

A Novel Blind Signature Scheme Based On Discrete Logarithm Problem With Un-traceability

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology

**In
Computer Science & Engineering**

By

**Biswa Bhusan Biswal
Roll No: 108CS040**

**Sukanta Kumar Mangal
Roll No: 108CS032**

**Under the guidance of
Prof. Sujata Mohanty**



**Department of Computer Science & Engineering
National Institute of Technology Rourkela
2012**

A Novel Blind Signature Scheme Based On Discrete Logarithm Problem With Un-traceability

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology

In Computer Science & Engineering

By

Biswa Bhusan Biswal
Roll No: 108CS040

Sukanta Kumar Mangal
Roll No: 108CS032



Department of Computer Science & Engineering
National Institute of Technology Rourkela
2012



**NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA
CERTIFICATE**

This is to certify that the thesis entitled, “**A Novel Blind Signature Scheme Based On Discrete Logarithm Problem With Un-traceability**” by **Mr. Biswa Bhusan Biswal & Mr. Sukanta Kumar Mangal** in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in *Computer Science & Engineering* at the National Institute of Technology, Rourkela is an authentic work carried out by them under my supervision and guidance.

To the best of my knowledge the matter embodied in the thesis has not been submitted to any other University/ Institute for the award of any Degree or Diploma.

Date:

Prof. Sujata Mohanty

Dept. of Computer Science & Engineering

National Institute of Technology

Rourkela-769008

ACKNOWLEDGEMENT

We take this opportunity to express our deepest sense of gratitude to our guide **Prof. Sujata Mohanty** , Computer Science & Engineering Department , for her valuable guidance , support , constant encouragement and kind help at different stages , for the execution & completion of this dissertation work .

We also express our sincere gratitude to **Prof. A.K. Turuk** , Head of the Department , Computer Science & Engineering , for providing valuable departmental facilities .

Submitted by:

Biswa Bhusan Biswal

Roll No: 108CS040

Sukanta Kumar Mangal

Roll No: 108CS032

Abstract

Blind Signatures are a special type of digital signatures which possess *two* special properties of *blindness* and *untraceability*, which are important for today's real world applications that require authentication , integrity , security , anonymity and privacy.

David Chaum was the first to propose the concept of blind signatures. The scheme's security was based on the difficulty of solving the *factoring problem*. Two properties that are important for a blind signature scheme in order to be used in various modern applications are blindness and untraceability. *Blindness* means that the signer is not able to know the contents of the message while signing it, which is achieved by disguising (or blinding) the message through various methods. *Untraceability* refers to preventing the signer from linking the blinded message it signs to a later unblinded version that it may be called upon to verify.

Blind signatures based on discrete logarithm problem are still an area with much scope for research. We aim to propose a novel blind signature scheme with untraceability , based on the discrete logarithm problem.

Contents

1	Introduction	7
1.1	Motivation	9
2	Literature Survey	10
2.1	Cryptography	11
2.1.1	Symmetric Key Cryptography	11
2.1.2	Asymmetric Key Cryptography	12
2.1.3	Hashing	12
2.2	Cryptanalysis	12
2.3	Security Services	13
2.3.1	Data Confidentiality	13
2.3.2	Data Integrity	13
2.3.3	Authentication	14
2.3.4	Non-Repudiation	14
2.3.5	Access Control	14
2.4	Digital Signature	15
2.5	Blind Signature	17
2.5.1	Definition	17
2.5.2	Properties	18
2.5.3	Mechanism	18
2.5.4	Applications	19
2.6	Cryptographic Background	20
2.6.1	Random Number Generation	20
2.6.2	Primality Test	20

2.6.3	Discrete Logarithm	21
2.6.4	Cryptographic Hash Functions	22
3	Blind Signature Scheme with untraceability	24
3.1	Lee , Hwang & Yang 's scheme	25
3.1.1	Algorithm	25
3.2	Proposed Scheme	27
3.2.1	Algorithm	27
3.2.2	Security Analysis	30
3.2.3	Performance Comparison	31
4	Implementation Details	32
4.1	Lee , Hwang & Yang's scheme's Output	33
4.2	Our Proposed Scheme's Output	34
4.2.1	Key Generation	34
4.2.2	Blinding	34
4.2.3	Signing	35
4.2.4	Unblinding	35
4.2.5	Verification	36
4.3	Comparison with Lee , Hwang & Yang's scheme	36
5	Conclusion	39
	<i>References</i>	41

List of Figures

1	Digital Signature Scheme	16
2	Concept of Blind Signature	17
3	Overall Representation of proposed scheme	29
4	Lee , Hwang & Yang's scheme's Output	33
5	Key Generation Phase	34
6	Blinding Phase	34
7	Signing Phase	35
8	Unblinding Phase	35
9	Verification Phase	36
10	Graphical Representation of the Comparison (Execution time v/s Message Size)	38

List of Tables

1	Comparative study of Computational Complexities	31
2	Execution time (in milliseconds) comparison for a message size of 29404 bytes	36
3	Comparison for various message sizes	37

Chapter 1

Introduction

1 Introduction

A digital signature or digital signature scheme is a mathematical scheme for demonstrating the authenticity of a digital message or document. A valid digital signature gives a recipient or receiver reason to believe that the message was created & sent by a known or trusted sender. Digital signatures are commonly used for software distribution, financial transactions, electronic voting and in other situations where it is important to detect forgery or tampering. Along with authentication, digital signatures also possess the property of integrity, which ensures that the received messages are not manipulated or modified or altered during the transmission of message from sender to receiver . Due to its importance and in order to use it in various kinds of applications, many types of digital signature scheme have been proposed such as blind signature , group signature , undeniable signature etc .

In the field of cryptography, a blind signature scheme, introduced by David Chaum[2]¹ is a special type of digital signature scheme in which the content of a message is hidden or disguised (blinded) before it is signed. The resulting blind signature obtained can be publicly verified against the original unblinded message in the manner of a normal digital signature. Blind signatures are usually used protocols or applications requiring privacy and anonymity , where the signer and message author are two different parties , for example in applications like cryptographic election systems and digital cash schemes.

An often used analogy to the blind signature scheme is the physical act of enclosing a message in a special 'write through' capable envelope, which after being sealed is signed by a signer or signing agent. Thus, the signer can not determine the message content, but a third party can later verify the signature and know that the signature is valid within the limitations of the underlying signature scheme .

Blind signatures can also be used to provide *untraceability* or *unlinkability*, which prevents the signer from linking the blinded message it signs, to a later unblinded version that it may be called upon to verify. In order to achieve this, the signer's response is first unblinded" prior to verification in such a way that the signature remains valid for the unblinded message. This can be useful in schemes or applications where anonymity is required, such as e-voting.

The security of the blind signature scheme proposed by us is based both on the difficulty of solving the discrete logarithm problem as well as the strength of the hash function used in its implementation. Our scheme also has two important properties, untraceability & unforgability, along with the basic properties of any digital signature scheme that are integrity and authentication.

1.1 Motivation

In 1994, *Carmenisch et al.* proposed blind signature schemes based on the discrete logarithm problem (DLP) [7], whose security was based on the difficulty of solving the discrete logarithm problem. Later, *Harn* showed that the blind signatures proposed by *Carmenisch et al.* could not meet the requirement of untraceability[8]. But, *Horster et al.* claimed that *Harn's* cryptanalysis was not correct [9]. When the signer traces the signature, he will obtain two pairs of signed messages that were satisfied by the equation of *Harn's* cryptanalysis. Therefore, the signer cannot trace back to the owner of the signature. However, *Cheng-Chi Lee, Min-Shiang Hwang and Wei-Pang Yang* have shown that *Horster's* comment is improper [1], the signer can record all information when the requester requests the blind signature to the signer and if he wants to know the owner of the signature, he still can use *Harn's* method. *Cheng-Chi Lee, Min-Shiang Hwang and Wei-Pang Yang* have also designed a blind signature scheme which not only overcomes the shortcoming of *Carmenisch et al.'s* scheme, but also achieves the properties, blindness and untraceability [1].

Today, due to the increasing number of applications of blind signatures, how to design a blind signature based on the discrete logarithm for untraceability is still an open question. Our aim is to propose and implement a new novel blind signature scheme based on DLP with untraceability which would be less complex & faster than Lee, Hwang & Yang's scheme [1] by reducing the number of computations as well as keeping them simple, such that our scheme can be used for developing E- Voting, E- Cash and other applications.

Chapter 2

Literature Survey

2. Literature Survey

2.1 Cryptography

Cryptography, in simple terms , can be defined as the practice & study of techniques of converting ordinary (in most cases , meaningful) text into unintelligible gibberish . Here, the ordinary information which gets converted is called as the plain text and the unintelligible output of the conversion is called the cipher text. This technique of conversion is also known as encryption , while the reverse technique of retrieving the original text from the cipher text is called as decryption . Earlier, cryptography mainly focused on these two processes that is encryption & decryption , and aimed at achieving mainly , confidentiality . But , in the modern era , with the development of technology as well as needs of many applications , this field has expanded to include the properties of integrity , non-repudiation , authentication , security etc. along with confidentiality , and as such three different standard mechanisms have been proposed [13,14].

- Symmetric Key Cryptography
- Asymmetric Key Cryptography
- Hashing

2.1.1 Symmetric Key Cryptography

In this type of cryptography, the *same key* is shared between the sender & the receiver, and is used for data transmission between the two. That means, the key used by the sender to encrypt the message, is also used by the receiver to decrypt the encrypted message. Encryption is done by some algorithm as chosen by the sender. Here, the security & confidentiality of the technique lie with the secrecy of the common key, so it has to be shared between the sender & receiver via a secure channel.

2.1.2 Asymmetric Key Cryptography

In this type of cryptography, instead of one key, two keys are used , one is called the *private* key & the other is called the *public* key . Here , the sender uses the public key of the receiver to

encrypt the message and sends it to the receiver , which is then decrypted by the receiver using his private key , in order to obtain the original message , as shown in the figure 1.[13,14] . This technique is also called the *Public key cryptography*.

2.1.3 Hashing

A third type of cryptographic algorithm is also available known as the cryptographic hash functions . Messages of any length can be provided as input, which are used to produce a short fixed length output called as hash or message digest. These are used mainly in digital signatures. For a strong & good hash function, an attacker cannot find two messages which produce the same message digest or hash. Examples include MD4, MD5, SHA-1 family, SHA-2 family, etc.

2.2 Cryptanalysis

Like cryptography includes the techniques of covering, hiding or disguising the original message for achieving confidentiality, cryptanalysis includes the techniques of uncovering the hidden messages, without the information of the secret parameters that are normally required to obtain the message. The goal of cryptanalysis is to find some weakness or insecurity in a cryptographic scheme, in order to crack the code. For this, one has to acquire a good knowledge of the system , how it works and finding the secret key . A wide variety of cryptanalytic attacks are available which can be classified into various groups, mainly based on what the attacker knows and what capabilities are available to him . Examples are chosen ciphertext attacks, chosen plaintext attacks, ciphertext only attack , known plaintext attack , etc . Cryptanalysis not only refers to penetrating the security of normal encryption , but also many other types of cryptographic algorithms and protocols . Though it mainly uses weaknesses in the algorithm themselves , other types of attacks which do not target the weaknesses in algorithm are often more effective than them , and thus are important too [13,14] .

2.3 Security Services

The International Telecommunication Union – Telecommunication Standardization Sector (ITU-T) provides some security services as well as some mechanisms to implement those services. The security services include the followings.

- Data Confidentiality
- Data Integrity
- Authentication
- Non repudiation
- Access Control

2.3.1 Data Confidentiality

International Organization for Standardization (ISO) defines Confidentiality in ISO-17799 as ensuring that information is accessible only to those authorized to have access and preventing its disclosure to unauthorized agents. Confidentiality is one of the design goals for many cryptosystems & it is one of the main principles of information security. This service may encompass confidentiality of the whole message or a part of it as well as protection against disclosure attack, traffic analysis & snooping [13,14] .

2.3.2 Data Integrity

It is defined as a service used for protection of data from unauthorized modification, insertion, deletion or replaying by an adversary, and it may encompass either the whole message or a part of it.

2.3.3 Authentication

This service is used to validate that both parties are who they claim to be. A peer entity authentication takes place in a connection oriented communication where it provides the authentication of the sender or receiver during connection establishment. A data origin authentication takes place in a connectionless communication where only the source of data is authenticated.

2.3.4 Non-repudiation

This service protects against repudiation by either the sender or the receiver of the data. In non-repudiation with proof of the origin, the receiver of the data can later prove the identity of the sender if denied, while with proof of delivery, the sender of data can later prove that data were delivered to the intended recipient [13,14]. Mechanisms to implement non-repudiation mainly include digital signatures, public key encryptions, etc.

2.3.5 Access Control

Access to protected information must be restricted to people who are authorized to access the information. Access control provides protection against unauthorized access to data . The term *access* in this definition is very broad and can involve reading, writing, modifying, executing programs, etc. [13,14] .

2.4 Digital Signature

Just like physical signatures validate physical printed documents, Digital signatures are digitally or electronically generated security marks which validate digital documents. A digital signature can be defined as a mathematical scheme used to verify the authenticity of a message or document. If a receiver can verify the digital signature of a sender successfully, then he can trust or believe that the message has come from a trusted or known sender. A digital signature is similar to a handwritten signature in many aspects , but there are some distinctions between them like , a digital signature is much more highly secure than a handwritten signature . While a handwritten signature can be forged, it is very difficult and infeasible (in case of strong digital signature schemes) to forge a digital signature. Also, while a handwritten signature is part of the document containing the message, digital signatures are sent as a separate document different from that of the message. Moreover, while there is a one-to-many relationship between a conventional signature and documents, a digital signature has a one-to-one relationship with the message.

Digital signatures use public key encryption systems in order to authenticate & verify digital documents & messages like e-mails etc. A typical digital signature scheme consists mainly of three algorithms:

- A key generation algorithm: - used to generate the public & private keys .
- A signing algorithm: - used to produce the signature using the message & the private keys.
- A verification algorithm: - verifies the signature using the message , public keys & the signature . Upon successful verification, accepts the message, else rejects it.

Here the type of cryptography used is asymmetric since, there are two types of keys involved, one is the private key to sign the message, & the other is the public key used to verify the signature.

Though there are many algorithms & schemes available & can be used , but usually a digital signature is implemented using cryptographic hash functions . Message digests of fixed size are created like 128 bits, 256 bits, 512 bits, etc. Which are then sent along with the message to

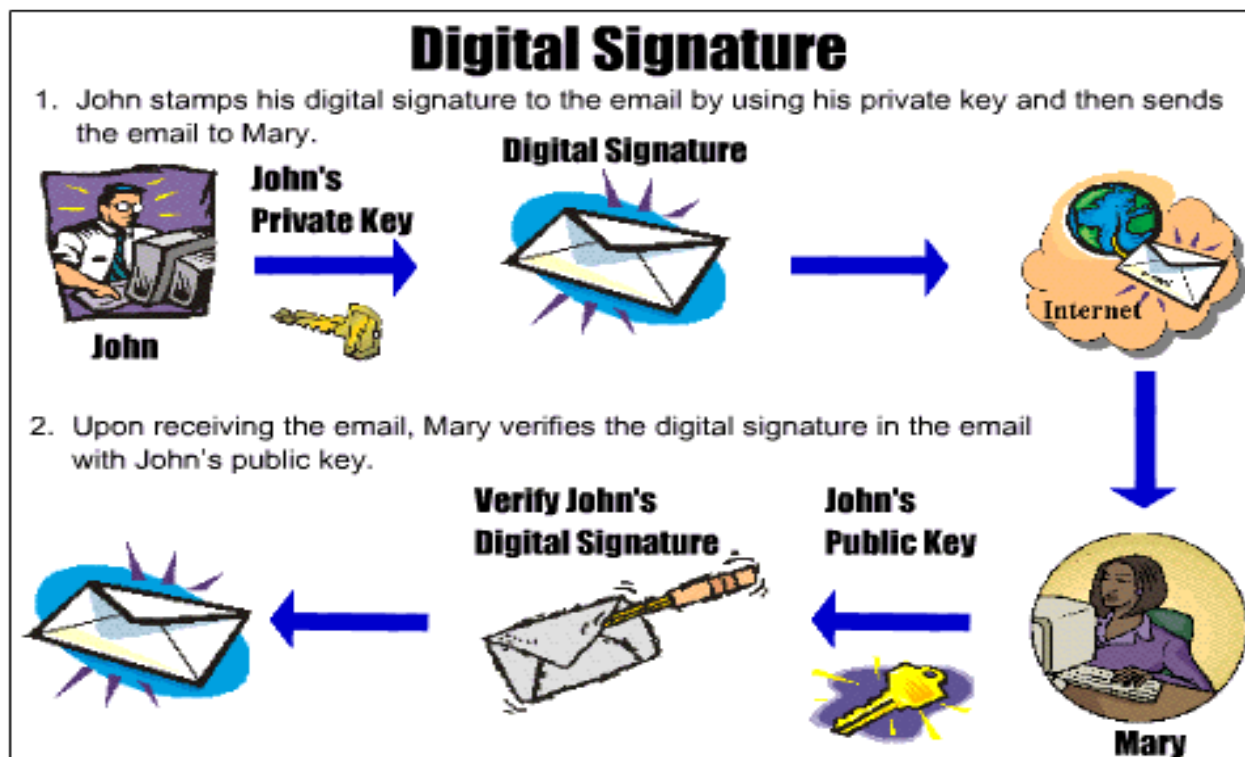


Figure 1 : Digital Signature Scheme

the receiver after encrypting the digest with the sender's private key . The receiver then calculates the hash of the message received & compares with the signature digest after decrypting it with the public key. If both are same, then the message is authenticated and unmodified , and if they are different , then either the sender is different or the message has been modified during transmission .

Digital signatures have found many applications in industries, various organizations etc. mainly due to the three services they provide that are authentication, non-repudiation & integrity. The various applications where they can be used include electronic voting, electronic cash schemes, software distributions, e-mails, etc. where detecting forgery or tampering is of prime importance.

2.5 Blind Signature

2.5.1 Definition

David Chaum [2] was the first to introduce or propose the concept of blind signatures. In normal digital signatures, where only two entities are involved & the sender signs the message , we assume that the sender & receiver know each other and the receiver trusts the sender . But in many cases this may not be true, that is both the entities may not know or trust each other . In this case, the role of a trusted third party comes into the picture , whom both the sender & receiver know and trust . Here, the trusted third party acts as the signer & generates his private & public keys. Upon receiving a message from the sender, he signs it using his private keys & then either sends it to the receiver himself, or sends it back to the sender who then sends the signed message to the receiver. The receiver then verifies the signature using the public keys of the signer. But, in this case the signer can see or know the contents of the message. In most of the applications, the sender would like his message to remain confidential or secret, and even the trusted third party or signing authority should not know the contents of his message. Here, the use of blind signatures comes into effect. Blind signatures are a special type of digital signatures where the message is blinded or disguised before being sent for signing . The message is blinded using various techniques & the private keys of the sender. The signer cannot see the contents of the message , he signs on the blinded message with his private keys , and sends it back to the

sender . The sender then obtains an unblinded version of the signature using his keys , and sends the signature – message pair to the receiver . The receiver verifies the signature using the public keys of signer as well as the message – signature pair itself.

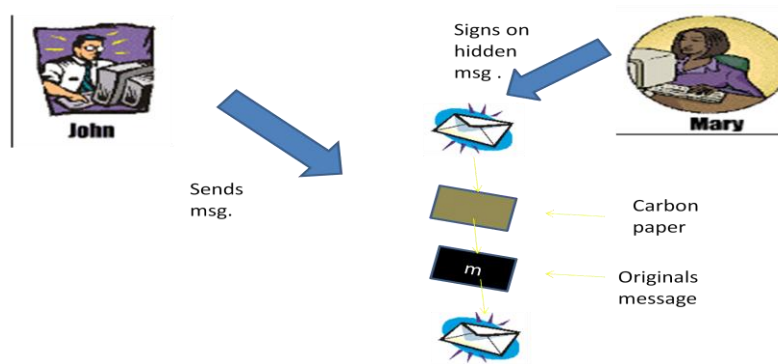


Figure 2: Concept of Blind Signature

2.5.2 Properties

Apart from providing the services like authentication, integrity, non-repudiation, *message confidentiality* etc., some additional properties are there, which if incorporated in the blind signature scheme will make them stronger as well as more useful for various applications.

- *Unlinkability or Untraceability* :- By this property , the signer cannot trace the sender of the message after the message-signature pair has been sent to the receiver as well as cannot determine whether an unblinded version of message was signed by him or not , if called upon to verify the same .
- *Anonymity* :- Due to the above property of untraceability , the identity of the sender can remain secret after the message has been received . This is important for applications where anonymity is very essential , like in electronic voting applications .
- *Unforgability* :- This property can be achieved by using some techniques or methods such as hashing , discrete logarithm problems etc. in the blind signature scheme , so that any attacker is not able to forge a signature & claim to be the sender .

2.5.3 Mechanism

A blind signature scheme disguises the message before sending it to the signing authority in order to preserve the secrecy of the message.

Participants : Basically , three entities participate in a blind signature scheme , the sender , who blinds the message before it gets signed & unblinds the signature received , the signer , who signs the blinded message , and the receiver , who verifies the signature & accepts or rejects the message accordingly .

A blind signature scheme typically consists of the following five phases:-

- *Key Generation* :- In this phase , the signer produces his private & public keys set . The private keys would be used for the signing of the message , while the public keys would be used for verification of the signature . The public keys are made available in the channel , so that the sender & receiver can get them .
- *Blinding* :- In this phase , the sender creates his own private & public keys as required by him , in order to blind or disguise the original message , using some algorithm or technique . He uses his own private keys as well as the public keys of the signer to blind the message . He then sends the blinded message to the signer.
- *Signing* :- After receiving the blinded message from the sender , the signer signs on it using his private keys & some signing algorithm . Then he sends the signature to the sender .
- *Unblinding* :- After receiving the signature on blinded message from the signer , the **sender** tries to obtain an unblinded version of the signature , i.e. the signature as it would have been if signed on original unblinded message . He does this using his keys & an unblinding algorithm . Finally , he sends the message - signature pair to the receiver .
- *Verification* :- After receiving the message – signature pair , the receiver uses some verification equation or algorithm to verify the signature using the signer's public keys as well as the message - signature pair itself . Then he accepts or rejects the message accordingly.

2.5.4 Applications

Blind signatures can be used in many applications , where authentication , integrity , non repudiation , sender privacy , etc. are necessary , like e-mails , e-cash systems , software distributions , documents verification etc. Blind signatures having the additional properties like

untraceability, can be used in applications where anonymity of the sender is required like electronic voting, etc.

2.6 Cryptographic Background

2.6.1 Random Number Generation

A random number generator is a computational algorithm, program or device used to generate a sequence of numbers having no pattern between them, i.e., they appear random .

Due to the large number of applications of randomness, a lot of methods have been developed over the years. Like in ancient times, dice , coins , playing cards , etc. were used for getting randomness . But with the increase in use & demand for generating large number of random numbers in short amount of time, these physical methods were rather inefficient & time consuming . But with the advancement in the technology & computer science , computational random number generators (RNGS) are available now , which are used in a wide variety of applications like government-run lotteries , games , modern slot machines , gambling , statistical sampling , computer simulations , completely randomized designs , cryptography , & in other fields or areas where unpredictability is required in the generation of outputs .

Random number Generation plays an important role in blind signature schemes. All the keys, both private & public, of the sender or signer are generated randomly using RNGs. This also helps in imparting the property of untraceability to the scheme. Each time a signer signs a message , due to randomness , the signatures generated are different from each other and thus no outsider can reveal the identity of the signer from the signature , neither can he determine whether two signatures are signed by the same signer or not .

2.6.2 Primality Test

A primality test is an algorithm to determine whether a given number is prime or not . It is very important in cryptography & has also got uses in various other fields of mathematics. This test does not give the prime factors of the number, rather only states that whether it is prime or not. Primality testing is comparatively easier (running time is polynomial in size of input) than

factorization which is a computationally difficult problem. Some tests prove that a number is prime , while others like Miller-Rabin test prove that a number is composite .

Primality tests are generally of two types, deterministic which always outputs that a given number is prime or composite and probabilistic which always gives correct results for a prime number but may falsely identify a composite as a prime number , though with very small amount of probability. Examples of deterministic tests include Agrawal , Kayal and Saxena 's algorithm , and that of probabilistic include Fermat 's Test , Square root test , Miller-Rabin Test , etc .

2.6.3 Discrete Logarithm

In mathematics, specifically in abstract algebra and its applications, discrete logarithms are group-theoretic analogues of ordinary logarithms. In particular, an ordinary logarithm $\log_a(b)$ is a solution of the equation $a^x = b$ over the real or complex numbers. Similarly, if g and h are elements of a finite cyclic group G then a solution x of the equation $g^x = h$ is called a discrete logarithm to the base g of h in the group G .

In general, let G be a finite cyclic group with n elements. We assume that the group is written multiplicatively. Let b be a generator of G ; then every element g of G can be written in the form $g = b^k$ for some integer k . Furthermore, any two such integers k_1 and k_2 representing g will be congruent modulo n . We can thus define a function ,

$$\log_b : G \rightarrow \mathbb{Z}_n$$

(where \mathbb{Z}_n denotes the ring of integers modulo n) by assigning to each g the congruence class of k modulo n . This function is a group isomorphism, called the discrete logarithm to base b .

The familiar base change formula for ordinary logarithms remains valid: If c is another generator of G , then we have

$$\log_c(g) = \log_c(b) \cdot \log_b(g) .$$

No efficient classical algorithm for computing general discrete logarithms $\log_b g$ is known. The naive algorithm is to raise b to higher and higher powers k until the desired g is found; this is

sometimes called *trial multiplication*. This algorithm requires running time linear in the size of the group G and thus exponential in the number of digits in the size of the group. There exists an efficient quantum algorithm due to Peter Shor [15].

More sophisticated algorithms exist, usually inspired by similar algorithms for integer factorization. These algorithms run faster than the naive algorithm, but none of them runs in polynomial time (in the number of digits in the size of the group).

- Baby-step giant-step
- Pollard's rho algorithm for logarithms
- Pollard's kangaroo algorithm (aka Pollard's lambda algorithm)
- Pohlig–Hellman algorithm
- Index calculus algorithm
- Number field sieve
- Function field sieve

2.6.4 Cryptographic Hash Functions

A cryptographic hash function is a hash function, that is, an algorithm that takes an arbitrary block of data and returns a fixed-size bit string, the (cryptographic) hash value, such that an (accidental or intentional) change to the data will (with very high probability) change the hash value. The data to be encoded is often called the "message," and the hash value is sometimes called the message digest or simply digests.

The ideal cryptographic hash function has four main or significant properties:

- it is easy to compute the hash value for any given message
- it is infeasible to generate a message that has a given hash
- it is infeasible to modify a message without changing the hash
- it is infeasible to find two different messages with the same hash

Cryptographic hash functions have many information security applications, notably in digital signatures, message authentication codes (MACs), and other forms of authentication. They can also be used as ordinary hash functions, to index data in hash tables, for fingerprinting, to detect

duplicate data or uniquely identify files, and as checksums to detect accidental data corruption. Indeed, in information security contexts, cryptographic hash values are sometimes called (digital) fingerprints, checksums, or just hash values, even though all these terms stand for functions with rather different properties and purposes.

Examples of cryptographic hash functions include MD-2 , MD-4 , MD-5 , SHA-0 , SHA-1 , SHA-256 (where the digest is of size 256 bits) , SHA-512 (where the digest is of size 512 bits) , etc.

Chapter 3

Blind Signature Scheme with Untraceability

3 Blind Signature Scheme with Untraceability

A typical blind signature scheme consists of three participants, the sender (or requester) , the receiver & the signer . In this scheme , the sender performs the tasks of blinding & unblinding , the signer signs the blinded message and the receiver verifies the signature & then accepts or rejects the message . With the added property of untraceability , the sender of the message cannot be traced after the delivery of the message .

3.1 Review of Lee , Hwang & Yang 's scheme [1]

3.1.1 Algorithm

The scheme consists of the three participants, the sender (or requester) , the receiver & the signer and has five phases :-

1. Key Generation (done by Signer) :-

- i. Two large primes p & q are generated , such that $q \mid (p-1)$.
- ii. Then we generate g such that $g \in Z_q^*$.
- iii. Then Signer generates his secret key x , & public key $y = g^x \bmod p$.
- iv. Then he generates k_1' , k_2' , b_1 , $b_2 \in Z_q$.
- v. Then he calculates :-

$$r_1' = g^{k_1'} \bmod p$$

$$\& r_2' = g^{k_2'} \bmod p , \text{ such that } \gcd(r_1', q) = 1.$$

- vi. Then he sends r_1' , r_2' , b_1 & b_2 to requester for blinding .

2. Blinding (done by Requester) :-

- i. The Requester has his message , let's denote it by ' m ' .
- ii. He generates a , b , c , d , e randomly .

iii. He calculates:-

$$r_1 = r_1'^{ab} g^c \mod p$$

$$r_2 = r_2'^{bb_2} g^e \mod p$$

$$r = (r_1 * r_2)^d \mod p$$

iv. Then he generates blinded message as follows :-

$$m_1' = mr_1'(r^{-1}/2) * ad \mod q$$

$$m_2' = mr_2'(r^{-1}/2) * bd \mod q$$

v. Then he sends m_1' & m_2' to signer for signing .

3. Signing (by Signer) :-

i. The signer generates signatures as follows :-

$$s_1' = xr_1' + k_1'b_1m_1' \mod q$$

$$s_2' = xr_2' + k_2'b_2m_2' \mod q$$

ii. Then he sends s_1' & s_2' to requester for unblinding .

4. Unblinding(by Requester) :-

i. The requester unblinds the signature as follows : -

$$s_1 = s_1' r_1^{-1} (r/2) + cdm \mod q$$

$$s_2 = s_2' r_2^{-1} (r/2) + edm \mod q$$

$$s = s_1 + s_2 \mod q$$

ii. The requester then sends the whole signature (m , r , s) to the receiver .

5. Verification (by Receiver) :-

i. The receiver checks if $g^s \equiv (y^r r^m) \mod p$.

ii. If it comes true, then the receiver knows that the message comes from an authenticated or authorised sender.

This scheme not only overcomes the shortcoming of *Carmenisch et al.'s* [7] scheme, but also achieves the properties, blindness and untraceability [1] .

3.2 Proposed Scheme

3.2.1 Algorithm

Our scheme also consists of the three participants , the sender(or requester) , the receiver & the signer and has the following five phases :-

1. Initialization (by Signer) :-

- a) The signer randomly selects large primes p_1 & p_2 such that

$$n = p_1 \cdot p_2 ,$$

$$p = 2n + 1 , \text{ and } p \text{ is prime.}$$

- b) Then he selects private keys x , r & public keys

$$y = g^x \bmod p , \text{ and}$$

$$h = g^r \bmod p , \text{ where } g \text{ is a large prime selected by signer randomly.}$$

- c) Then he sends public key set (p, g, y, n, h) to the requester (who wants his message to be signed).

2. Blinding (by Requester) :-

Suppose a requester wants to obtain a signature on message m . The steps are:

- a) He selects private keys a, b & w randomly and computes

$$(i) \quad u_1 = H(hg^a y^b \bmod p, m) \bmod n , \text{ where } H \text{ is a cryptographic hash function (preferably SHA-512) .}$$

$$(ii) \quad u_2 = u_1 + b \bmod n .$$

$$(iii) \quad k = g^w \bmod p .$$

- b) Then he sends (u_2) , the blinded message, to the signer.

3. Signing (by Signer) :-

After receiving (u_2) from requester, the signer computes

$$z = (r + u_2 x) \bmod n , \text{ and sends } z \text{ to requester .}$$

4. Un-blinding (by Requester) :-

a) After receiving z , the requester computes

$$z' = z + a + w \bmod n .$$

b) Then he sends (z', u_1, k) as the Blind signature on message m , to the receiver .

5. Verification (by Receiver) :-

The signature's authenticity can be verified by checking the following equation:-

$$H(g^{z'}(y)^{-u_1} k^{-1} \bmod p, m) \bmod n = u_1 \bmod n .$$

Correctness :

The *proof of equality* is as follows:-

$$\begin{aligned} LHS &= H(g^{z'}(y)^{-u_1} k^{-1} \bmod p, m) \bmod n \\ &= H(g^{z+a+w}(y)^{-u_1} k^{-1} \bmod p, m) \bmod n \\ &= H(g^a g^w g^r g^{u_2 x}(y)^{-u_1} k^{-1} \bmod p, m) \bmod n \\ &= H(hg^a (kk^{-1}) g^{xu_1} g^{xb}(y)^{-u_1} \bmod p, m) \bmod n \\ &= H(hg^a (y^{u_1} y^{-u_1}) y^b \bmod p, m) \bmod n \\ &= H(hg^a y^b \bmod p, m) \bmod n \\ &= u_1 \bmod n . \\ &= RHS. \end{aligned}$$

Thus the correctness of the scheme is proved.

The following figure (figure- 1) is an overall representation of our scheme : -

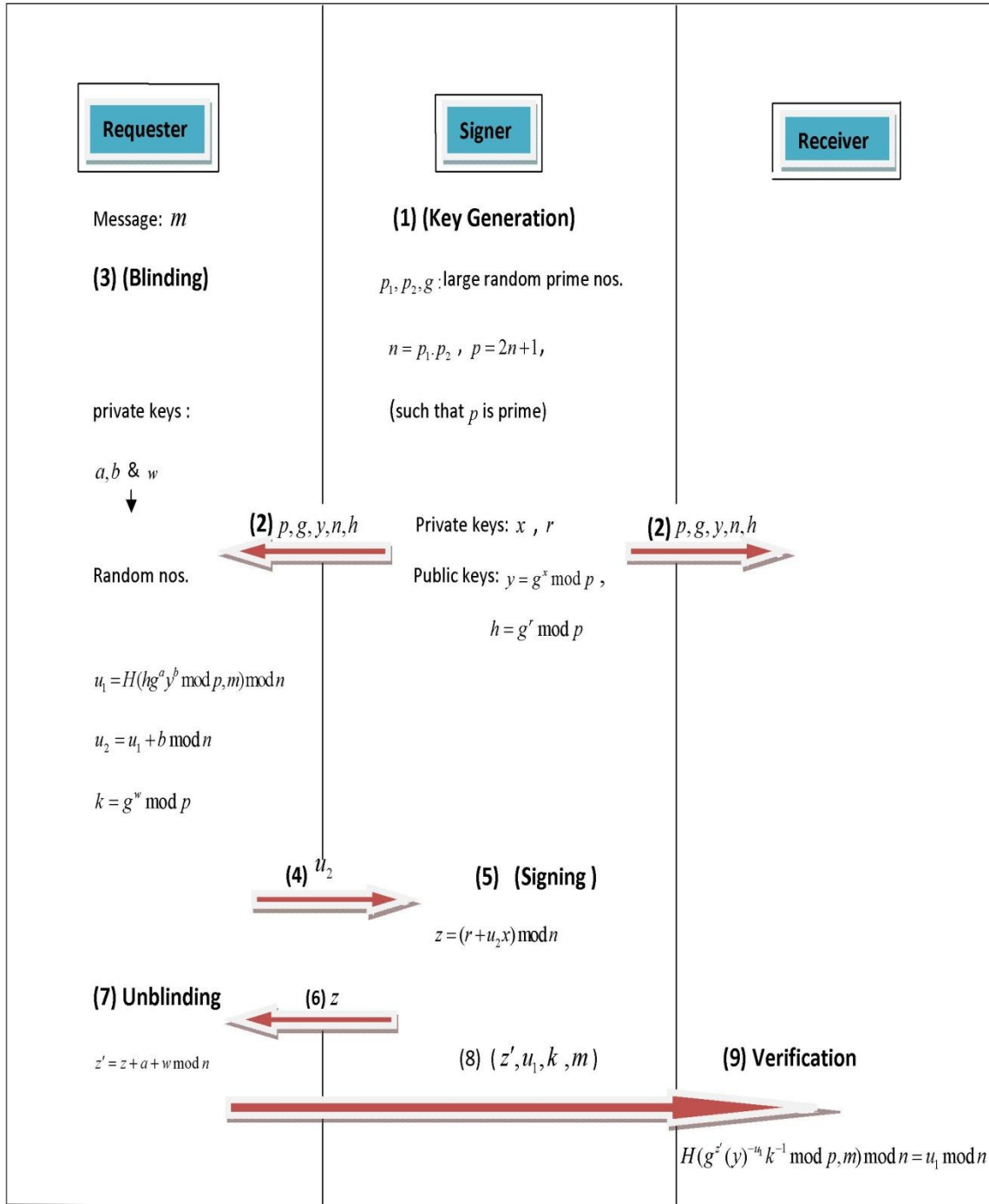


Figure - 3

3.2.2 Security Analysis

The security of our scheme is based both on the strength of the *hash function* and the difficulty of solving the *discrete logarithm problem* [7,10,11,12].

Untraceability :-

In our scheme , it is infeasible for the signer to trace the blind signature , which is demonstrated as follows : For each blinded message that is sent to the signer , he can keep a record of the values : (u_2, z) , and when the requester reveals (z', u_1, k, m) to receiver/public , he (signer) can calculate a value , b' from $b' = u_2 - u_1 \bmod n$.

But from z' and z , he can calculate the value $a' + w' = z' - z \bmod n$. From k , he can't calculate w' due to the difficulty of discrete logarithm problem . So, since w' is unknown, a' can't be calculated, so he can't trace the message by using $u_1 = H(hg^{a'} y^{b'} \bmod p, m) \bmod n$.

Therefore, our algorithm satisfies the property of untraceability.

Forgery attack analysis :-

We are using *SHA-512* as our hash function and its strength is based on the fact that, given a message digest or hash value, it is infeasible to get the message from it.

- Based on discrete logarithm problem, given y and g , it is infeasible to compute x (private key) from $y = g^x \bmod p$.
- For passing the verification equation: $H(g^{z'}(y)^{-u_1} k^{-1} \bmod p, m) \bmod n = u_1 \bmod n$, successfully , an attacker has to randomly choose any two values from z', u_1 and k , and compute the third one .

1. If he chooses z' and u_1 randomly, then it is infeasible to find k , since hash function is non-invertible .

2. If he chooses k and either z' or u_1 , then similarly it is infeasible to find u_1 or z' , respectively, due to the hash function as well as due to the difficulty of solving the discrete logarithm problem.

- Given a valid signature (z', u_1, k, m) , it is infeasible to derive another valid signature (z'', u_1', k', m') such that $H(g^{z''}(y)^{-u_1'}(k')^{-1} \bmod p, m') \bmod n = u_1' \bmod n$, due to the infeasibility of inverting the hash function.

3.2.3 Performance Comparison

Type of Computation	Lee , Hwang & Yang's scheme	Proposed scheme
Multiplication	29M	7M
Exponentiation	11E	7E
Hash	0H	2H
Inverse	3I	1I

Table 1 : Comparative study of Computational Complexities

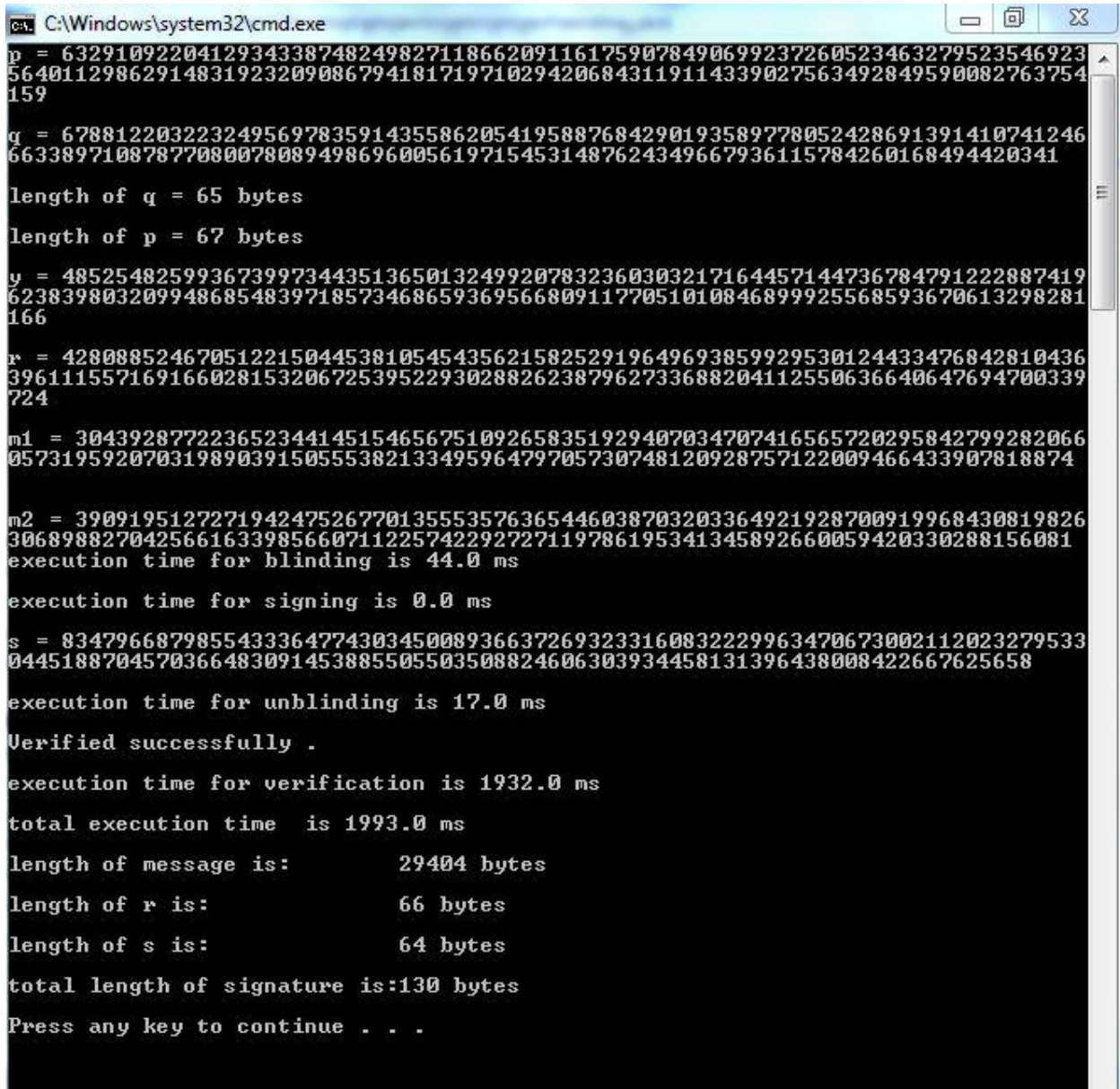
Our proposed scheme uses lesser number of operations than Lee , Hwang & Yang's scheme and thus the total execution time will be lesser than their scheme.

Chapter 4

Implementation Details

4 Implementation Details

4.1 Lee , Hwang & Yang's scheme's Output



```
C:\Windows\system32\cmd.exe
p = 6329109220412934338748249827118662091161759078490699237260523463279523546923
56401129862914831923209086794181719710294206843119114339027563492849590082763754
159
q = 6788122032232495697835914355862054195887684290193589778052428691391410741246
663389710878770800780894986960056197154531487624349667936115784260168494420341
length of q = 65 bytes
length of p = 67 bytes
y = 4852548259936739973443513650132499207832360303217164457144736784791222887419
62383980320994868548397185734686593695668091177051010846899925568593670613298281
166
r = 4280885246705122150445381054543562158252919649693859929530124433476842810436
39611155716916602815320672539522930288262387962733688204112550636640647694700339
724
m1 = 304392877223652344145154656751092658351929407034707416565720295842799282066
0573195920703198903915055538213349596479705730748120928757122009466433907818874
m2 = 390919512727194247526770135553576365446038703203364921928700919968430819826
3068988270425661633985660711225742292727119786195341345892660059420330288156081
execution time for blinding is 44.0 ms
execution time for signing is 0.0 ms
s = 8347966879855433364774303450089366372693233160832229963470673002112023279533
0445188704570366483091453885505503508824606303934458131396438008422667625658
execution time for unblinding is 17.0 ms
Verified successfully .
execution time for verification is 1932.0 ms
total execution time is 1993.0 ms
length of message is:      29404 bytes
length of r is:            66 bytes
length of s is:            64 bytes
total length of signature is:130 bytes
Press any key to continue . . .
```

Figure 4

4.2 Our Proposed scheme's output

4.2.1 Key Generation

```
C:\Windows\system32\cmd.exe
TCP-Signer Waiting for Requester on port 1255
THE REQUESTER /127.0.0.1:54643 IS CONNECTED

Key Generation phase
-----
Public keys :

y1 = 783020173189297314287219153693231753064902469917501834331381856111013924191
3613223488572632571221902413035225955788170347172394756096735759408651831177447
h = 1153338961357370999693920678253109932126693097677440882512864793958530669590
5580068903147290576494391373236895413708006517792098241855595466151981572041599
g = 6807560868853329232669873291836902046489630984246888489554334140601129078936
357697774715014179681449631941178555779705551444420676853669406359078447423224
p = 2477697136969775807972384163684944867873863389360424757240345661123841516646
0704301874786501813157328164781527963480062201129449808849417491971875024184499
n = 1238848568484887903986192081842472433936931694680212378620172830561920758323
0352150937393250906578664082390763981740031100564724904424708745985937512092249

Sending public keys to SignerPublicData File
Public Keys Sent
-----
```

Figure 5 : Key Generation Phase

4.2.2 Blinding

```
C:\Windows\system32\cmd.exe
Receiving data from SignerPublicData File
Received following Public data of Signer :

y1 = 783020173189297314287219153693231753064902469917501834331381856111013924191
3613223488572632571221902413035225955788170347172394756096735759408651831177447
h = 1153338961357370999693920678253109932126693097677440882512864793958530669590
5580068903147290576494391373236895413708006517792098241855595466151981572041599
g = 6807560868853329232669873291836902046489630984246888489554334140601129078936
357697774715014179681449631941178555779705551444420676853669406359078447423224
p = 2477697136969775807972384163684944867873863389360424757240345661123841516646
0704301874786501813157328164781527963480062201129449808849417491971875024184499
n = 1238848568484887903986192081842472433936931694680212378620172830561920758323
0352150937393250906578664082390763981740031100564724904424708745985937512092249

Blinding Phase
execution time for blinding is 111.0 ms
Blinded message u2 = 81354889877539763928716658835965651912921679932961906851887
34031448028632872411681164150301138823290658558857868504651086908431800451545260
360531061161562

Sending u2 to Signer ....
u2 sent .
-----
```

Figure 6 : Blinding Phase

4.2.3 Signing

```
Receiving Blinded message from Requester :
RECEIVED: u2 = 81354889877539763928716658835965651912921679932961906851887340314
48028632872411681164150301138823290658558857868504651086908431800451545260360531
061161562
-----
Signing phase
Signing phase completed
execution time for signing is 2.0 ms
Signature z = 725732807472988660919375898617473503780503108069468773871176412570
37480004751346373388806719809406857979085870543978881945331625937078645294491566
17558014
-----
Sending signature z to Requester
Signature Sent
Signer's Work Over .
```

Figure 7 : Signing Phase

4.2.4 Unblinding

```
RECEIVED:signature z from Signer = 725732807472988660919375898617473503780503108
06946877387117641257037480004751346373388806719809406857979085870543978881945331
62593707864529449156617558014
-----
Unblinding Phase
execution time for unblinding is 1.0 ms
Unblinded Signature z1 = 5987978688805824952177231332291866236864906827823097320
54227908744623656100033555181942557658358611153585074585027830707288284226851500
408487835706568786
-----
Total Signature :
z1 = 598797868880582495217723133229186623686490682782309732054227908744623656100
033555181942557658358611153585074585027830707288284226851500408487835706568786
u1 = 335862064529799822009529463519075437435957291036167130164831318892517089987
5206019248323269908222313354517146285733475294638516406275200617776720742864828
k = 1944717532107693732199497088626989940391968533660202703593012890702639725525
4311511502586941814761904386771243428993419373017592712646516087669885172368807
length of u1 is:          64 bytes
length of z1 is:          64 bytes
length of k is:           65 bytes
total length of signature is:193 bytes
length of message is:      29404 bytes
-----
Sending Signature & message to senderdata file .....
Message & Signature Data sent to Receiver for Verification
-----
Requester's job completed .
```

Figure 8 : Unblinding Phase

4.2.5 Verification

```

C:\Windows\system32\cmd.exe
Receiving Public data of Signer .....
Received following Public data of Signer :
v1 = 783020173189297314287219153693231753064902469917501834331381856111013924191
3613223488572632571221902413035225955788170347172394756096735759408651831177447
h = 1153338961357370999693920678253109932126693097677440882512864793958530669590
5580068903147290576494391373236895413708006517792098241855595466151981572041599
g = 680756086885329232669873291836902046489630984246888489554334140601129078936
357697774715014179681449631941178555779705551444420676853669406359078447423224
p = 2477697136969775807972384163684944867873863389360424757240345661123841516646
0704301874786501813157328164781527963480062201129449808849417491971875024184499
n = 1238848568484887903986192081842472433936931694680212378620172830561920758323
0352150937393250906578664082390763981740031100564724904424708745985937512092249
-----
Receiving Signature & message for Verification .....
Received following Signature :
z1 = 598797868880582495217723133229186623686490682782309732054227908744623656100
033555181942557658358611153585074585027830707288284226851500408487835706568786
u1 = 335862064529799822009529463519075437435957291036167130164831318892517089987
5206019248323269908222313354517146285733475294638516406275200617776720742864828
k = 1944717532107693732199497088626989940391968533660202703593012890702639725525
4311511502586941814761904386771243428993419373017592712646516087669885172368807
-----
Verification starts
lhs = 33586206452979982200952946351907543743595729103616713016483131889251708998
75206019248323269908222313354517146285733475294638516406275200617776720742864828
rhs = 33586206452979982200952946351907543743595729103616713016483131889251708998
75206019248323269908222313354517146285733475294638516406275200617776720742864828
Verified successfully .
execution time for verification is 42.0 ms
Total execution time is 156.0 ms
-----
Message & signature Verified .

```

Figure 9 : Verification Phase

4.3 Comparison with Lee , Hwang & Yang's scheme

Phase	Lee , Hwang & Yang's scheme	Proposed scheme
Blinding	44.0	111.0
Signing	0.0	2.0
Unblinding	17.0	1.0
Verification	1932.0	42.0
Total	1993.0	156.0
Signature Length (in bytes)	130	193

Table 2 : Execution time (in milliseconds) comparison for a message size of 29404 bytes .

We have implemented Lee, Hwang and Yang's scheme [1] as well as our proposed scheme and found out the **execution times** for various phases along with the signature length for different message sizes. Here, *existing* refers to Lee, Hwang and Yang's scheme [1] and proposed refers to our scheme.

Chapter 5

Conclusion

5 Conclusion

We have proposed a new blind signature scheme based on discrete logarithm problem for untraceability. The security of our scheme is based both on the difficulty of solving the discrete logarithm problem as well as the strength and security of hash functions. Our scheme not only provides *untraceability* but also *unforgability*. It can be used in various applications like e-voting and e-cash systems etc.

References

- [1] Cheng-Chi Lee, Min-Shiang Hwang ,Wei-Pang Yang , A new blind signature based on the discrete logarithm problem for untraceability , Applied Mathematics and Computation 164 (2005) 837–841 .
- [2] D. Chaum, Blind signatures for untraceable payments, in: Advances in Cryptology, CRYPTO_82, 1982, pp. 199–203.
- [3] C.-C. Chang, M.-S. Hwang, Parallel computation of the generating keys for RSA cryptosystems, IEE Electronics Letters 32 (15) (1996) 1365–1366.
- [4] R.L. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public key cryptosystems, Communications of the ACM 21 (February) (1978) 120–126.
- [5] M.-S. Hwang, C.-C. Lee, Y.-C. Lai, Traceability on low-computation partially blind Signatures for electronic cash, IEICE Transactions on Fundamentals on Electronics, Communications and Computer Sciences E85-A (5) (2002) 1181–1182.
- [6] Y.-L. Tang, M.-S. Hwang, Y.-C. Lai, Cryptanalysis of a blind signature scheme based on elgamal signature, International Journal of Pure and Applied Mathematics, in press.
- [7] J. Carmenisch, J. Piveteau, M. Stadler, Blind signatures based on discrete logarithm problem, in: Advances in Cryptology, EUROCRYPT_94, Lecture Notes in Computer Science, vol. 950, 1994, pp. 428–432.
- [8] L. Harn, Cryptanalysis of the blind signatures based on the discrete logarithm problem, IEE Electronic Letters (1995) 1136–1137.
- [9] P. Horster, M. Michels, H. Petersen, Comment: Cryptanalysis of the blind signatures based on the discrete logarithm problem, IEE Electronic Letters 31 (21) (1995) 1827.
- [10] T. ElGamal, A public-key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactions on Information Theory IT-31 (July) (1985) 469–472.

- [11] M.-S. Hwang, C.-C. Chang, K.-F. Hwang, An ElGamal-like cryptosystem for enciphering Large messages, *IEEE Transactions on Knowledge and Data Engineering* 14 (2) (2002) 445–446.
- [12] M.-S. Hwang, C.-C. Lee, E.J.-L. Lu, Cryptanalysis of the batch verifying multiple DSA-type digital signatures, *Pakistan Journal of Applied Sciences* 1 (3) (2001) 287–288.
- [13] B Forouzan . *Cryptography and Network Security* . TMH .
- [14] W Stallings . *Cryptography and Network Security* . Prentice Hall .
- [15] Shor, Peter (1997). "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". *SIAM Journal on Computing* **26** (5): 1484–1509.