ISSN : 2231-0711

Mahmood Khalel Ibrahim et al | International Journal of Computer Science Engineering and Technology(IJCSET) | April 2014 | Vol 4, Issue 4,137-142

Secure SMS System for E-Commerce Applications

Mahmood Khalel Ibrahim, Wasan Zaki Ameen

College of Information Engineering,

Al-Nahrain University,

Baghdad, Iraq

***Abstract*— In this paper, a simple e-commerce application for shopping is presented with implementing a new version of Diffie-Hellman protocol, to insure mutual authentication between client and server along with exchanging key securely through unsecure channel.**

The proposed system provides the major security services which are Authenticity, Integrity and Confidentiality, by implementing Zero Knowledge Proof (ZKP) for authenticity, keyed massage authentication code algorithm (HMAC-SHA1) for integrity and Advanced Encryption Standard (AES) for Confidentiality.

***Keywords*— Authenticity, Integrity, Confidentiality, Zero Knowledge Protocol, Diffie-Hellman, AES, HMAC.**

I. INTRODUCTION

SMS is a very popular wireless service throughout the world. It is the transmission of alphanumeric message between two parties. It enables the communication between the mobile subscribers and external systems such as paging, electronic mail and voice-mail systems. It will be the most attractive and effective service for future commercial use [1].

SMS is a part of Global System for Mobile Communication (GSM) networks that allows the alphanumeric message up to 160 characters to be sent and received via the network operator's SMS center to the mobile subscribers. If the subscriber is not reachable, then SMS are stored in the GSM operator's SMS center and delivered when it is reachable. The existing SMS is the transmission of just plaintext. It can be easily read by the intruder or even the persons of the operator. Therefore, it is not secured enough for M-Commerce or mobile banking [2].

II. ARCHITECTURE OF SMS IN GSM NETWORK

Short messages are delivered in GSM signaling channels between the Mobile Station (MS) and the Base Transceiver Station (BTS). Fig.1 shows the basic architecture for GSM-SMS. The messages flow as normal calls, but they are routed from the MSC to a Short Message Service Center (SMSC). The SMSC stores the message until it can be delivered to the recipient(s) or until the message's validity time has elapsed. The recipient can be a normal MS user or a SMS gateway. The gateways are servers that are connected to one or more SMSCs to provide SMS applications for the MS users. These applications include ring tone and icon delivery, entertainment, bank services, and many other beneficial services [3].

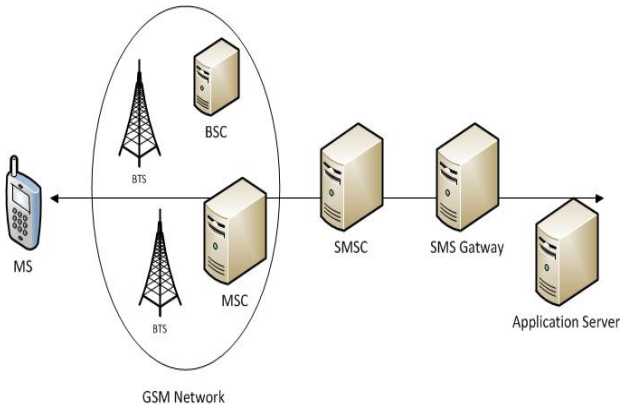

Fig. 1 SMS Network

III. SMS SECURITY CHALLENGES

SMS plays a very vital role in the mobile banking or M-Commercial purpose because of its simplicity and cheapness. SMS based M-Commerce comes as an alternative for online transaction systems. But there is some security issues related to services of SMS for such M-Commerce or mobile banking. [4]. The services includes:

A. *Confidentiality:* It is a process of protecting the data from disclosure to unauthorized adversaries during a communication. Confidentiality means that the message can only be read by authorized people (sender and receiver), since it prevents the eavesdroppers to observe the data during a communication. This service can be realized by encryption/decryption operations [5].

B. *Data Integrity:* It is concerned with the trustworthiness, origin, completeness, and correctness of information as well as the prevention of improper or unauthorized modification of information. Message authentication codes are used for maintaining the integrity of data [6].

C. *Identity authentication*: In order to prevent illegal intervention, system should verify the legitimacy of the user's identity. This process is called "Entity Authentication". This can be done with one of three kinds of witnesses: something know, something possessed, or something inherent. Password, challenge-response, zero-knowledge, and biometrics are the most common authentication methods [7].

IV. ZERO KNOWLEDGE AUTHENTICATION

A zero-knowledge proof (ZKP) is a proof of some statement which reveals nothing other than the veracity of the statement. The word "proof" here is not used in the traditional mathematical sense. Rather, a "proof", or equivalently a "proof system", is an interactive protocol by which one party (called the *prover*) wishes to convince another party (called the *verifier*) that a given statement is true. In ZKP, the *prover* proves that he/she knows a secret without revealing it [8].

ZKP model of computation defined as an interactive proof system (*P,V*), where *P* is a prover, and *V* is a verifier. Protocol (*P, V*) is for proving a language membership statement for a language over {*0.1*}. Let *L* be a language over {*0, 1*}*, for a membership instance $x \in L$, *P* and *V* must share the common input *x*, proof instance is denoted as (*P, V*) (*x*). *P* and *V* are linked by a communication channel over which they exchange a sequence, called proof transcript *a1, b1, a2, b2... an, bn*. Proof transcript interleaves prover's transcript and verifier's transcript. Each element *ai, bi* exchanged is bounded by polynomial time in |*x*| and proof instance (*P, V*) (*x*) must terminate in polynomial time in |*x*|. Upon completing the interaction, the output of the protocol should be of form (*P, V*) (*x*) $\in$ {*Accept, Reject*} representing *V*'s acceptance or rejection of *P*'s claim that $x \in L$ [9].

Three properties are expected from a zero-knowledge proof [10]:

A. Completeness:

An interactive proof (protocol) is complete if, given an honest Prover (P) and an honest Verifier (V), the protocol succeeds with overwhelming probability (i.e., the verifier accepts the prover's claim).

B. Soundness:

An interactive proof (protocol) is sound if there exists an expected polynomial time algorithm *M* with the following property: if a dishonest *prover* (impersonating *P*) can with non-negligible probability successfully execute the protocol with *V*, then *M* can be used to extract from this prover knowledge (essentially equivalent to *P*'s secret) which with overwhelming probability allows successful subsequent protocol executions.

C. Zero-knowledge:

A protocol has zero-knowledge property if it is able to be simulated in the following sense; there exists an expected polynomial-time algorithm (*simulator*) which can produce, upon input of the assertion(s) to be proven but without interacting with the real prover, transcripts indistinguishable from those resulting from interaction with the real prover.

V. ZKP PROTOCOL

The proposed ZKP based on D-H key exchange algorithm in the sense that both parties (the prover and the verifier) exchange non secret information and did not revealing secrets to get one identical secret key. This means that the prover can prove to the verifier that he knows the secret [9].

The prover (Alice) proves to the verifier (Bob) that she knows a secret by calculating the key (K) and resend Bob's reply (R_2) to the verifier (Bob) encrypted with the generated secret key (K). Bob will encrypt his own reply (R_2) with the generated secret key (K) and match the two encrypted information, if matched then Alice is verified, otherwise it is rejected.

The verifier also needs to prove to the prover that he is honest by sending his reply R_1 together with encrypted R_1, then the verifier decrypt R_1' by his key and match R_1 and R_1' , if they matched then the verifier is honest.

The used algorithm has been developed to resists the man-in-the-middle attack. For more details refer to [9]. Fig. 2 shows the procedure of the used ZKP protocol. The protocol performed as follows:

1. Alice (the prover) chooses a large random number x, such that $0< x <p$ and calculate $R_1 = g^x \bmod p$.
2. Alice sends R_1 to Bob.
3. Bob (the verifier) chooses another large random number y, such that $0< y < p$ and calculate $R_2 = g^y \bmod p$, $K_{Bob} = (R_1)^y \bmod p$, and $C_1 = E(K_{Bob}, R_2)$.
4. Bob sends ($R_2 \| C_1$) to Alice.
5. Alice, calculates $K_{Alice} = (R_2)^x \bmod p$, decrypt ($R_2' = D(K_{Alice}, C_1)$) and verify ($R_2 = R_2'$). If they matched then the process proceeds; otherwise the verifier is dishonest.
6. Alice encrypt ($C_2 = E(K_{Alice}, R_1 \| R_2)$) and send it to Bob.
7. Bob decrypt C_2 to get R_1' and R_2'.
8. Bob verify ($R_1 = R_1'$); if they are equal then Alice is verified (Accepted), otherwise it is a dishonest prover (rejected).

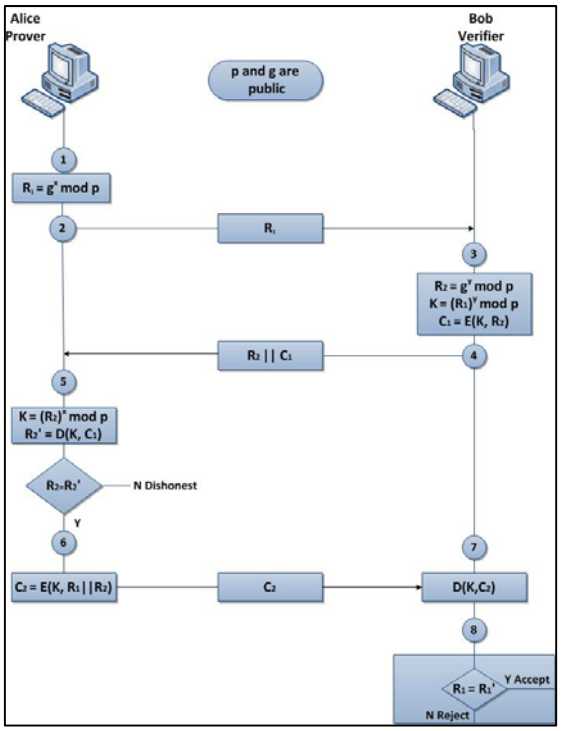

Fig. 2 Used ZKP protocol procedure

VI. SYSTEM ARCHITECTURE

To achieve the system robustness and flexibility to potential changes, the popular two-tier (layer) architecture is deployed in the system. The architecture is composed of two layers: the client application layer and server application layer. The two-layer architecture aims to make the application development and implementation easier and more efficient.

The client application layer offers the user a friendly and convenient entry to communicate with the system while the server application layer performs the controlling functionalities and manipulating the underlying logic connection of information flows and connection with the embedded database, which can store, index, manage and model information needed for this application.

To demonstrate the security services proposed by the system, a case study application (simple E-Commerce application for shopping) has been implemented. Fig. 3 shows a block diagram which illustrates the general structure of the system; the components of the system will be discussed in detail in the next sections.

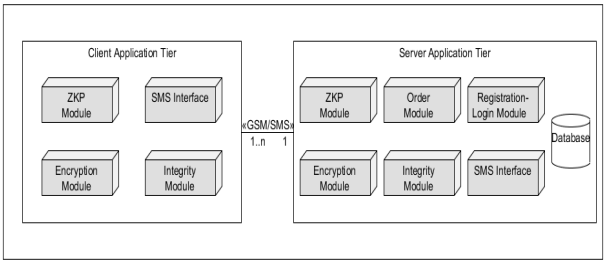

Fig. 3 General structure of the system (2-tier architecture model)

Fig. 4 illustrates the system architecture with the implemented functions.

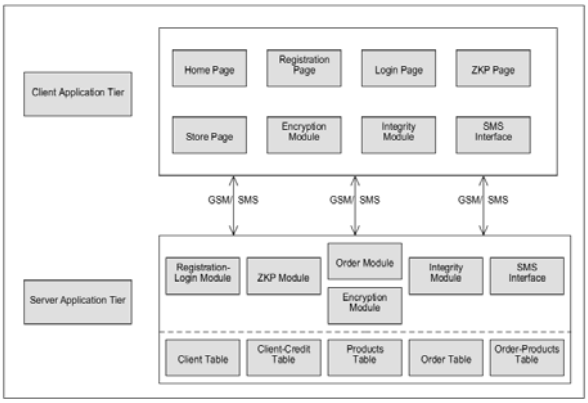

Fig.4 System Architecture

The client application layer is implemented by using android operating system and java programming language to generate the application. The user can access the data on the server through the android application.

The server application layer presents the server application logic and hiding technical details from the users, the application is generate by using windows operating system and java programming language. This layer works with H2 embedded database that used to design database of the system.

The two layers consist of the following modules:

A. *Registration-Login Module:*

After the client registers new account and sends registration SMS to the server, the server validate its integrity HMAC then add the information to the system database and send registration successful SMS to the client. After the registration process, the client should login and send login SMS to the server, the server validate its integrity HMAC and check the system database for the existence of the client, if the information is found then login successful SMS will be sent to the client and the client application go to ZKP page. Fig. 5 shows registration successful SMS in the client application.

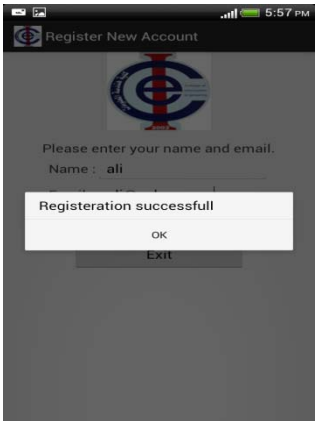

Fig. 5 Client Registration Successful SMS

Fig. 6 shows the server login successful SMS.

Fig. 6 Server Login Successful SMS

B. *ZKP Module:*

The client should enter his own secret number in android application, which is *x*, at the same time the server will enter his own secret number which is *y* in server application, the implemented protocol which is the modified Diffie-Hellman will calculate at each side R_1, R_2 at client and server side respectively, R_1 will passes to the server side using the formula:

$$R_1 = g^x \bmod p$$

Fig. 7 shows the ZKP module (R_1) SMS in client application.

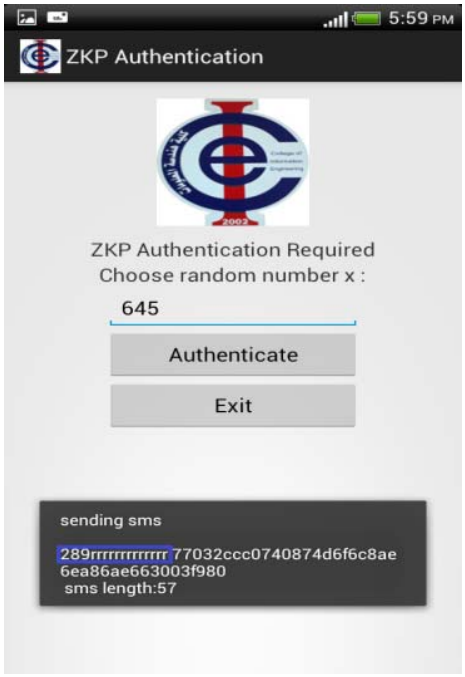

Fig. 7 R_1 SMS in Client ZKP Module

Then R_2 will be also passed to the client side along with C_1, in the following formula:

$$R_2 = g^y \bmod p$$

$$C_1 = E(K_{Server}, R_2)$$

Next step is calculating K at each side using the following formula respectively at the server and client side:

$$K_{Server} = (R_1)^y \bmod p$$

$$K_{Client} = (R_2)^x \bmod p$$

The same value of K will be established in both sides. This value is the secret key, which will be used in implementing the integrity and confidentiality security services. Fig. 8 shows the server ZKP module that calculate the R_2, C_1, K_{Server}.

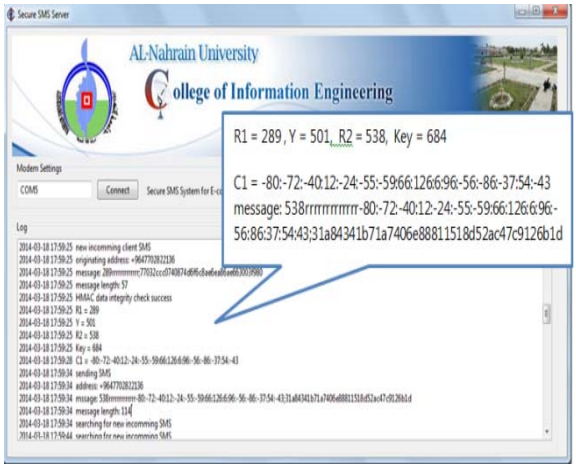

Fig. 8 Server ZKP Module

Then the client calculates C_2 and sends it to the server using the following formula:

$$C_2 = E(K_{Client}, R_1 || R_2)$$

Fig. 9 shows the ZKP module (C_2) SMS in the client application.

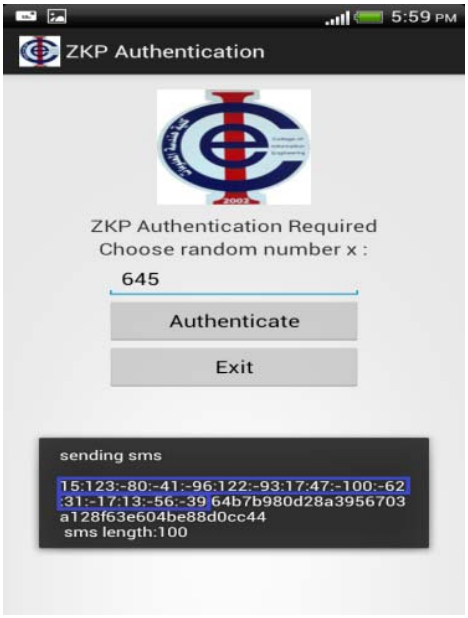

Fig.9 C_2 SMS in Client ZKP Module

The interactions are designed in a way that cannot lead to revealing or guessing the secret of both of the interconnected parties. After exchanging values the prover will be either accepted, or rejected and halt the system.

For illustration purpose fig. 10 presents the modified Diffie- Hellman mathematical computations.

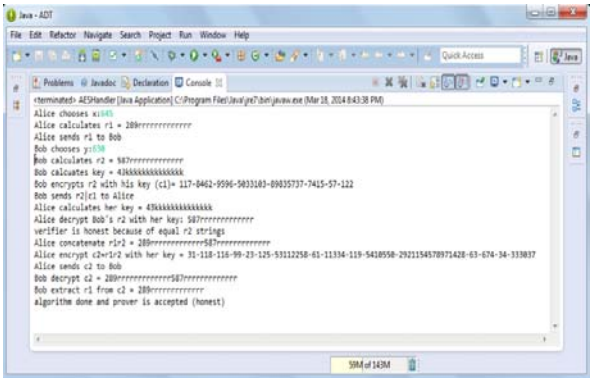

Fig. 10 Diffie-Hellman arithmetical computations

C. Order Module:

After a successful authentication of the user, the client selects the products from store in android application and enters their quantities, after that he should fill the field of credit card number and generate purchase order SMS with encrypted credit card number and send it to the server. When the server receives this SMS from the client, first it validates its integrity HMAC and then decrypts the credit card number. Then the server sends order status SMS to the client. Fig. 11 shows the purchase order SMS.

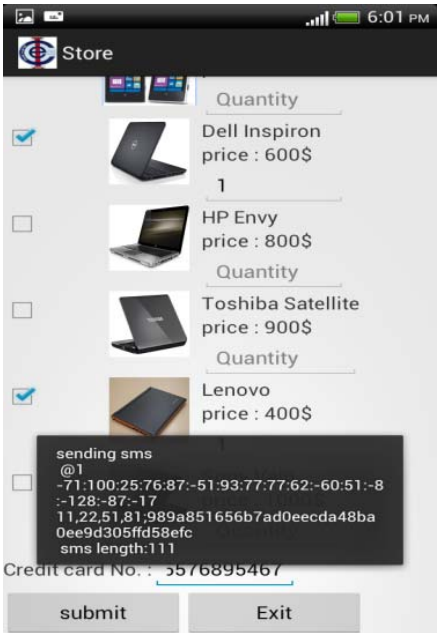

Fig.11 Purchase Order SMS

If the purchase is done successfully, the server sends purchase list to the client who contains client name, product name, quantity and the total price of the order. Fig. 12 shows the purchase list.

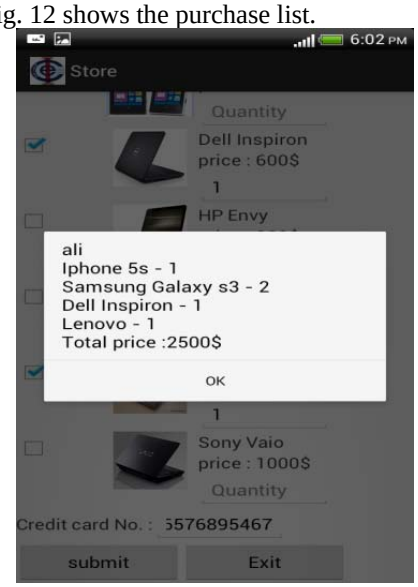

Fig. 12 Purchase List

D. Encryption Module:

This module is responsible for encryption and decryption processes required in ZKP protocol at ZKP module and encryption and decryption of the credit card number in order module. It uses AES encryption algorithm along with the key resulted from the ZKP protocol.

E. Integrity Module:

This module is responsible of calculating messages hash of all SMS sent between the client and server, and attach it to message after that, it represents the 40 bytes at the end of each SMS. This function is accomplished by using hash function HMAC-SHA1.

Fig. 13 shows the encryption and integrity process in client application.

Fig. 13 Encrypted credit card and HMAC

Fig. 14 shows the encryption and integrity process in server application.

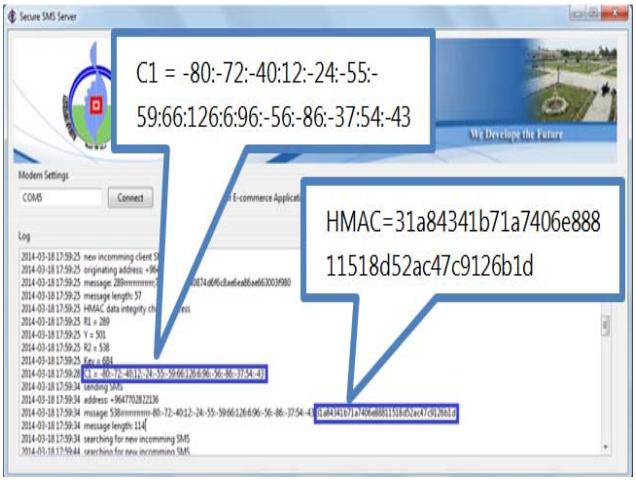

Fig. 14 Encrypted C_1 in server and HMAC

F. Database Module:

This module is responsible for storing information needed for the system and for optimizing the data access. The database of the system consists of five tables in order to store information; the client table stores the client personal information, the client credit table stores the credit card number and the balance of the client, the products table stores the information of the products and their price and quantity, the order table stores the order date and total price of the purchase order process, the order products table stores the order ID and product ID.

VII. CONCLUSIONS

Through the design and implementation of the system, some conclusions are drawn, these are:

a) The system uses the modified version of Diffie-Hellman algorithm as Zero-Knowledge interactive protocol to perform the mutual authentication between the prover and the verifier which is an important issue as a security requirement, which protect the system from intruders and fake servers. With the addition to authentication, secure key generation and distribution has been performed by this algorithm as an additional function. Theoretically Man-In-the-Middle attack and Discrete Logarithm attack has been treated in the modified version of the protocol.

b) The integrity and confidentiality of the system which is another crucial issue to keep the data transmitted over unsecure channel unrevealed and unchanged by using a standard cryptographic suit which is used by standard security protocols such as AES, HMAC-SHA1.

c) The system used Java development environment instead of Microsoft.Net environment, because the java projects make it possible to implement the android applications and desktop applications besides it is open source software.

d) The system uses SMS in the implementation with message length 160 characters; this length is too small to be used in security and order process, Multimedia Messaging Service (MMS) that can be deal with huge data can be used instead.

e) In this research, GSM network has been used due to the availability of the service and extensive coverage which make it better than using WIFI network.

REFERENCES

[1] Md. Asif Hossain, et al., "A Proposal for Enhancing The Security System of Short Message Service in GSM", 2nd International Conference on Anti-counterfeiting, Security and Identification (ASID), Guiyang, 2008.

[2] Neetesh Saxena and Ashish Payal, "Enhancing Security System of Short Message Service for M-Commerce in GSM", International Journal of Computer Science & Engineering Technology (IJCSET), Vol. 2 No. 4, 2011.

[3] Narendra S. Chaudhari and Neetesh Saxena, "A Secure Digital Signature Approach for SMS Security", International Journal of Computer Applications (IJCA), Special issues on IP Multimedia Communications (1), pp. 98-102, October 2011.

[4] Ashish Ranjan, et al., "A Review of Secure SMS Based M-Commerce", International Journal of Engineering Sciences & Emerging Technologies (IJESET), Vol. 1, Issue 2, pp. 1-7, Feb 2012.

[5] Mahmut Özcan, "Design and Development of Practical and Secure E-Mail System", M.Sc. Thesis, Sabancı University, February 2003.

[6] Eric Maiwald," Fundamental security concepts", McGraw-Hill, 2004.

[7] Behrouz A. Forouzen,"Cryptography and network security", McGrew-Hill, Special Indian edition, 2008.

[8] Austin Mohr, "A Survey of Zero-Knowledge Proofs with Applications to Cryptography", April 2007.

[9] Mahmood Khalel Ibrahem, "Modification of Diffie–Hellman Key Exchange Algorithm for Zero Knowledge Proof", ICFCN' 12, conf. Baghdad, IEEE, 2012.

[10] Alfred J. Menezes, et al., "Applied Cryptography", first edition, CRC Press, October June 1996.