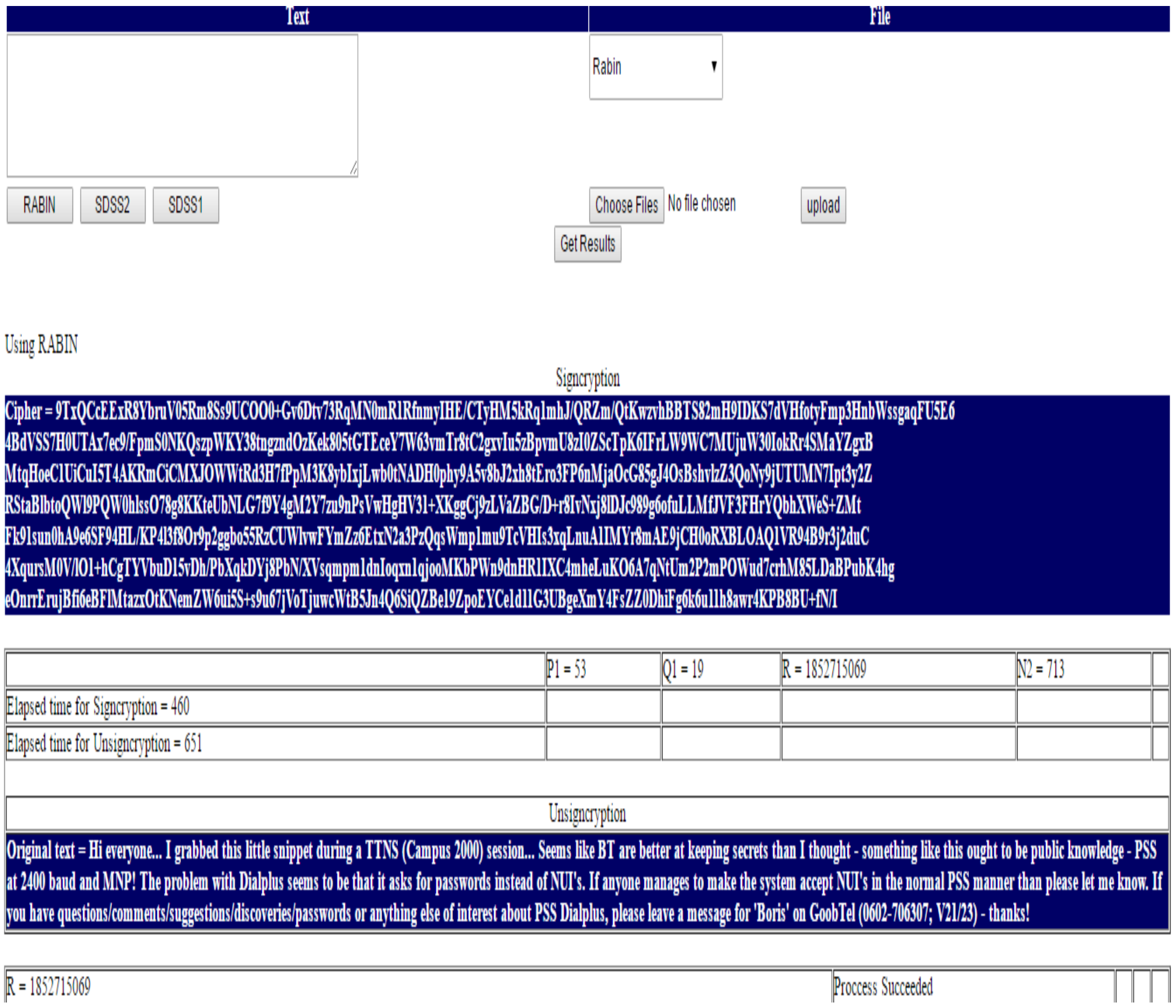


Here we upload a file from the dataset of files we downloaded from the internet, and then we choose the algorithm from the dropdown list and press the upload button to view the result as shown below.

Finally choose the Rabin algorithm



After we press the upload button we see the result like this screen



Using RABIN

Signcryption

```
Cipher = 9TxQCcEExR8YbruV05Rm8Ss9UCOO0+Gv6Dtv73RqMN0mR1RfnmyIHE/CTyHM5kRq1mhJ/QRZm/QtKwzvhBBTS82mH9IDKS7dVHfotyFmp3HnbWssgaqFU5E6
4BdVSS7H0UTAx7ec9/FpmS0NKQszpWKY38tngzndOzKek805tGTEceY7W63vmTr8tC2gxvIu5zBpvmU8zI0ZScTpK6IFrLW9WC7MUjuW30IokRr4SMaYZgxB
MtqHoeC1UiCuI5T4AKRmCiCMXJOWWtRd3H7fPpM3K8ybIxjLwb0tNADH0phy9A5v8bJ2xh8tEro3FP6nMjaOcG85gJ4OsBshvlzZ3QoNy9jUTUMN7Ipt3y2Z
RStaBlbtoQWl9PQW0hlssO78g8KKteUbNLG7f9Y4gM2Y7zu9nPsVwHgHV31+XKggCj9zLVaZBG/D+r8IvNxj8lDJc989g6ofuLLMfJVF3FHrYQbhXWeS+ZMt
Fk91sun0hA9e6SF94HL/KP4l3f8Or9p2ggbo55RzCUWlvwFYmZz6EtxN2a3PzQqsWmp1mu9TcVHIs3xqLnuA1IMYr8mAE9jCH0oRXBLOAQ1VR94B9r3j2duC
4XqursM0V/lO1+hCgTYVbuD15vDh/PbXqkDYj8PbN/XVsqmpm1dnIoqxn1qjooMKbPWn9dnHR1IXC4mheLuKO6A7qNtUm2P2mPOWud7crhM85LDaBPubK4hg
eOnrrErujBfi6eBFlMtazxOtKNemZW6ui5S+s9u67jVoTjuwcWtB5Jn4Q6SiQZBel9ZpoEYCe1dl1G3UBgeXmY4FsZZ0DhiFg6k6u1lh8awr4KPB8BU+fN/I
```

	P1 = 53	Q1 = 19	R = 1852715069	N2 = 713	
Elapsed time for Signcryption = 460					
Elapsed time for Unsigncryption = 651					

Unsigncryption

Original text = Hi everyone... I grabbed this little snippet during a TTNS (Campus 2000) session... Seems like BT are better at keeping secrets than I thought - something like this ought to be public knowledge - PSS at 2400 baud and MNP! The problem with Dialplus seems to be that it asks for passwords instead of NUI's. If anyone manages to make the system accept NUI's in the normal PSS manner than please let me know. If you have questions/comments/suggestions/discoveries/passwords or anything else of interest about PSS Dialplus, please leave a message for 'Boris' on GoobTel (0602-706307; V21/23) - thanks!

R = 1852715069	Process Succeeded			