DEVELOPMENT OF TIME-STAMPED SIGNCRYPTION SCHEME AND ITS APPLICATION IN E-CASH SYSTEM

Thesis submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Sanjib Kumar Baral
(Roll: 109CS0435)

Sourav Dash
(Roll: 109CS0070)

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India

DEVELOPMENT OF TIME-STAMPED SIGNCRYPTION SCHEME AND ITS APPLICATION IN E-CASH SYSTEM

Thesis submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Sanjib Kumar Baral
(Roll: 109CS0435)

Sourav Dash
(Roll: 109CS0070)

under the guidance of

Prof. Sujata Mohanty
NIT Rourkela

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India
May 2013

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India.

May 10, 2013

Certificate

This is to certify that the thesis entitled ***Development of Time-stamped Signcryption Scheme and its Application in E-cash System*** by ***Sanjib Kumar Baral and Sourav Dash*** in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Computer Science and 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.

Prof. Sujata Mohanty
Dept. of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769008

Acknowledgment

We are indebted to our guide Prof. Sujata Mohanty for giving us an opportunity to work under her guidance. Like a true mentor, she motivated and inspired us through the entire duration of our work, without which this project could not have seen the light of the day. We also convey our heart-felt sincerity to Prof. B. Majhi for his support and timely suggestions.

We convey our regards to all the other faculty members of Department of Computer Science and Engineering, NIT Rourkela for their valuable guidance and advices at appropriate times. We would like to thank our friends for their help and assistance all through this project.

Last but not the least, we express our profound gratitude to the Almighty and our parents for their blessings and support without which this task could have never been accomplished.

Sanjib Kumar Baral
109CS0435
B.Tech
Department of CSE, NIT Rourkela

Sourav Dash
109CS0070
B.tech
Department of CSE, NIT Rourkela

Abstract

A signcryption scheme combining public key encryptions and digital signatures in one logical step can simultaneously satisfy the security requirements of confidentiality, integrity, authenticity and non-repudiation and with a cost significantly lower than that required by the traditional "signature followed by encryption" approach. This thesis presents a new generic concept of time-stamped signcryption scheme with designated verifiability. Here an authenticated time-stamp is associated with the signcrypted text which can only be verifiable by a specific person, known as the designated verifier. The time-stamp is provided by a trusted third party, namely, Time Stamping System (TSS). The scheme is proved to be secure, as, no one, not even the signcrypter or TSS can produce a valid signcrypted text on behalf of them. We analyzed the security of the proposed scheme and found that it can withstand some active attacks. This scheme is resistant against both inside and outside attacks. The security of our scheme is based upon the hardness of solving Computational Diffie Hellman Problem (CDH), Discrete Logarithm Problem (DLP) and Integer Factorization Problem (IFP). The proposed scheme is suitable in scenarios such as, on-line patent submission, on-line lottery, e-cash, e-bidding and other e-commerce applications. Also we propose an e-cash system based on our proposed time-stamped signcryption scheme which confirms the notion of e-cash securities like anonymity of the spender, unforgeablity of the digital coin, prevention of double spending.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the ever rising volume of information in the present computational scenario the need for security has never been so high. Apart from the volume, the worth of the information has been the highest. Like any other asset of an organization, company or even individual, information is the most valuable of all. The decrease in remoteness of the information and the ever increasing Web Applications adds to the need to prevent it from falling in the wrong hands, i.e. increasing the confidentiality of the information. In present day most of the communication takes place on the insecure channel which makes the message vulnerable to multiple threats. On the other hand creating a secure channel is quite expensive and not scalable. While sending a message over an insecure channel such as internet to a person we must primarily provide four different security requirements such as,

- Confidentiality - Protection from disclosure to unauthorized persons
- Integrity - Maintaining data consistency
- Authentication - Assurance of identity of person or originator of data
- Non-repudiation - Originator of communications can't deny it later

These are the primary security goals a message passing system must satisfy for a successful communication. In the past, cryptography mainly concerned with the confidentiality factor. Then, the most application of cryptography were in the department of defense or other organization to collect and report secret information on an enemy or competitor. In the last decade the other security goals like integrity verification, user authentication, digital signatures etc. have been added to the confidentiality factor. Even in ancient times cryptography was used on the basis of some simple ciphers like the Caesars cipher and passing the keys through a secured courier system.

1.1 Introduction to Cryptography

Cryptography is the science of hiding information and preventing it from being changed and accessed by the unauthenticated users/user programs. It includes applying various modern public key cryptographic schemes to create an encrypted text which would be completely unintelligible to the adversary (Eve) and can only be accessed by a person or group of persons authenticated to do so [1, 2]. The complete process can be named as Encryption. While Symmetric Key Cryptography (common key) was used in the past they are kind of obsolete now due to various advancements in technological field and introduction of various algorithms to check for prime numbers. Now a days Asymmetric key Cryptography is preferred. So we can classify the cryptography into two parts

- Symmetric Key Cryptography (common key)
- Asymmetric Key Cryptography (different key)

1.1.1 Symmetric Key Cryptography

In symmetric key cryptography the sender encrypts the message and sends it by a key say k. The receiver decrypts the message after receiving the message by using the same key k. The assumption is based upon the fact that here, both the sender and receiver use a common key and the transmission of the message and the key of cipher text is done in an insecure channel. This system is vulnerable and flawed if the key k is leaked and it is known to the adversary.

1.1.2 Asymmetric Key Cryptography

To overcome the problems of the symmetric key cryptography or the common key cryptography public key cryptosystem or public key encipherment is used, we have the same situation as of symmetric key cryptosystem, with a few exception. First, there are two keys instead of one, one public key and one private key. To send a secured message, the sender encrypts with receivers public key. To decrypt the message the receiver uses his own private key.

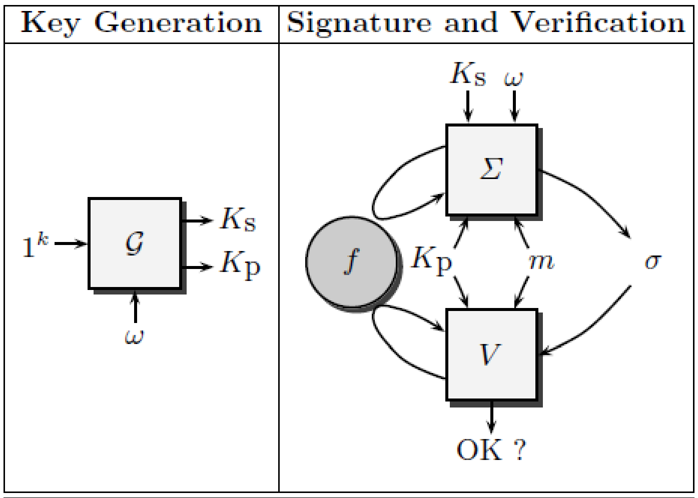
Diffie and Hellman first outlined the Public Key Cryptography methodology, which could be used for key agreement and had application to other cryptographic problems [3]. Diffie and Hellman did not provide concrete constructions for how this concept of public-key cryptography could be implemented in practice. It was not until the fundamental work of Rivest, Shamir, and Adleman that the first public-key cryptosystem was realized [4]. The concept of Public Key Cryptography gave birth to the new remarkable concept of Digital Signature. Electronically digital signature is the analogous of the traditional handwritten signature. The purpose of the digital signature is to enable a person to digitally sign some type of electronic document. One would like for these digital signatures to have the same properties as traditional signatures: they should be easy to produce, easy to check and yet difficult to forge. By using the private key to sign, and the public key to verify, this notion was achieved. As time passed, several other realizations of public-key cryptosystems and digital signatures were proposed.

1.2 Digital Signature

For a document the most important security goal is it's Authenticity. In the physical world conventionally the signature is included in the document as a part of it, which is not in case of digital signature as the signer or the sender sends the message and the signature as two separate documents to the receiver which receives both documents and starts the verification process of checking whether that the signature actually belongs to the sender. If verified then the document is accepted else rejected. Digital signature was first proposed by Diffie and Hellman [3]. In this section, we give a more precise definition of signature scheme which is based on [5]. The figure 1.1 is from [6].

Definition 1.2.1 (Signature Scheme): A signature scheme is composed of the following three polynomial time algorithms:

- The key generation algorithm (G). On input 1^k, where k is the security parameter, the algorithm G produces a pair (K_p, K_s) of matching public and secret keys. Algorithm G is probabilistic.
- The signing algorithm (Sign). Given a message m and a pair of matching public and secret keys (K_p, K_s), Sign produces a signature §. The signing algorithm might be probabilistic, and in some schemes may also receive other inputs.



Sizes of objects	
k is the security parameter Outputs of f are of size k K_P is of length n	$\Big\}$ such that $k \gg \log n$

FIGURE 1.1: Signature Scheme

- The verification algorithm (Ver). Given a signature § on a message m and a public key K_p, Ver tests if § is a valid signature of m with respect to K_p. In general, the verification algorithm need not be probabilistic

1.3 Signature-Then-Encryption

The digital signature paradigm only provides the Authenticity part of the security goal. To enhance the security to include Confidentiality along with the Authenticity we make use of both signature and encryption. It can be done in two separate simple steps of signing the document using some signing schemes and the encrypting it based on some predefined encryption schemes. The various steps can be written as the follows;

- Signing is done using a Public key DS (Digital Scheme) scheme
- Encrypting the message along with the signature with the use of a existing private key encryption algorithm under randomly chosen message encryption key
- Encrypt the random message encryption key using the receivers public key
- Send the message

1.3.1 Disadvantage of Signature-Then-Encryption

The Signature-Then-Encryption process is associated with several problems and is quite trivial combination of the two completely independent processes which can be easily done in separate steps and has no meaning in combining it into one. Various disadvantages are as follows;

- Unnecessarily consumptiom of machine cycles through out the entire ongoing process.
- Introduction of extended bits to original messages which in turn increases the communicational cost.
- Adds to the computational cost which requires a comparable amount of time for signature verification and decryption cost of delivering a message is essentially the sum of the cost for digital signature and that for encryption!

1.4 Signcryption

The word signcryption was first coined by Yuliang Zheng in the year 1997 at Monash University, Australia. According to him signcryption is a cryptographic primitive which combines both the functions of digital signature and public key encryption logically in a single step, and with a computational cost significantly less than that needed by the traditional signature-then-encryption approach [7].After Zhengs proposal, much research has been carried out in the area of signcryption and its variants [8–12].

It is a new paradigm in public key cryptography that solves all the problems associated with the previous Signature-Then-Encryption method. It takes logically a single step to fulfill the functions of both digital signature and public key encryption simultaneously [7]. The cost, both computational and communicational cost is lower than that required in the previous Signature-Then-Encryption. The first attempt to provide formal security analysis of signcryption schemes was made by Steinfeld and Zheng [13], who proposed a signcryption scheme based on the integer factorization problem and provided a formal security model and proof for the unforgeability of the proposed scheme. Signcryption can be implemented using various schemes like ElGamals Shortened Digital Signature, Schnorrs scheme and any other digital signature schemes in conjunction with a public key encryption schemes like DES and 3DES.The choice is made by the user based on the level of security desired.

1.4.1 Signcryption Features

Basically a signcryption process in consistent of 3 phases implementing different algorithms: Key Generation, Signcryption, Unsigncryption. The generation phase creates all the pairs of public and private keys for Alice and Bob. The signcryption phase creates the signcryption parameter (c,r,s) and sends it to Bob and in the verification or the designcryption phase the bob verifies the signature using various designcryption algorithm .

Various features of Digital Signcryption are as follows;

- Unique Unsigncryptability- A message m is signcrypted using a signcryption algorithm to give a signcrypted output c. The receiver can apply unsigncryption algorithm on c to verify the message m. This unsigncryption is unique to the message m and the sender.
- Security - Ensures that the message sent cant be forged by a untrusted party .It also ensures the contents of the message are confidential and ensures nonrepudiation
- Efficiency - Computational involved when applying the Signcryption, Unsigncryption algorithms and communicational overhead is much smaller than signature-then-encryption schemes.

1.5 Time-stamp

Time-stamping is a technique for providing proof of existence of certain digital document or data prior to a specific time [14]. Time-stamping is now widely recognized as an important mechanism used to ensure the integrity of digital data. Time-stamping is highly required in many domains like patent submissions, electronic votes or electronic commerce. Time-stamping is usually enforced to ensure non-repudiation. A digital signature can only be legally binding if it was made when the user's certificate was still valid, and a time-stamp on a signature can successfully prove this. Time-stamps are provided by trusted third parties, known as Time Stamping Systems [15, 16]. Without a time-stamp, signed documents can

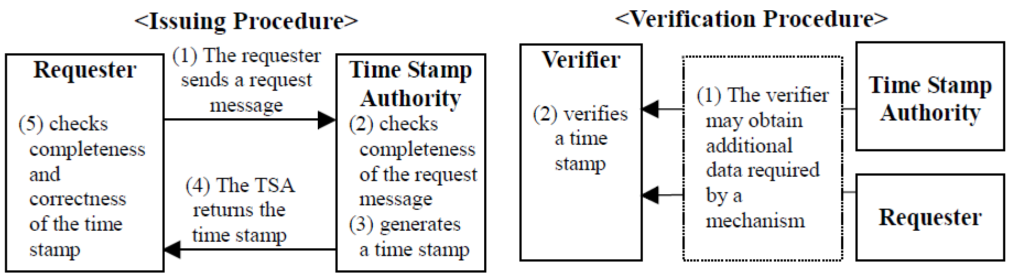

<Issuing Procedure>
Requester
(5) checks completeness and correctness of the time stamp
(1) The requester sends a request message
(4) The TSA returns the time stamp
Time Stamp Authority
(2) checks completeness of the request message
(3) generates a time stamp
<Verification Procedure>
Verifier
(2) verifies a time stamp
(1) The verifier may obtain additional data required by a mechanism
Time Stamp Authority
Requester

Figure 1.2: Issuing and Verification Procedures of a Time Stamp in ISO/IEC WD 18014

neither be trusted when the signers signature key was lost, stolen, or accidentally compromised, nor solve the cases when the signer himself repudiates the signing, claiming that he has accidentally lost his signature key. Even if a time-stamp is included in the document, no one can believe that the time-stamp is true because the signer may forge it. To overcome this problem, a trusted party, known as Time-Stamping Service (TSS), is needed to authenticate the time when the signature is generated [16].

Digital time-stamping systems were first introduced by Haber and Stornetta [17]. After this, a lot of research has been done in the area of time-stamped signatures [15–17]. In 2004, Sun et. al proposed a time-stamped signature scheme based on absolute temporal authentication which was secure against the forward forgery [16]. Z. Shao proposed a more efficient scheme capable of withstand both forward and backward forgery with low computational cost [15].

1.5.1 Time-stamping Procedure

ISO/IEC WD 18014 provides a generic model on which time-stamping services are based. The framework developed in ISO/IEC JTC1/SC27 gives an outline of the procedure for time-stamping procedure [18]. The issuing procedure consists of mainly five steps which is shown in the figure 1.2 from [18].

Step 1: The requester sends a request message for time-stamp to TSA (Time-stamping Authority)

Step 2: After getting a request message TSA checks the completeness of the request message

Step 3: After completeness check, TSA generates the time-stamp which includes at least a time parameter, a hash value of the data to be time-stamped and data to bind the time parameter to the hash value using a cryptographic technique

Step 4: TSA sends the time-stamp to the requester

Step 5: The requester checks the correctness and completeness of the time-stamp

In the verification procedure a verifier obtains other data required from other entities and verifies the time-stamp.

1.5.2 Classification of Time-stamp

Conventionally time stamping schemes have been generally classified into three categories: simple, linking and distributed schemes [19].

Simple Scheme: As the name states the schemes of this category are very simple, but its security depends on time-stamp issuer's reliability. Haber and Stornetta have pointed out that if an issuer fraudulently alters the time parameter of a certain time-stamp, nobody can detect the alteration [20]. As countermeasures against the problem, the linking and distributed schemes have been developed.

Linking Scheme: The time-stamp issuer creates a time-stamp which includes data which are included in other time-stamps issued earlier. So a sequence of time-stamps is generated. If a dishonest issuer wants to fraudulently alter a particular time-stamp, it has to alter all the time-stamps relating to that time-stamp. Thus it is more difficult for an adversary to manipulate a time-stamp in the linking scheme than in the simple scheme.

Distributed Scheme: In this scheme a time-stamp is created with the cooperation of multiple issuers. One of the prime targets of this scheme is to strengthen security against the dishonest issuers manipulation of a time-stamp by sharing the secret data used to generate a time-stamp among the issuers.

1.6 E-cash System

Online digital content transactions through e-commerce are growing exponentially. In this respect, well-designed electronic payment schemes and high-quality digital contents are two critical factors [21]. Untraceable electronic cash schemes make

TABLE 1.1: Strengths and Limitations of three time-stamping Schemes

Schemes	Strengths	Limitations
Simple Scheme	The system is relatively simple	It is necessary to assume that the issuer is the trusted third party.
Linking Scheme	The assumption that the issuer is the trusted third party is rendered unnecessary, for example, by the periodical publication of a part of a chain of time-stamps.	The system is relatively complicated because additional operations for linking all time stamps are needed.
Distributed Scheme	The assumption that the issuers are the trusted third parties is rendered unnecessary by sharing the secret data among multiple issuers.	The system is relatively complicated because multiple issuers generate a time-stamp cooperatively.

it possible for customers to pay the e-cash to the merchants through communication networks under privacy protection. Therefore, there is a need to invent new electronic payment protocols with strong cryptographic algorithms that will eventually replace present day paper-based cash schemes. Chaum suggested the first electronic cash system in 1983 [22].

An e-cash system has some well-defined features. We outline some of them below [21].

- *Anonymity:* The spender or the payer of the cash must remain anonymous. If the coin is spent legitimately, neither the merchant nor the bank can identify the payerme
- *Unreusability:* The digital cash cannot be copied and reused. Then we have to minimize the risks for forgery and establish a good authenticity system.
- *Unforgeability:* Only authorized parties (i.e. the bank) can produce digital coins.
- *Off-line Payment:* The transaction can be done off-line, meaning no communication with the central bank is needed during the transaction.
- *Transferability:* Electronic transactions are online ond off-line.
- *Divisibility:* Digital cash can be divided into smaller amounts.
- *Portability:* The security and use of digital cash is not dependent on any physical location. The cash can be transferred through computer networks into storage devices and vice versa

Three types of entities are usually present in an e-cash system.

- Payer or Consumer
- Payee or Merchant
- Financial network with whom both payer and payee have accounts (usually a Bank)

Three types of transactions take place in a normal e-cash system

- **Withdrawal:** the payer withdraws money in terms of digital coin
- **Payment:** the payer transfers the digital coin to the payee.
- **Deposit:** the payee transfers digital coin received to the bank account.

1.7 Our Contribution

In this thesis, we have contributed the following:

- We have proposed a new time-stamped signcryption scheme based upon hard computations such as DLP, IFP and CDH. We have analyzed the security of the scheme and also implemented it.
- We have designed an e-cash system based upon the proposed time-stamped signcryption scheme.

1.8 Organization of Thesis

In Chapter 2, we have given the literature survey which includes the review of a time-stamped signature scheme and one signcryption scheme. At the end we have given the mathematical preliminaries which have been used throughout the thesis.

In Chapter 3, we have proposed our new time-stamped signcryption scheme. Also the security analysis and the implementation result has been given in this chapter.

In Chapter 4, we have proposed a new e-cash system. Also the security is analysed in this chapter.

In Chapter 5 we provide conclusion of this thesis and future research directions.

Chapter 2

Literature Survey

2.1 Review of Time-stamped Signature Scheme

The existing time-stamped signature scheme is based upon DLP [23]. The signature consists of time-stamp which is properly authenticated and is provided by a trusted third party, known as Time-Stamping Service (TSS), which attaches the time-stamp without the knowledge of the content of the document. The signature can be verified universally, i.e., anyone can verify the validity of the signature using systems public parameters. The security of this scheme is proved based upon the assumption of some hard problems in computer science like integer factorization problem and DLP.

The scheme consists of 3 phases; key generation phase or the setup phase, the time-stamped signature generation phase and signature verification phase. There are three parties present in the existing time-stamped signature scheme: A signer who signs the document, a verifier who verifies the document and a TSS who provides time-stamp to the document. The operations of various phases are shown in Figure 1.In the proposed scheme, the following notations are used.

x_A: the private key of the signer.

y_A: the public key of the signer.

x_B: the private key of the TSS.

y_B: the public key of the TSS.

p: a large prime with 512 bits

q: a large prime with q|p–1.

g: an order q generator in Z_{p^*}

h(.): a collision resistant hash function.

2.1.1 Key Generation

The TSS chooses an integer n as the product of two large primes p and q such that $p = 2fp^{'} + 1$ and $q = 2fq^{'} + 1$, where f, $p^{'}$ and $q^{'}$ are all large primes. Then he chooses a generator $g \in Z_p^*$ with order q and a secure hash function h( ) such as SHA 1.Here p, g and h( ) are public system parameters which are authentically known to all users.

The TSS chooses his private key $x_B \in Z_q^*$ and computes his public key $y_B = g^{x_B}$(mod p). Similarly, the signer chooses his private key $x_A \in Z_q^*$ and computes $y_A = g^{x_A}(modp)$ as his public key.

2.1.2 Signature Generation:

This algorithm takes the message m, public and private keys of signer and that of TSS and outputs timestamped signature $\sigma = (l, S)$

The steps of time-stamped signature generation are given below.

Step 1: The signer computes f as follows and sends it to the TSS.

$$f = h(m) \tag{2.1}$$

Step 2: After receiving f, the TSS chooses $k \in Z_q^*$ and computes

$$r = f.g^{k+t}.y_B(modp) \tag{2.2}$$

Here t is the timestamp of the signature. The value of t may be current date or time of the signature generation. Then the TSS sends r to the signer

Step 3: After obtaining r, the signer computes l and partial signature (S_1) as follows.

$$l = h(m, r) \tag{2.3}$$

$$S_1 = l.x_A(modp) \tag{2.4}$$

After that, the signer sends S_1 to the TSS.

Step 4: The TSS computes S as follows and sends it to the signer.

$$S = k - S_1 + x_B(modp) \tag{2.5}$$

The signature of message m is $\sigma = (l, S)$ with timestamp t.

2.1.3 Signature Verification:

Anyone having signature $\partial = (l, S)$ with timestamp t of message m can verify as follows.

Step 1: First computes f and $r^{'}$ as given below.

$$f = h(m) \tag{2.6}$$

$$r' = f.y_A^l.g^{S+t}(modp) \tag{2.7}$$

Step 2: Then computes

$$l' = h(m, r') \tag{2.8}$$

If $l = l'$, then the signature is considered to be a valid one. Otherwise, it is rejected.

2.1.4 Correctness Verification and Security analysis:

Two well-known computationally hard problems: integer factorization and discrete logarithm problem [24] are the basis of the security of the existing time-stamped signature scheme.In the proposed scheme, p and q are very large primes, Hence the solving for p and q lies in the complexity of solving integer factorization problem. Also, to get k from r, an adversary has to solve discrete logarithm problem. Also the TSS cannot generate the full signature S by himself, as the partial signature is constructed using the secret key of signer and is secured under DLP assumption. The time-stamped signature (S, l) is indeed a valid authenticated signature from the trusted party TSS, the correctness of the signature is given below.

$$r' = f.y_A^l.g^{S+t}(modp) \tag{2.9}$$

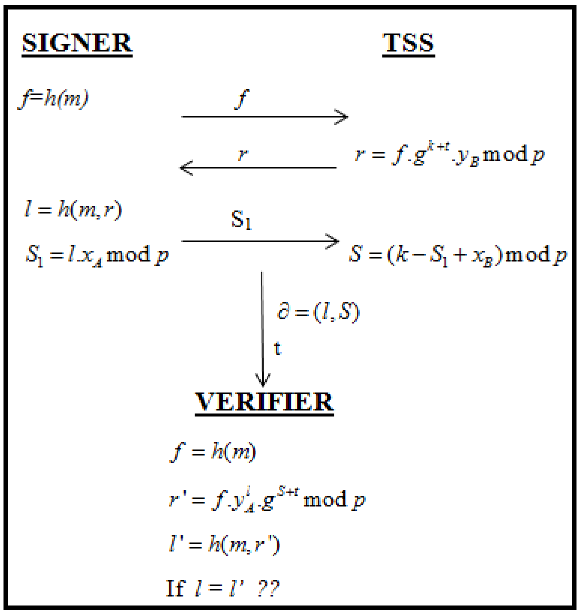


FIGURE 2.1: Operations of the Time-stamped Signature Scheme

$$= f.g^{x_A.l}.g^{k-S_1+x_B+t}(modp) \tag{2.10}$$

$$= f.g^{x_A.l}.g^{k-l.x_A+x_B+t}(modp) \tag{2.11}$$

$$= f.g^{x_A.l+k-l.x_A+x_B+t}(modp) \tag{2.12}$$

$$= f.g^{k+x_B+t}(modp) \tag{2.13}$$

$$= f.g^{k+t}.g^{x_B}(modp) \tag{2.14}$$

$$= f.g^{k+t}.y_B(modp) = r \tag{2.15}$$

The time-stamped signature (S, l) is universally verifiable. Anyone having access to the message and public parameters of the system, can verify the authenticity of the signature.

2.2 Review of Tso et.al. Signcryption Scheme:

The signcryption scheme proposed by Tso et.al. is a generic signcryption scheme with three parameters (c,r,s) of the signcrypted cipher text [25]. This scheme is the improvement of the Huang et. als scheme since the scheme overcomes the weak points of Huang et. als scheme [26]. Also this scheme provided public verifiability of the signature. Also the scheme supports the verification with only the message and Signcryption and the rest information provided by the receiver are not required. The scheme was mainly useful in the applications where the memory size is restricted.

2.2.1 Setup :

On input a security parameter 1 to the setup algorithm, the following outputs form the public parameters.

p, q : two large prime such that q is a prime divisor of p – 1 .

g: an element of Z_p^* of order q

H : $\{0, 1\}^* \in Z_q^*$: a collision resistant one way function.

2.2.2 KeyGen :

The KeyGen algorithm generates the public and private key pairs of signer and receiver.

Alices Private Key $x_A \in Z_q^*$

Alices Public key $y_A = g^{x_A}$(mod p)

Bobs Private Key $x_B \in Z_q^*$

Bobs Public key $y_B = g^{x_B}$(mod p)

2.2.3 Signcrypt :

To signcrypt a message $m \in Z_p \setminus$ {-1,0,1} Alice does the following steps

1. Chooses a random $no \in Z_q^*$, computes $e = y_B^k$ (mod p) and $e^{'} = e/gcd(e, p-1)$

2. Compute $c = m^{e^{'}}$ (mod p)

3. Computes $r = H(m, y_A, y_B, g^k modp)$

4. Computes $s = k - rx_A$ mod q

The signcrypt message is $\sigma = (c, r, s)$

2.2.4 Designcrypt :

Bob receives $\sigma = (c, r, s)$ from Alice and first recovers $\hat{e}$ and $e^{'}$ by computing $\hat{e} = y_B^s y_A^{rx_B}$ mod p and $e^{'} = \hat{e}/gcd(\hat{e}, p-1)$. Then he solves $e^{'}d = 1$ mod p–1 and then message by computing $\tilde{m} = c^d$ mod p. Bob accepts $\tilde{m}$ if and only if

$$r = H(\tilde{m}, y_A, y_B, y_A^r g^s modp)$$

2.2.5 Correctness :

The message m can be recovered by the receiver successfully since,

$$\hat{e} = y_B^s y_A^{rx_B} modp \tag{2.16}$$

$$= y_B^s g^{rx_B x_A} modp \tag{2.17}$$

$$= y_B^s y_B^{rx_A} modp \tag{2.18}$$

$$= y_B^{s+rx_A} modp \tag{2.19}$$

$$= y_B^k modp = e \tag{2.20}$$

Moreover $e^{'} = e/gcd(e, p-1)$, so $e^{'}$ and $p-1$ are co-primes and a d exists such that

$$e^{'}d = 1 \text{ mod p–1}$$

Therefore $c^d = m^{e^{'}d} = m$.

2.3 Mathematical Preliminaries

We used the following basic notation, definitions and models used throughout this thesis.

2.3.1 Notation and Terminology

All groups discussed in this thesis are assumed to be abelian. Groups of prime order have useful properties and are widely used in cryptography. All groups of prime order are cyclic.

Definition 1. A group G is cyclic if there is an element $g \in G$, such that for each $g^{'} \in G$, there is an integer a with $g^{'} = g^{a}$ .Such an element is called a generator of G. For any prime integer p, the field of integers modulo p is denoted by Z_p. The cyclic multiplicative group of nonzero elements in Z_p is denoted as Z_p^* [27].

2.3.2 Discrete Logarithmic Problem (DLP)

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. Briefly, if G is a finite group, the problem discrete logarithm in G is the following computational problem: given elements α and β in G, determine an integer x such that, $\alpha^x = \beta$ provided that such an integer exists [24].

2.3.3 Computational Diffie-Hellman problem

The Diffie-Hellman problem is stated as follows. If g is a generator of some group (typically the multiplicative group of a finite field) and x, y are randomly choosen integers. Consider a cyclic group G of order q. The CDH assumption states that, given (g, g^a, g^b) for a randomly choosen generator g and $a, b \in \{0, ..., q - 1\}$ it is computationally infeasible to compute the value of [6].

2.3.4 The Integer Factorization Problem

Definition 2: The integer factorization problem is the following: given a positive integer n, find its prime factorization; that is, write $n = p_1^{e_1} p_2^{e_2} ... p_k^{e_k}$ where the p_i are pairwise distinct primes and each $e_1 \geq 1$ [27].

2.3.5 Safe Primes

These primes are called ”safe” because of their relationship to strong primes. A prime number is a strong prime if $q + 1$ and $q - 1$ both have large prime factors. For a safe prime,$q = 2p + 1$, the integer p is a large prime factor. Safe primes are also important in the area of cryptography because of their use in discrete logarithm-based techniques like Diffie-Hellman key exchange. If $2p + 1$ is a safe prime, the multiplicative subgroup of numbers modulo. $2p + 1$ has a subgroup of large prime order. The reason for using safe primes is to minimize the modulus [23].

Chapter 3

Proposed Time-stamped Signcryption Scheme

In this chapter we propose the concept of a strong designated verifier time-stamped signcryption scheme. With this scheme, a user can anonymously signcrypts a message in collaboration with a trusted authority, known as Time Stamping System (TSS) and the ciphertext can only be deciphered by the designated verifier. The recipient knows that the information is from the user. However, except for the recipient, no one can verify the authenticity of the message. Our scheme is motivated by signcryption scheme, time-stamp and strong designated verifier signature scheme. In this scheme, only an authorized person can recover the message from a signcrypted text. This scheme establishes strong non repudiation between the signcrypter and a designated verifier. The proposed scheme is proved to be resistant against some active attacks and very much applicable in e-cash and e-voting systems.

There are three parties in the proposed scheme: the signer who signcrypts a document; the TSS, who is responsible for adding time-stamps into the signcryption; and the verifier, who checks the validity of the time-stamped signcryption. This scheme consists of three phases: key generation, signcryption generation and Designcryption. In the proposed scheme, we shall use the following notations.

x_S: the private key of the signer.

y_S: the public key of the signer.

x_T: the private key of the TSS.

y_T: the public key of the TSS.

x_V:	the private key of the Verifier.
y_V:	the public key of the Verifier.
p:	a large prime with 512 bits
q:	a large prime with q|p–1.
g:	an order q generator in Z_{p^*}
h(.):	a collision resistant hash function.

3.1 Key Generation:

The TSS chooses an integer n as the product of two large primes p and q such that $p = 2fp^{'} + 1$ and $q = 2fq^{'} + 1$, where f, $p^{'}$ and $q^{'}$ are all large primes. Then he chooses a generator $g \in Z_p^*$ with order q and a secure hash function h( ) such as SHA 1.Here p, g and h( ) are public system parameters which are authentically known to all users.

The TSS chooses his private key $x_T \in Z_q^*$ and computes his public key $y_T = g^{x_T}$(mod p). The Verifier chooses his private key $x_V \in Z_q^*$ and computes his public key $y_V = g^{x_V}$(mod p). Similarly, the signer chooses his private key $x_S \in Z_q^*$ and computes $y_S = g^{x_S}(modp)$ as his public key.

3.2 Signcryption Generation:

To signcrypt a message m, this algorithm needs the public key of verifier and the private keys of Signer and TSS (Time Stamping Service). This algorithm outputs time-stamped signcryption $\sigma = (c, r_2, s)$

The steps of time-stamped signcryption generation are given below.

Step 1: The signer computes f as follows and sends it to the TSS.

$$f = h(m) \tag{3.1}$$

Step 2: After receiving f, the TSS chooses $k_1 \in Z_q^*$ and computes

$$r_1 = g^{k_1}(modp) \tag{3.2}$$

$$e_1 = h(t, f) \tag{3.3}$$

$$s_1 = e_1.x_T + k_1 \tag{3.4}$$

Here t is the time-stamp of the signature. The value of t may be current date or time of the signature generation. TSS sends r_1, s_1, e_1, t to the signer

Step 3: In this step the signer first computes $e_1' = h(t, f)$ and compares it with e_1. If $e_1 = e_1'$, the time-stamp is accepted otherwise it is rejected and the signcryption process is aborted from here. If the time-stamp is accepted the signcryption process is followed as below.

The signer chooses a random number $k_2 \in Z_{q^*}$, then computes

$$e_2 = {y_V}^{k_2} modp \tag{3.5}$$

$$e_2' = e_2/gcd(e_2, p-1) \tag{3.6}$$

$$c = m^{e_2'} modp \tag{3.7}$$

$$r_2 = h(m, g^{k_2} modp) \tag{3.8}$$

$$s = k_2 - r_2.x_S - s_1 modp \tag{3.9}$$

The signcryption of message m is $\sigma = (c, r_2, s)$ with time-stamp t and it is sent to the verifier for designcryption.

3.3 Designcryption:

Step:1 After receiving $\sigma = (c, r_2, s)$ from the signer, the verifier first recovers $\hat{e_2}$ and e_2' by computing

$$\hat{e_2} = y_V^s.y_S^{r_2 x_V}.y_T^{e_1 x_V}.r_1^{x_V} \tag{3.10}$$

$$e_2' = \hat{e_2}/gcd(\hat{e_2}, p-1) \tag{3.11}$$

Step 2: Then he solves d from $e'd = 1$mod p-1 and then recover the message by computing

$$\tilde{m} = c^d modp \tag{3.12}$$

$$r_2' = h(\tilde{m}, (y_S^{r_2}.y_T^{e_1}.g^s.r_1) modp) \tag{3.13}$$

The verifier accepts $\tilde{m}$ if and only if $r_2 = r_2'$ Otherwise, it is rejected.

3.4 Correctness Verification:

The message m can be recovered succesfully by the verifier if the signer produced σ by the proposed scheme honestly, since

$$\hat{e_2} = y_V^s . y_S^{r_2 x_V} . y_T^{e_1 x_V} . r_1^{x_V} \tag{3.14}$$

$$= y_V^s . g^{x_V r_2 x_S} . g^{x_V e_1 x_T} . g^{x_V k_1}. \tag{3.15}$$

$$= y_V^s . y_V^{r_2 x_S} . y_V^{e_1 x_T} . y_V^{k_1} \tag{3.16}$$

$$= y_V^{s + r_2 x_S + e_1 x_T + k_1} \tag{3.17}$$

$$= y_V^{s + r_2 x_S + s_1} \tag{3.18}$$

$$= y_V^{k_2} \tag{3.19}$$

$$= e_2 \tag{3.20}$$

Moreover $e_2' = e_2/gcd(e_2, p-1)$, so e_2' and p-1 are co-primes and a d exists such that $e_2'.d = 1$ mod p-1. Therefore

$$c^d = m^{e_2'.d} = m \tag{3.21}$$

After $\tilde{m}$ is recovered, if $\tilde{m} = m$ then the equation $r_2 = r_2'$ will hold. Since

$$y_S^{r_2} . y_T^{e_1} . g^s . r_1 \tag{3.22}$$

$$= g^s . g^{x_S r_2} . g^{x_T e_1} . g^{k_1} \tag{3.23}$$

$$= g^s . g^{x_S r_2} . g^{s_1} \tag{3.24}$$

$$= g^{s + r_2 . x_S + s_1} \tag{3.25}$$

$$= g^{k_2} \tag{3.26}$$

Hence message $\tilde{m}$ will be accepted as the required message if and only if $r_2 = r_2'$.

3.5 Discussion and Security Analysis of the Proposed Scheme

Our scheme provides the 4 notion of security i.e authentication, confidentiality, integrity, and non-repudiation.In this section we will see how our scheme will satisfy

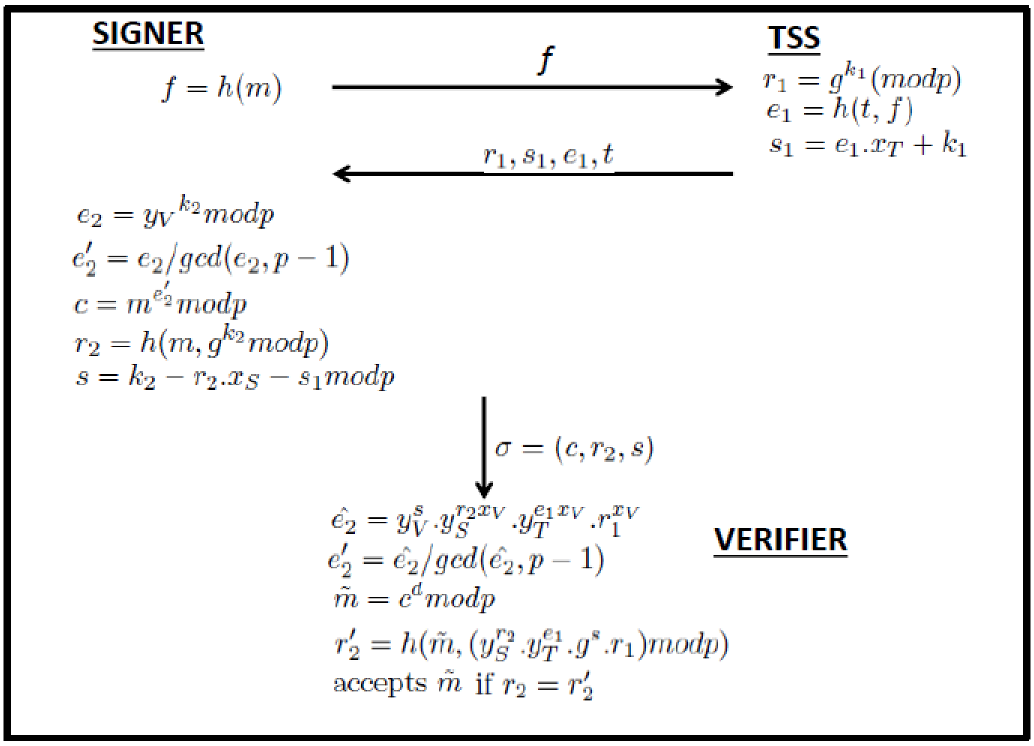

SIGNER
$f = h(m)$
f
TSS
$r_1 = g^{k_1}(modp)$
$e_1 = h(t, f)$
$s_1 = e_1.x_T + k_1$
r_1, s_1, e_1, t
$e_2 = {y_V}^{k_2} modp$
$e_2' = e_2/gcd(e_2, p-1)$
$c = m^{e_2'} modp$
$r_2 = h(m, g^{k_2} modp)$
$s = k_2 - r_2.x_S - s_1 modp$
$\sigma = (c, r_2, s)$
$\hat{e_2} = y_V^s . y_S^{r_2 x_V} . y_T^{e_1 x_V} . r_1^{x_V}$
$e_2' = \hat{e_2}/gcd(\hat{e_2}, p-1)$
$\tilde{m} = c^d modp$
$r_2' = h(\tilde{m}, (y_S^{r_2} . y_T^{e_1} . g^s . r_1) modp)$
accepts $\tilde{m}$ if $r_2 = r_2'$
VERIFIER

FIGURE 3.1: Operations of the Proposed Time-stamped Signcryption Scheme

all these notion of security.

First of all the signer creates a digest of the original message $f = h(m)$ to be sent to the TSS. This is done because the message should not be disclosed to the TSS. In this way the integrity and the confidentiality of the message is preserved in the first step. In the second step the time-stamping is done. The parameters k_1 and the private key x_T are secret and only known to the TSS. Since our scheme provides absolute temporal authentication, absolute time is integrated in this step, which may be the current system time. The TSS generates a partial signature s_1 which includes the private key x_T and e_1 which in turn includes the time-stamp t. The overall operations of the proposed scheme have been shown in the Figure1.

To be able to verify signed documents in a long period of time, the TSS must be trustable. However, if a TSS is damaged or hacked, then all issued time-stamps become invalid. To resolve the problem, after issuing an amount of time-stamps which is called a session, all issued time-stamps in one round are published. This feature makes it possible that the TSS does not have to be unconditionally trusted all the times because all publicized time-stamps cannot be forged. Therefore, we only have to trust TSS during the session that a time-stamp was requested and issued for a particular message.

In the next step signer first validates the time-stamp integrated by TSS by comparing $e_1 = e_1'$ or not. Then signer generates k_1 which is private and only known to signer. Then it computes e_2 and e_2' to do the encryption. then finally the signature is generated after computing r_2. Finally c,r_2, s together constitute the signcryption of the given message m. In this step encryption and signature is done in one logical step unlike "signature then encryption".

The original message can only be derivd from the signcrypted message only and only by the deisgnated verifier since the calculation includes the private parameter x_V of the verifier. Hence the confidentiality is preserved. Again to get the message the verifier uses the public key of signer y_S, which shows that the our scheme preserves authenticity.

Our scheme is designed in such a way that neither the TSS nor the signer can generate a complete signcryption all alone, because to generate the signcryption we need s_1 and for that the secret parameters of TSS must be known which is not possible. So signer can not generate it alone and in the same way the TSS can not know the private parameters of signer, so TSS also can not generate it alone.

If the signer denies the signature of the message m, the verifier can prove the dishonesty of the signer by opening m with the original signcryption $\sigma = (c, r_2, s)$. With this information anyone can veriify the validity of the original signcryption by checking

$$r_2' = h(\tilde{m}, (y_S^{r_2}.y_T^{e_1}.g^s.r_1)modp) \tag{3.27}$$

This shows that our scheme preserves the non-repudiation property.

Attack 1: An adversary tries to reveal the secret key x from the public parameters.

Security analysis: The adversary can tries to obtain x_S or x_T or x_V from public keys y_S, y_T and y_V has to solve the equation of form $Y = g_x$ (mod n), which is clearly a discrete logarithm problem. An adversary tries to forge a signature have to obtain secret parameters of both signer and TTS, whose security lies in the difficulty of solving integer factorization and discrete logarithm problem.

Attack 2: An adversary tries to directly forge the time-stamp.

Security analysis: To forge the time-stamp the adversary has to compute s_1 for which two secret parameters x_T and k_1 are needed. To get x_T adversary has to solve the equation $y_T = g^{x_T}$(mod p) which is clearly a DLP. Again to get k_1 it has to solve $r_1 = g^{k_1}$ which is again a DLP.

Attack 3: An adversary tries to derive the message form the ciphertext.

Security analysis: The message m can not be derived from the ciphertext $c = m^{e_2'}$ mod p becaus of the discrete log problem (DLP). To break the confidentiality of the scheme one must know e_2, e_2' or d. on the otherhand according to our scheme, the adversary cannot derive e_2' and d without knowing e_2 where $e_2 = {y_V}^{k_2} mod p$ is the common secret between signer and receiver.

Attack 4:A dishonest signer tries to forge the time-stamped signcryption all alone.

Security analysis: To do this the signer has to compute s_1 for which two private parameters x_T and k_1 are needed. To get the these parameters the signer has to solve DLP, which is computationally hard problem.

Attack 5:A dishonest TSS tries to forge the time-stamped signcryption all alone.

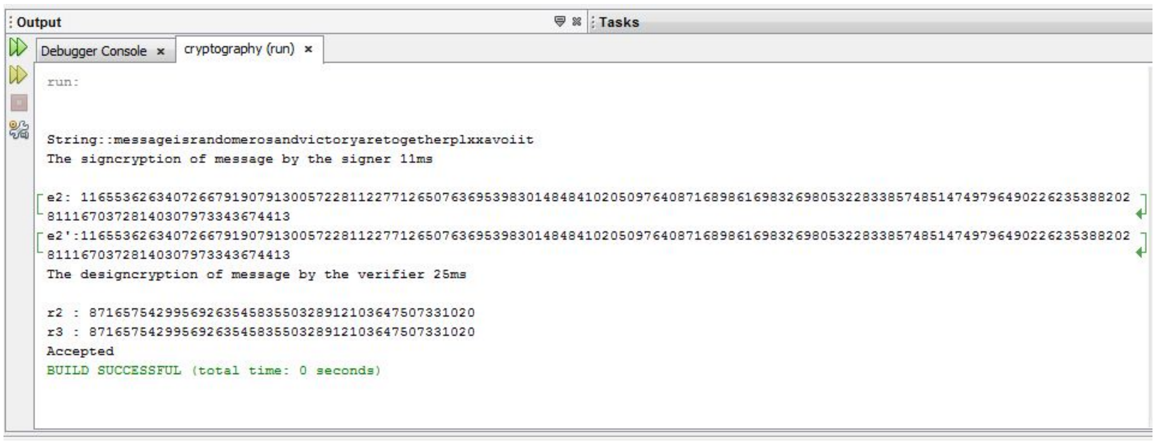
Security analysis: To do this the TSS has to compute s for which two private parameters x_S and k_2 are needed. Again to get the these parameters the signer has to solve DLP, which is computationally hard problem.

3.6 Performance Evaluation and Implementation Result

The proposed scheme is implemented in Java using java.security package. We have taken input as different size of messages and the time for Signcryption generation and designcryption is compared for different message sizes. The comparison is given in the following table

Table 3.1: Performance Comparison of the proposed scheme for different sizes of message

Message Size (in kb)	Signcryption Time (in ms)	Designcryption Time (in ms)	Length of Signcryption (in bytes)
0.05	11	25	64
0.5	18	30	64
5	189	34	64
50	557	40	64



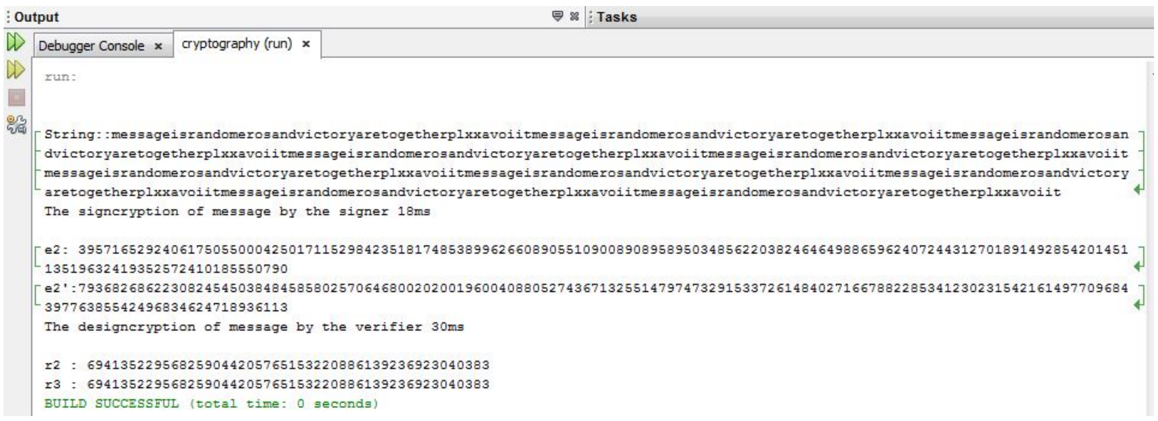
```
run:


String::messageisrandomerosandvictoryaretogetherplxxavoiit
The signcryption of message by the signer 11ms

e2: 11655362634072667919079130057228112277126507636953983014848410205097640871689861698326980532283385748514749796490226235388202
81116703728140307973343674413
e2':11655362634072667919079130057228112277126507636953983014848410205097640871689861698326980532283385748514749796490226235388202
81116703728140307973343674413
The designcryption of message by the verifier 25ms

r2 : 8716575429956926354583550328912103647507331020
r3 : 8716575429956926354583550328912103647507331020
Accepted
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 3.2: Screenshot of the output for message size 0.05KB



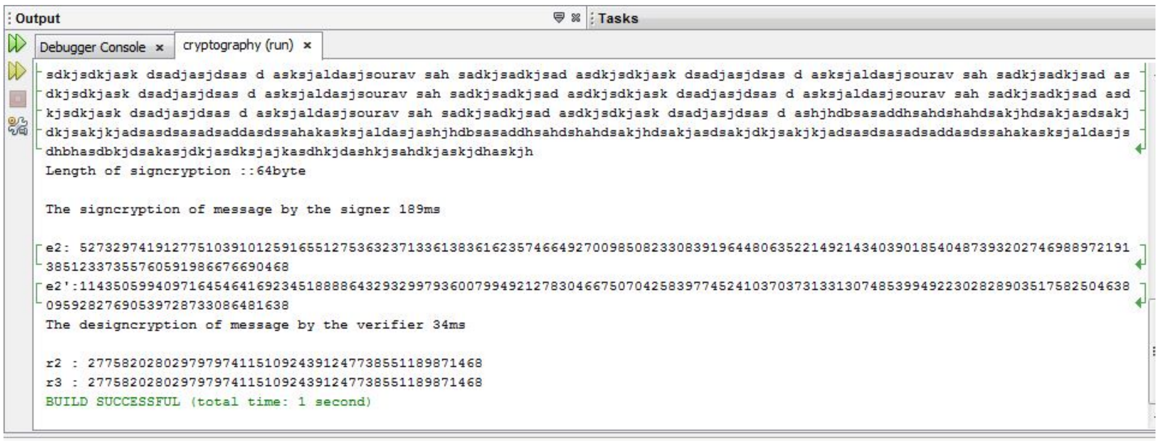
```
run:


String::messageisrandomerosandvictoryaretogetherplxxavoiitmessageisrandomerosandvictoryaretogetherplxxavoiitmessageisrandomerosan
dvictoryaretogetherplxxavoiitmessageisrandomerosandvictoryaretogetherplxxavoiitmessageisrandomerosandvictoryaretogetherplxxavoiit
messageisrandomerosandvictoryaretogetherplxxavoiitmessageisrandomerosandvictoryaretogetherplxxavoiitmessageisrandomerosandvictory
aretogetherplxxavoiitmessageisrandomerosandvictoryaretogetherplxxavoiitmessageisrandomerosandvictoryaretogetherplxxavoiit
The signcryption of message by the signer 18ms

e2: 39571652924061750550004250171152984235181748538996266089055109008908958950348562203824646498865962407244312701891492854201451
13519632419352572410185550790
e2':79368268622308245450384845858025706468002020019600408805274367132551479747329153372614840271667882285341230231542161497709684
39776385542496834624718936113
The designcryption of message by the verifier 30ms

r2 : 694135229568259044205765153220886139236923040383
r3 : 694135229568259044205765153220886139236923040383
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 3.3: Screenshot of the output for message size 0.5KB



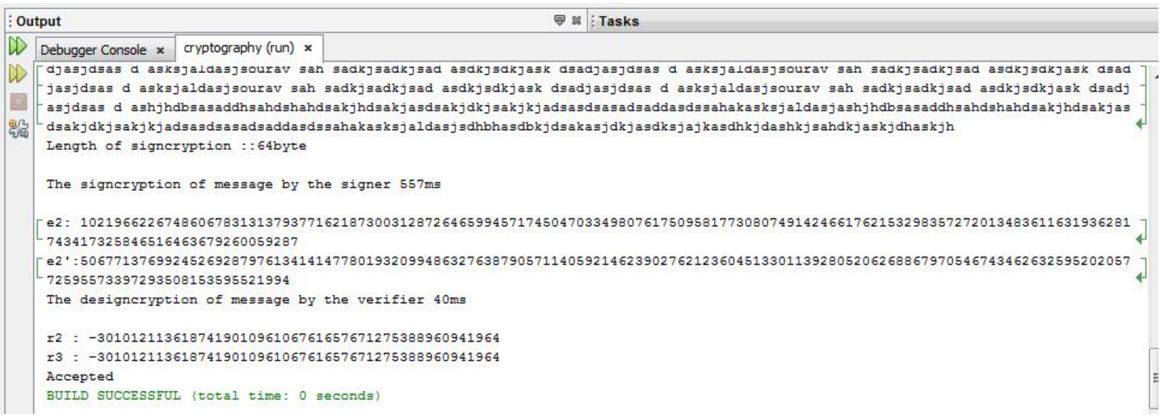
```
sdkjsdkjask dsadjasjdsas d asksjaldasjsourav sah sadkjsadkjsad asdkjsdkjask dsadjasjdsas d asksjaldasjsourav sah sadkjsadkjsad as
dkjsdkjask dsadjasjdsas d asksjaldasjsourav sah sadkjsadkjsad asdkjsdkjask dsadjasjdsas d asksjaldasjsourav sah sadkjsadkjsad asd
kjsdkjask dsadjasjdsas d asksjaldasjsourav sah sadkjsadkjsad asdkjsdkjask dsadjasjdsas d ashjhdbsasaddhsahdshahdsakjhdsakjasdsakj
dkjsakjkjadsasdsasadsaddasdssahakasksjaldasjashjhdbsasaddhsahdshahdsakjhdsakjasdsakjdkjsakjkjadsasdsasadsaddasdssahakasksjaldasjs
dhbhasdbkjdsakasjdkjasdksjajkasdhkjdashkjsahdkjaskjdhaskjh
Length of signcryption ::64byte

The signcryption of message by the signer 189ms

e2: 52732974191277510391012591655127536323713361383616235746649270098508233083919644806352214921434039018540487393202746988972191
38512337355760591986676690468
e2':11435059940971645464169234518888643293299793600799492127830466750704258397745241037037313313074853994922302828903517582504638
09592827690539728733086481638
The designcryption of message by the verifier 34ms

r2 : 27758202802979797411510924391247738551189871468
r3 : 27758202802979797411510924391247738551189871468
BUILD SUCCESSFUL (total time: 1 second)
```

Figure 3.4: Screenshot of the output for message size 5KB



```
djasjdsas d asksjaldasjsourav sah sadkjsadkjsad asdkjsdkjask dsadjasjdsas d asksjaldasjsourav sah sadkjsadkjsad asdkjsdkjask dsad
jasjdsas d asksjaldasjsourav sah sadkjsadkjsad asdkjsdkjask dsadjasjdsas d asksjaldasjsourav sah sadkjsadkjsad asdkjsdkjask dsadj
asjdsas d ashjhdbsasaddhsahdshahdsakjhdsakjasdsakjdkjsakjkjadsasdsasadsaddasdssahakasksjaldasjashjhdbsasaddhsahdshahdsakjhdsakjas
dsakjdkjsakjkjadsasdsasadsaddasdssahakasksjaldasjsdhbhasdbkjdsakasjdkjasdksjajkasdhkjdashkjsahdkjaskjdhaskjh
Length of signcryption ::64byte

The signcryption of message by the signer 557ms

e2: 1021966226748606783131379377162187300312872646599457174504703349807617509581773080749142466176215329835727201348361631936281
743417325846516463679260059287
e2':50677137699245269287976134141477801932099486327638790571140592146239027621236045133011392805206268867970546743462632595202057
72595573397293508153595521994
The designcryption of message by the verifier 40ms

r2 : -301012113618741901096106761657671275388960941964
r3 : -301012113618741901096106761657671275388960941964
Accepted
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 3.5: Screenshot of the output for message size 50KB

Chapter 4

Proposed e-cash System

In this chapter we present the proposed e-cash system which preserves the basic security requirements of an e-cash system. Our proposed e-cash system is based upon the proposed Time-stamped Signcryption scheme discussed in the last chapter. In this e-cash system we have 4 parties.

- **Bank** which pays the customer and merchant the requested amount if they satisfy the required condition.
- **Customer (Payer)** who withdraws some amount from bank and pays to the merchant
- **Merchant(Payee)** who receives the payment from the customer and deposits to the bank
- **TSS (Time Stamping System)** who attaches the time-stamp to the digital coin.

Three types of transactions are there in the proposed system

- **Withdrawal:** the payer withdraws money in terms of digital coin .
- **Payment:** the payer transfers the digital coin to the payee.
- **Deposit:** the payee transfers digital coin received to the bank account.

In the proposed system, we shall use the following notations.

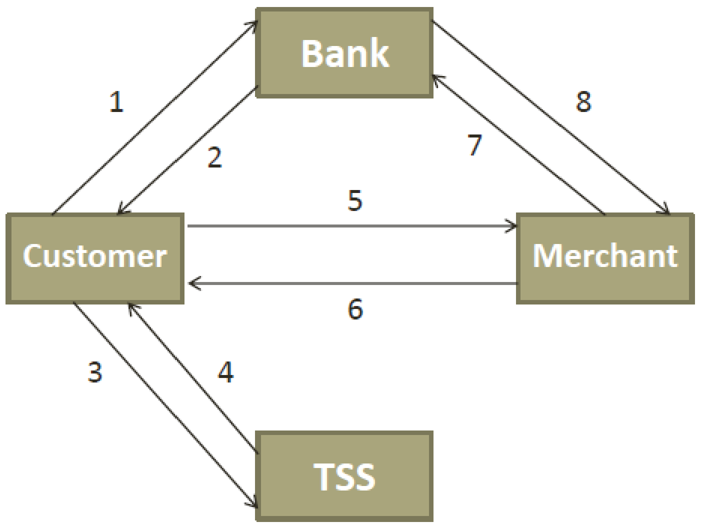


FIGURE 4.1: Proposed e-cash System

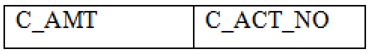
x_C: the private key of the Customer.
y_C: the public key of the Customer.
x_M : the private key of the Merchant.
y_M : the public key of the Merchant.
x_T : the private key of the TSS.
y_T : the public key of the TSS.
x_B : the private key of the Bank.
y_B : the public key of the Bank.
$H(.)$: a collision resistant hash function.

$Sign()$: A signing algorithm (e.g Elgamal or Schnorr Signature)
$Verify()$: A verification algorithm of the corresponding signing algorithm
$Signcrypt()$: The Signcryption algorithm of proposed scheme in previous chapter
$Designcrypt()$: Designcryption algorithm of proposed scheme in previous chapter
$Time\_stamp()$: Time-stamping algorithm performed by TSS to add time-stamp

4.1 Steps in the Proposed e-cash System

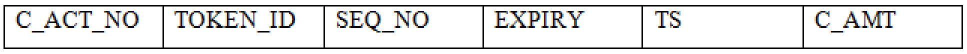
Step 1: First of all the customer creates customer info message (CI) as follows

C_AMT: the amount to be requested to the bank by the customer C_ACT_NO: account number of the customer.



C_AMT	C_ACT_NO

FIGURE 4.2: Customer Information (CI)



C_ACT_NO	TOKEN_ID	SEQ_NO	EXPIRY	TS	C_AMT

FIGURE 4.3: Digital Coin Information (COIN)

Customer sends a request to the bank to issue digital coin. The request is in the form of a digital signature on the CI with customers private key.

$$S_1 = Sign(x_C, CI) \tag{4.1}$$

Customer sends S_1 along with CI to bank

Step 2: When Bank receives S_1 and the CI, it first verifies the authenticity of the request with the public key of the customer (y_C).

$$verify(S_1, y_C, CI) \tag{4.2}$$

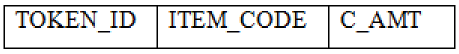
After successful verification, bank first checks whether the ACT_NO exits in the bank or not and if exists then whether the account has balance more than the requested amount.(C_AMT). If balance is available bank creates a digital coin (COIN) which contains the following

C_ACT_NO: Customers account number
TOKEN_ID: COIN number
SEQ_NO: Transaction number
EXPIRY: date and time of COINs expiry after which COIN becomes invalid
TS: Time-stamp on the COIN i.e the time and date when the COIN is created
C_AMT: the amount the COIN holds.

Then bank puts its own digital signature on the COIN with its own private key.

$$S_2 = Sign(x_B, COIN) \tag{4.3}$$

Bank sends the signatureS_2 along with the COIN to the customer.



TOKEN_ID	ITEM_CODE	C_AMT

FIGURE 4.4: Order Information (OI)

Step 3: When customer receives COIN and the signature S_2, it first verifies the authenticity of the COIN with the public key of the bank

$$verify(S_2, y_B, COIN) \tag{4.4}$$

If verified the COIN is accepted by the customer. Now the customer creates a digest of the COIN with a collision less hash function

$$h = H(COIN) \tag{4.5}$$

Customer sends this digest h to the TSS to get a time-stamp.

Step 4: TSS puts time-stamp according to steps of the proposed time-stamped Signcryption scheme discussed in previous chapter

$$T = Time\_stamp(h) \tag{4.6}$$

T is sent back to the customer

Step 5: After receiving the time-stamp (T), customer generates the Signcryption on the coin

$$\sigma = Signcrypt(x_C, y_M, COIN, T) \tag{4.7}$$

Now the customer creates the order information (OI) which contains the item information to be purchased and the amount to be paid.

TOKEN_ID: ID of the digital COIN
ITEM_CODE: The item code to be purchased from the merchant
C_AMT: Amount to be paid by customer to merchant

Then the customer puts its own signature on the OI.

M_ACT_NO	M_AMT

FIGURE 4.5: Merchant Information (MI)

$$S_3 = Sign(X_c, OI) \tag{4.8}$$

Customer sends both the signature S_3 and the signcrypted COIN (σ) along with the OI to the merchant.

Step 6: Merchant verifies the authenticity of the signatureS_3 with the public key of customer

$$verify(S_3, y_C, OI) \tag{4.9}$$

If the signature is verified successfully, it accepts the signcrypted COIN (σ) and OI both and sends an acknowledgement to the customer. The acknowledgement can be a simple message or e-mail.

Step 7: Now merchant creates merchant information (MI) to send to the bank to deposit the COIN and get the required amount to his account.

M_ACT_NO: Merchants account number M_AMT: The amount merchant is expected to get from the bank

Then merchant puts its own signature on the MI with its private key x_M

$$S_4 = Sign(x_M, MI) \tag{4.10}$$

Merchant then sends both the signcrypted $COIN(\sigma)$ and S_4 to the Bank along with the MI.

Step 8: After receiving signcrypted COIN (σ),S_4 and MI, Bank first of all verifies the authenticity of the signature.

$$verify(S_4, y_M, MI) \tag{4.11}$$

If verified successfully, that means the MI is from the authorized merchant. Then Bank designcrypts the signcrypted COIN using the steps of the proposed time-stamped signcryption scheme.

$$Designcrypt(\sigma, MI, y_B, y_T, y_C) \tag{4.12}$$

In the designcryption process it verifies that the COIN is authenticated one and also the COINs content is decrypted. Then Bank compares C_AMT from COIN and M_AMT from MI. If both match, then Bank credits that much amount to the merchants account no (M_ACT_NO) and a successful acknowledgement is sent to the Merchant. Else a negative acknowledgement is sent to the merchant stating that there is some mismatch and the case is resolved between customer and merchant.

4.2 Security Analysis of the Proposed e-cash System

THEOREM 1- *If the time-stamped signcryption is secure against forgery then the digital coin (COIN) is also unforgeable.*

Proof: To forge the digital coin COIN, the adversary has to generate a valid time-stamped signcryption scheme which means it has to get the private keys of the Customer and also the TSS. To get the private keys the adversary has to solve the DLP, which has been assumed to be computationally hard. Hence our digital coin (COIN) is unforgeable.

THEOREM 2- *The customer can pay the COIN full anonymously without revealing its identification and other information.*

Proof: The customer account information is included in the digital coin COIN which is sent to the merchant in a signcrypted format. Since our signcryption scheme is unforgeable, hence the merchant cannot break it and so it neither can know the identification of customer or its account number. Hence the scheme provides confidentiality to the active COIN and confirms the anonymity of the customer.

Chapter 5

Conclusion and Future Scope

In this thesis we propose a strong designated verifier time-stamped signcryption scheme based upon two hard problems i.e. discrete logarithm problem and integer factorization problem. This scheme preserves all the required security properties such as authenticity, integrity, confidentiality and non-repudiation. Neither the signer, nor the TSS can produce a valid signcryption all by themselves. Using the time-stamp we can solve the case when the signer himself repudiates the signing, claiming that has accidentally lost his private key. Also we have shown the few active attacks and how our scheme is resistant against those attacks. The proposed scheme can have wide applications in real life scenarios, such as, in e-voting, online auction, online lottery and transfer of patents where both confidentiality and authenticity is highly required.Also we propose an e-cash system based on our proposed time-stamped signcryption scheme which confirms the notion of e-cash securities like anonymity of the spender, unforgeablity of the digital coin, prevention of double spending

In future research can be done on our scheme to lower its computation cost and communication overhead. Also research can be done to incorporate time-stamping feature to some of the highly proved secured signcryption schemes which can be applicable to highly security sensitive application like e-bidding, e-voting, e-transactions etc.

Bibliography

[1] Behrouz A. Forouzan. *Cryptography and Network Security.* Tata McGraw-Hill, 2007.

[2] William Stallings. *Cryptography and Network security: Principles and Practices.* Prentice Hall Inc., 1999.

[3] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(5):644–654, 1976.

[4] Rivest, Shamir, and Adleman. A method for obtaining digital signatures and public key cryptosystems. In *SIMMONS: Secure Communications and Asymmetric Cryptosystems*, 1982.

[5] Goldwasser, Micali, and Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SICOMP: SIAM Journal on Computing*, 17, 1988.

[6] Pointcheval and Stern. Security proofs for signature schemes. In *EUROCRYPT: Advances in Cryptology: Proceedings of EUROCRYPT*, 1996.

[7] Zheng. Digital signcryption or how to achieve cost(signature & encryption) ¡¡ cost(signature) + cost(encryption). In *CRYPTO: Proceedings of Crypto*, 1997.

[8] Petersen and Michels. Cryptanalysis and improvement of signcryption schemes. *IEEPCDT: IEE Proceedings on Computers and Digital Techniques*, 145, 1998.

[9] J. Mao J Zhang. A novel identity based multi-signcryption scheme. *Computer Communications*, 32:14–18, 2009.

[10] H. Du Z. Jin, Q.Wen. An improved semantically-secure identity-based signcryption scheme in the standard model. *Computers and Electrical Engineering*, 36:545–552, 2010.

[11] C. H. Tan. Analysis of improved signcryption scheme with key privacy. *Information Processing Letters*, 99:135–138, 2006.

[12] Chung Ki Li and Duncan S. Wong. Signcryption from randomness recoverable public key encryption. *Inf. Sci*, 180(4):549–559, 2010.

[13] Baek, Steinfeld, and Zheng. Formal proofs for the security of signcryption. *JCRYPTOL: Journal of Cryptology*, 20, 2007.

[14] Masashi Une. The security evaluation of time stamping schemes: The present situation and studies, December 21 2001.

[15] Z. Shao. Security of the design of time-stamped signature. *Journal of Computer and System Sciences*, 72:690–705, 2006.

[16] H.T. Yeh H.M. Sun, B.C. Chen. On the design of time-stamped signatures. *Journal of Computer and System Sciences*, 68:598–610, 2004.

[17] D. Bayer, S. Haber, and W. S. Stornetta. Improving the efficiency and reliablility of digital time-stamping. In *Sequences '91: Methods in Communication, Security, and Computer Science*, pages 329–334, Berlin, 1992. Springer-Verlag.

[18] International Organization for Standardization and International Electrotechnical Commission. Iso/iec working draft 18014-1: Information technology - security techniques -time stamping services -part 1: Framework. Technical report, 2000.

[19] Henri Massias and Jean Jacques Quisquater. Time and cryptography. Technical report, 1997.

[20] Stuart Haber and W. Scott Stornetta. How to time-stamp a digital document. *Journal of Cryptology*, 3(2):99–111, 1991.

[21] Ziba Eslami and Mehdi Talebi. A new untraceable off-line electronic cash system. *Electronic Commerce Research and Applications*, 10(1):59–66, 2011.

[22] Chaum, Fiat, and Naor. Untraceable electronic cash. In *CRYPTO: Proceedings of Crypto*, 1988.

[23] S. Baral S. Mohanty, B. Majhi. A novel time-stamped signature scheme based upon dlp. In *1st International conference on Recent Advances in Information Technology (RAIT)*, pages 6–10, 2012.

[24] T. ElGamal. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31:469–472, 1985.

[25] Raylin Tso, Takeshi Okamoto, and Eiji Okamoto. An improved signcryption scheme and its variation. pages 772–778. IEEE Computer Society, 2007.

[26] Huang and Chang. An efficient convertible authenticated encryption scheme and its variant. In *ICIS: International Conference on Information and Communications Security (ICIS), LNCS*, 2003.

[27] Paul C. van Oorschot Alfred J. Menezes and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

Dissemination

1. Sujata Mohanty, **Sanjib Kumar Baral, Sourav Dash**, Banshidhar Majhi A Designated Verifiable Time-stamped Signcryption Scheme in *The Fourth International Workshop on Ad-hoc, Sensor and Ubiquitous Computing (ASUC 2013)*, Chennai, India, July 13-15, 2013 (Accepted)