PGP Modification for Securing Digital Envelope Mail Using COM+ and Web Services

Tarek Salah Sobh and Mohamed Ibrahiem Amer
(Corresponding author: Tarek Salah Sobh)

Information Systems Department, Egyptian Armed Forces
110 Zhraa Nasr City, Flat 2 Stage 1, Cairo, Egypt (Email: tarekbox2000@yahoo.com)
(Received Dec. 26, 2009; revised and accepted Apr. 19 & June 7, 2010)

Abstract

Certified email is a value-added service for standard email systems, which guarantees the fairness, i.e., the intended recipient gets the mail content if and only if the mail originator receives a non-repudiation receipt showing that the message has been received by the recipient. Most of certified email protocols schemes have more or less weaknesses and/or security flaws. E-mail communication is insecure it can be read and modified as they are passed through the Internet as clear-text. In the worst case, fairness cannot be achieved since one dishonest party can mount some attacks to cheat the honest party such that the latter cannot get the expected items. In this paper, the proposed model is an attempt to standardize a protocol used to encrypt and digitally sign e-mail correspondence. Also, end-to-end security is discussed and a mechanism of key management is proposed if this service is requested by the user. We analyze some certified email protocols and propose some improvements in order to avoid security problems. Component object model and Web services are used to implement a prototype to the proposed model.

Keywords: Component object model, e-mail security protocols, encryption, PGP, Web services

1 Introduction

The lack of evidence for message receipt is a missing piece of the infrastructure required for the more professional use of email [18, 21, 24, 25, 29]. Certified email uses the notion of a digitally signed receipt and strengthens the binding between the evidence and the mail being certified. In other words, the main purpose of a certified email scheme is to achieve the fair exchange of a message and an irrefutable receipt in the sense that either the sender obtains the receipt from the receiver and the receiver accesses the content of the email simultaneously, or neither party gets the expected item. This undeniable but publicly verifiable receipt serves as an evidence for non-repudiation of receipt (NRR). Namely, if the receiver denies having received the delivered message from the sender, the sender can provide NRR evidence to an arbitrator to show that this claim is untrue. Some email schemes also provide evidences for non-repudiation of origin (NRO) [9, 13, 15]. Similarly, NRO evidence protects the receiver from the sender's dishonest denial that he/she did not deliver a particular message to this receiver, though this is the fact. We remark that certified email schemes supporting NRO service are functionally equivalent to non-repudiation protocols [4, 10, 11, 14, 16, 33].

Certified email has been discussed for years, and there are two major classes of schemes to address this technical problem: schemes that require the existence of a Trusted Third Party (TTP), and schemes that do not require the existence of a TTP [30]. Oppliger [24] showed clearly that the second class, i.e., either based on a simultaneous secret exchange or trusted system is inappropriate to provide certified mail services for the Internet. Specifically, there are two major disadvantages in the schemes based on a simultaneous secret exchange. One is that they are highly interactive between participants, and the other is that those schemes are based on an unrealistic assumption: both of participants have equal or very similar computing power. As for certified mail schemes based on trusted systems, their usefulness and security are often overestimated. Therefore, the use of TTPs seems advantageous and various types of TTPs can be considered according to their involvement in the certified email protocol: schemes with in-line TTPs [6], schemes with on-line TTPs [1, 24, 28, 33], and schemes with off-line TTPs [2, 5, 7, 9, 13, 15, 19, 34].

An improvement to reduce the TTP's involvement is the use of an on-line TTP. The on-line TTP is actively involved during each session of the certified email protocol but not at each step of the protocol. Its task may only deal with signaling information, such as cryptographic keys and/or receipts [24, 30]. In the academic literature, there is often an emphasis on reducing the role and the expense of a TTP. Protocols with a light-weight TTP have been proposed. For example, Abadi and Needham [1] proposed an efficient certified email scheme with a light on-line TTP. A key feature of their scheme is not to de-

ploy any public-key infrastructure; and further, Imamoto and Sakurai [13] revised their scheme in order to provide the non-repudiation of origin service.

A big step towards more efficient solutions was the introduction of off-line TTPs. That is, an off-line TTP involves in a protocol only in abnormal cases, e.g., a dishonest party is trying to cheat or the communication channel fails to work. There are two examples of cheating: a recipient does not issue a receipt though he/she has already obtained the message, or an originator deliberately refuses to reveal the full message to the recipient despite him/her indeed got a valid receipt. The TTP can help the victim to achieve a fair result in these abnormal cases. In other words, a fair certified email scheme with an off-line TTP guarantees that cheating is not actually beneficial to the cheater. Therefore, it is expected that such a protocol will be executed normally in most of the time. Based on this observation, the approach using an off-line TTP is also called optimistic [3].

In this work, the proposed model is an attempt to standardize a protocol used to encrypt and digitally sign e-mail correspondence. We analyze Pretty Good Privacy (PGP) as certified email protocols and propose some improvements to avoid security problems. We further give several informal but useful guidelines to design secure and standard certified email protocols. End-to-end security is discussed and a mechanism of key management is proposed if this service is requested by the user. Also, Component Object Model (COM) and Web services are used to implement a prototype to the proposed model.

This paper is organized as follows: Section 2 introduces traditional COM objects and XML Web services then describes e-mail security protocols and explores the details of PGP e-mail protocol. Section 3 produces the proposed model. Section 4 describes the proposed system working in operation. Finally Section 5 presents model conclusion.

2 Secure System of Web Services Included E-Mail

This section contains brief discussion of Web services, e-mail security protocols, and Pretty Good Privacy (PGP) one of the most popular example of e-mail security protocols.

2.1 COM and XML Web Services

In order to introduce platform independent and language independent applications that is reusable, extensible and portable. Component Object Model (COM) and XML Web services the best way to do so.

2.1.1 Traditional COM

In terms of software longevity, Microsoft's COM has enjoyed a lengthy and successful life. Formally solidified circa 1993, COM formalized a specific process for building reusable, binary software components.

When developers abide by the rules of COM, they are presented with a number of Desirable byproducts. One of the great byproducts of COM components is their Language-independent nature. This trait allows software developers to build COM Servers in one language (such as VB 6.0) and reuse them in any number of other COM-aware languages (such as C^{++}).

1) **Component Services**
Component Services (COM^{+}) is the next evolution of Microsoft COM and Microsoft Transaction Server (MTS). COM^{+} builds on and extends applications written using COM, MTS, and other COM-based technologies. COM^{+} handles many of the resource management tasks that developers previously had to program, such as thread allocation and security.

2) **Classes and Interfaces**
One of the central architectural foundations in COM programming is the separation of implementation (class) from protocol (interface). Simply put, an interface is a collection of semantically related methods that may be implemented by a given COM class (often called a co-class). Once a co-class has been instantiated by a particular client, the in-memory representation is termed a COM object.

   The odd thing about programmatic interfaces (as opposed to GUI interfaces) is the fact that the interfaces never define member variables, implementation logic, or other coding items that would mark them as a useful entity. Rather, the sole purpose of a programmatic interface is to specify the calling conventions a client must abide by to communicate with the implementing co-class. Once an interface has been defined (using the syntax of your favorite programming language), any number of COM classes may choose to support the specified interface. Given that an interface is a grouping of semantically related methods, it is common (and helpful) to regard an interface as a specific behavior that the class in question supports.

3) **Interface Definition Language**
Interface Definition Language (IDL) is the language used to define interface in simple standard formal definition classes can use this definition to build their implementation of the targeted interface. IDL is not considered a programming language rather than a semantic definition language.

2.1.2 XML Web Services

XML which stands for (eXtensible Markup Language) is not a really language but a framework for defining and using markup languages. Markup languages are used for creating units of information called XML document which have two standard representations: as linear text with markup and as tree data structure [31, 32].

No official definition of Web services in included in some standard or a W3C recommendations. However there is a general agreement that Web service is a distributed application that allows maximum interoperability between components written in different language and running on different platform. It is also generally agreed upon that kind of interoperability is achieved by encoding components in an XML protocol [31].

Web Services use agreed-upon, XML-based message format called Simple Object Access Protocol (SOAP). SOAP message may or may not be delivered over HTTP. FTP, SMTP and even proprietary messaging architecture are also a possibility. It provides a meta-description of its access point(s) and interfaces in an XML-based Web Services Description Language (WSDL) [32]. It is registered with one of several synchronized, online registries. Web Service vision and its classical architecture is described as follows:

1) publish step is the first and performed once. The Web service publishes its registry description (which contains, among other things, its WSDL description).

2) Find step a customer looking for a service of that type can find it in the registry.

3) Bind step the customer uses the found registry information to "Bind" itself to the service.

2.2 E-mail Security Protocols

Cryptographic algorithm is a mathematical function used for encryption or decryption? With most modern cryptography, the ability to keep encrypted information secret is based not on the cryptographic algorithm, which is widely known, but on a number called a key that must be used with the algorithm to produce an encrypted result or to decrypt previously encrypted information. Decryption with the correct key is simple. Decryption without the correct key is very difficult, and in some cases impossible for all practical purposes [4, 7, 23].

With most symmetric-key encryption algorithms, the same key is used for both encryption and decryption.

Public-key or asymmetric encryption involves a pair of keys–a public key and a private key–associated with an entity that needs to authenticate its identity electronically or to sign or encrypt data [17, 22, 33]. Each public key is published, and the corresponding private key is kept secret [12, 26]. Data encrypted with your public key can be decrypted only with your private key.

The most commonly used implementations of public-key encryption are based on algorithms patented by RSA Data Security.

Public Key Infrastructure (PKI) is a networked environment that contains the set of standards and services that facilitate the use of public key cryptography and X509 certificates [20, 28].

Certificate is an electronic document used to identify an individual, a server, a company, or some other entity and to associate that identity with a public key, public-key cryptography uses certificates to address the problem of impersonation [12, 20]. Certificates help prevent the use of fake public keys for impersonation. Only the public key certified by the certificate will work with the corresponding private key possessed by the entity identified by the certificate.

The certificate issued by the Certificate Authorities (CA) binds a particular public key to the name of the entity the certificate identifies.

Therefore, basic needs have emerged:

1) **Confidentiality.** The e-mail can only be read by the intended recipient. This is ensured using encryption.

2) **Authentication.** The e-mail has been written by particular person and has not been altered on its way over the Internet. This can be accomplished using digital signatures.

3) **Nonrepudiation.** Prevent either sender or receiver from denying transmitted e-mail thus when an e-mail is sent, the receiver can prove that the alleged sender in fact sent the message. Similarly, when an e-mail is received, the sender can prove that the alleged receiver in fact received the message; this can be accomplished by using PKI.

There are two ways of encoding a signed mail:

1) **Clear-signed.** A clear-signed message contains the original message as clear text. This enables non-S/MIME-compatible mail reader software to read the contents of the message.

2) **Opaque-signed.** Using opaque-signing, the original message is not included as clear text but base64-encoded. This eliminates the risk of the original text being changed while being transmitted (e.g. through automatic conversion performed by some mail transfer agent on the way). If the text was changed even slightly, the message digest would be different, thereby invalidating the digital signature. However, opaque-signed messages have the drawback that they cannot be read by non-S/MIME-compatible mail readers.

2.3 Pretty Good Privacy

Pretty Good Privacy (PGP) is an open source software package for e-mail security. It provides authentication through the use of digital signature; confidentiality through the use of symmetric block encryption; compression using ZIP algorithm; e-mail compatibility using Rdix-64 encoding scheme, and segmentation and reassembly to accommodate long e-mail. PGP incorporates tools for developing a public-trust model and public-key certificate management [8, 17]. In this work we agree with the idea about the PGP is not so good for secure because key exchange protocol is insecure. But still PGP one of

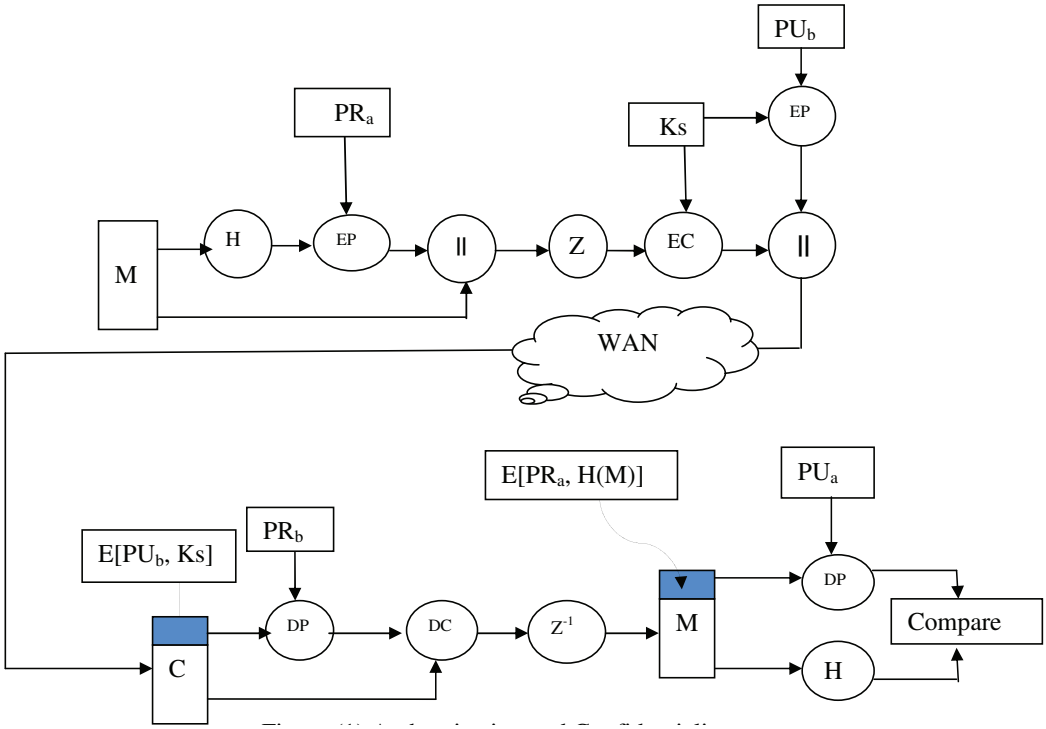

Figure 1: Authentication and confidentiality

the most popular e-mail security and a lot of organizations depend on PGP to secure their e-mail [12, 22]. This is why we move toward trying to enhance these security lakes.

Figure 1 presents authentication and confidentiality in PGP and contains the following notations:

- M = Plain Message;
- C = Ciphered Message;
- Ks = session key used in symmetric encryption scheme;
- PR_a = Private Key of user A, used in public key encryption;
- PU_a = Public key of user A, used in public key encryption;
- PU_b = Public key of user B, used in public key encryption;
- EP = Public key Encryption process;
- DP = Public key decryption process;
- EC = Symmetric encryption process;
- DC = Symmetric decryption process;
- H = hash function;
- $||$ = Concatenation;
- Z= Compression using ZIP algorithm;
- $R64$ =conversion to radix 64 format.

The block diagrams in Figure 1 describe the operation of a typical standard PGP protocol. In the Next graph we describe the location of our work in the original PGP protocol.

3 Proposed System Model

Figure 2 is a general block diagram of our proposed system model. Both Figures 3 and 4 represent the detailed diagram for each stage in proposed modification. The highlighted gray regions describe the location of the modifications, which are 7 Major Modification that can be classified into three major groups.

1) **Algorithm Obtaining Mechanisms (AOM).** This group is responsible of obtaining Symmetric algorithm and asymmetric algorithm b; d modifications are presented and belong to this group. As known securing data uses many algorithms with a common security rules like encryption, decryption, hashing and permutations, for each one of them there are a many classifications and algorithms. In the proposed modifications to PGP system we not only use one algorithm for symmetric and asymmetric algorithms but more than one algorithm. AOM method is used for obtaining such algorithms and creates hundreds of combinations from predefined symmetric and asymmetric algorithms libraries and time by time increases in the system.

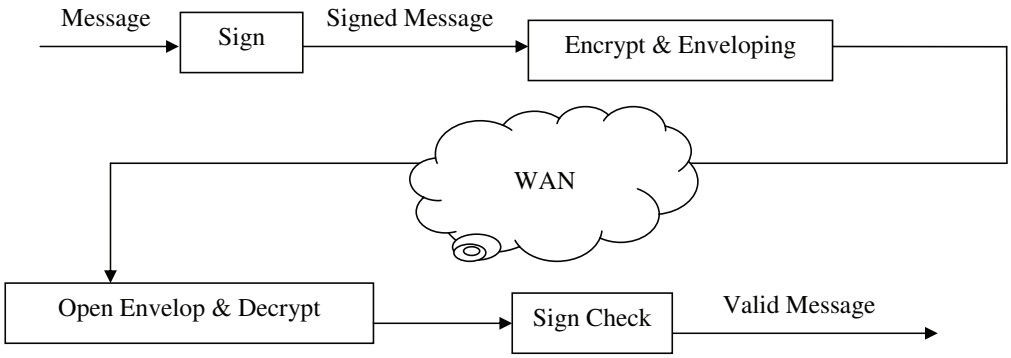

Figure 2: Block diagram for proposed system

2) **Keys Obtaining Mechanisms (KOM).** Modifications a, c, e belongs to this group done through obtaining the following encryption keys symmetric key Ks, public key PU and/or private key PR. For all algorithms obtained in AOM there is a need for their keys. KOM mechanism is responsible to get these keys either by using XML Web services or from stored keys on e-tokens.

3) **Data Permutation Mechanisms (DPM).** Modifications f, g belongs to this group done through what we call it shuffling and deshuffling algorithms.

The proposed modification splits the operation of PGP into four main stages: A) Signing, B) Encryption and enveloping, C) Open envelop and decrypt and finally D) Sign check.

1) **Signing Stage.** In This stage the message is signed using the private key of the sender obtained by modification a by the asymmetric algorithm obtained by modification b, the result is then compressed using a compression algorithm.

2) **Encryption & Enveloping Stage.** In this stage the message after signing is subject to symmetric encryption with a key obtained by modification c for the symmetric encryption algorithm obtained by modification d, Then either the symmetric key used in encrypting the message (if the mode was key exchange) or its seeds (if the mode was key deduction) is encrypted by the asymmetric algorithm using the public key obtained by modification e and concatenated to the message then the whole mixture is shuffled in modification f.

   At the receivers side the message is subject to the reverse operations that are applied to at sender side.

3) **Open Envelop and Decrypt.** The receiver after receiving the message extracts the shuffling seed used in shuffling the message to deshuffle the message (modification g), Extract the session key, decrypt it using his private key use this key to decrypt the message.

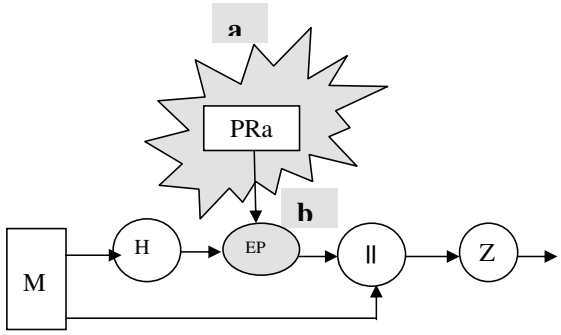

Figure 3: Detailed description of the signing phase

4) **Sign Check Stage.** After decrypting the message the user decompress the message, split it to obtain the plain text and the hash digest and then use his private key to sign it and compare his sign with the hash received if the result was equivalent then the message is authenticated and no change was done from the path from sender from receiver.

   In modification groups (AOM, KOM) the selection may be selected manually by the sender, forced by system administrator or purely random selected by the system itself. The selection process based on a XML Request/Response pairs to a Certificate Authority Database (CADB) that holds the public keys of the users. Figure 7 introduce A-XML Initialization vector Request (XMLIV).

   According to Figure 7 XMLIV composed of the next fields:

   - **S_ID (Sender ID):** 64-bit field containing senders ID in the CADB. The CA uses this field to determine the selection (Manually, forced by system administrator, randomly) Method according to his policy.
   - **R_ID (Receiver ID):** 64-bit field containing receivers ID in the CADB. The CA uses

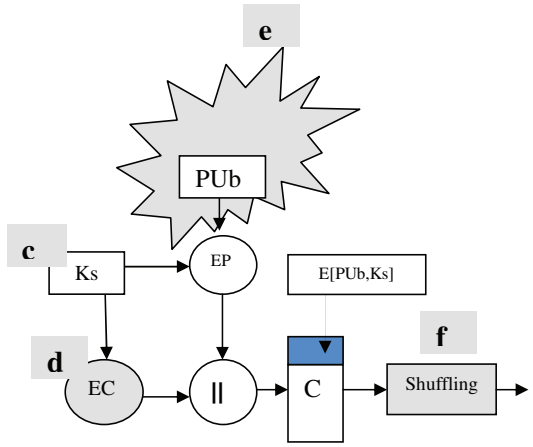

Figure 4: Detailed description of encryption and enveloping stage

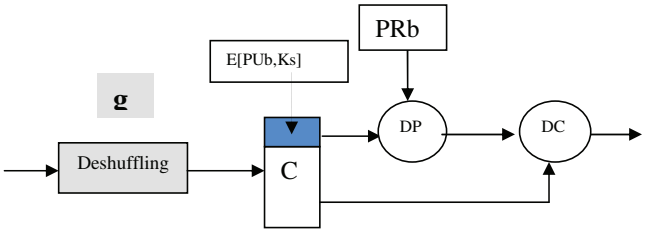

Figure 5: Detailed description of open envelop and decrypt stage

this field to retrieve the public key of receiver from the public keys database according to the selected asymmetric algorithm obtained from AS_ID as described later.

- **SY_ID:** 16-bit Symmetric Encryption Algorithm ID suggested by Sender if the selection method was manually the CA approve the selection, if the selection was forced or random the CA completely ignore this field and respond with the appropriate Symmetric Encryption algorithm ID.
- **AS_ID:** 15-bit Asymmetric Encryption Algorithm ID suggested by Sender if the selection method was manually the CA Approve the selection, if the selection was forced or random the CA completely ignore this field and respond with the appropriate Asymmetric Encryption algorithm ID.
- **SDF:** 1-bit Session-key Delivery Flag value 0 means Key-Exchange value 1 means Key-Deduction(described later). Its value is suggested by Sender if the selection method was manually the CA approve the selection, if the selection was forced or random the CA completely ignore this field and respond with the appropriate Session-key Delivery Flag.

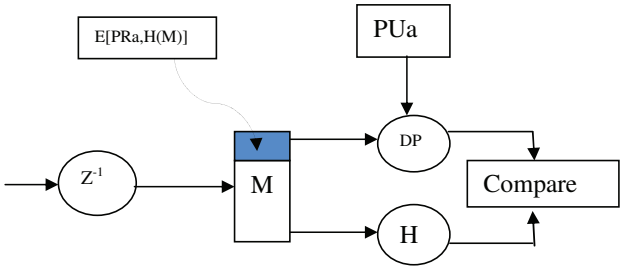

Figure 6: Detailed description of open envelop and decrypt stage

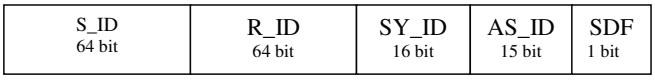

Figure 7: 20-byte XML initialization vector request (XMLIV)

According to Figure 8 XML Response Vector (XMLRV) is composed of SY_ID, AS_ID and SDF fields. These fields are the same fields described in XMLIV but the values of these fields are determined by CA itself either the original values suggested by the sender in XMLIV (if the selection method was manually) or the forced values by system administrator or totally random. R_PU is the receiver's public key for the selected public key algorithm in CADB.

3.1 Algorithm Obtaining Mechanisms (AOM)

Figure 9 presents generic COM$^+$ architecture of the proposed system. The core of this architecture is a system engine which is allocated inside e-mail server framework software. A COM$^+$ objects are used to encapsulate an implementation of both symmetric and asymmetric encryption processes based on a standard interfaces defined to standardize the internal method signatures (name, parameters and return types). These interfaces are defined using Interface Definition Language (IDL) which is independent of platform or implementation language.

For Symmetric, Asymmetric algorithms when a user receive XMLRV containing an SY_ID or AS_ID, or when receive an email containing algorithms that are not registered as a COM$^+$ objects in his machine he sends a special http: request (containing the required algorithm) for CA to download the appropriate .dll or .sys files with the implementation of the SY_ID, AS_ID then he register them in his system, create

SY_ID	AS_ID	SDF	R_PU
16 bit	15 bit	1 bit	4096bit

Figure 8: 516-byte XML response vector (XMLRV)

object and finally use them to encrypt, decrypt authenticate the message. So the user can start his system with no algorithm installed and gradually as send, receive mails the COM+ libraries starts to grow in his machine.

3.2 Keys Obtaining Mechanisms (KOM)

In all proposed mechanisms ID is used to obtain the keys that are involved in the generation of the transmitted message via non-secure channels, these keys are heavily dependent on the selected Algorithm in AOM.

3.2.1 Public Key Obtaining Modification

XML Web Service hosted at the certificate authority that is responsible for Generation of both Public and private key pairs. The private key as mentioned is saved to key container while the public is inserted into public keys database and accessed via XML Web Service the sender uses the XML Web Service to obtain the destinations public key used to encapsulate the session key and authentication data as mentioned before throw XMLRV.

3.2.2 Symmetric Key Obtaining Modification

Key deduction and key exchange are proposed to obtain the session key Ks. In both of two methods the key is randomly generated by the sender according to the selected algorithm in AOM but the difference between the two methods is: in key exchange the key itself is delivered with the message the receiver has no need to regenerate or deduce the key, but in key deduction the random seeds used in key generation process according to the selected algorithm in AOM at sender site are delivered with the message, the receiver uses these random seeds to regenerate the session key for the symmetric decryption process (DC).

3.2.3 Modifications a: Private Key Obtaining

E-Tokens and Smart Cards or any physical key container used to save the private key. These private keys must be originally generated and saved to Tokens or Smart Cards by a trusted certificate authority (CA) that hosts the XML Web Service -described next- to distribute the public keys to all network nodes.

3.3 Data Permutation Mechanisms

PGP has a fixed message format with a fixed structure as shown in Figure 10. This structure is easily to be used to discover security parts like encrypted session key that attacker can use any known attack methods to break it and reveal the message. Especially with the vast increase in computing capabilities and super computers attacker is capable to reduce the time used to analysis the security parameters. Here, we introduce Data Permutation Mechanisms (DPM) called Shuffling and Deshuffling used to scramble the message and destroy any known fixed structure of the message. DPM is the final step in the process of preparing the email envelope before sending the email in the none-secured channel. In this process the email message -after encryption- is subject to a permutation that scramble the order of its content based on a random seed generated by sender and attached encrypted (by receivers public key) to the message so the original PGP header is not displayed with the message, just only the encrypted seed of the shuffling process that also can be hidden inside the message, or even left in a known position.

Both Figure 11 and Figure 12 describe the flow chart of the Shuffling and Deshuffling process respectively.

In this section, we will show that our proposed model is also an efficient one. The proposed system was implemented using RSA algorithm. The time taken for each ATPKSS operation is 47 m sec using 1024-bit key length for authentication or encryption. In order to apply the proposed system in a real networks both mobile stations and network entities must change to include digital certificates and a public key algorithm. Therefore, we believe our proposed model is really a simple and efficient three-party key exchange protocol, and can be applied in practice.

PGP introduce authentication and confidentiality to the message data, PGP headers and addresses of delivered messages are added later at a higher layer of application programs and are out of the PGP processing. So, the shuffling modification does really obscure the location of PGP headers because the delivery of the message from sender to receiver doesn't rely on theses header.

As we mentioned above in the proposed model, we use AOM, KOM and DPM, the way we use these mechanisms shortens the drawbacks of key exchange. Changing the algorithm (AOM), changing the key (KOM) and permutation of data (DPM) minimize the chance for any adversary to obtain the key, even he/she (during exchange process) obtained the message he still need to rearrange the message (function DPM), If so he/she still need to know where and what is the key(s) inside the rearranged message (function KOM), and even if so he/she still need to know the combination of algorithms used in securing the message (function of AOM) and what is the three algorithms.

3.4 Modification Complexity

Recalling to our modifications that are in the three major groups (AOM, KOM and DPM) the first two groups are only XML requests and responses that from the client

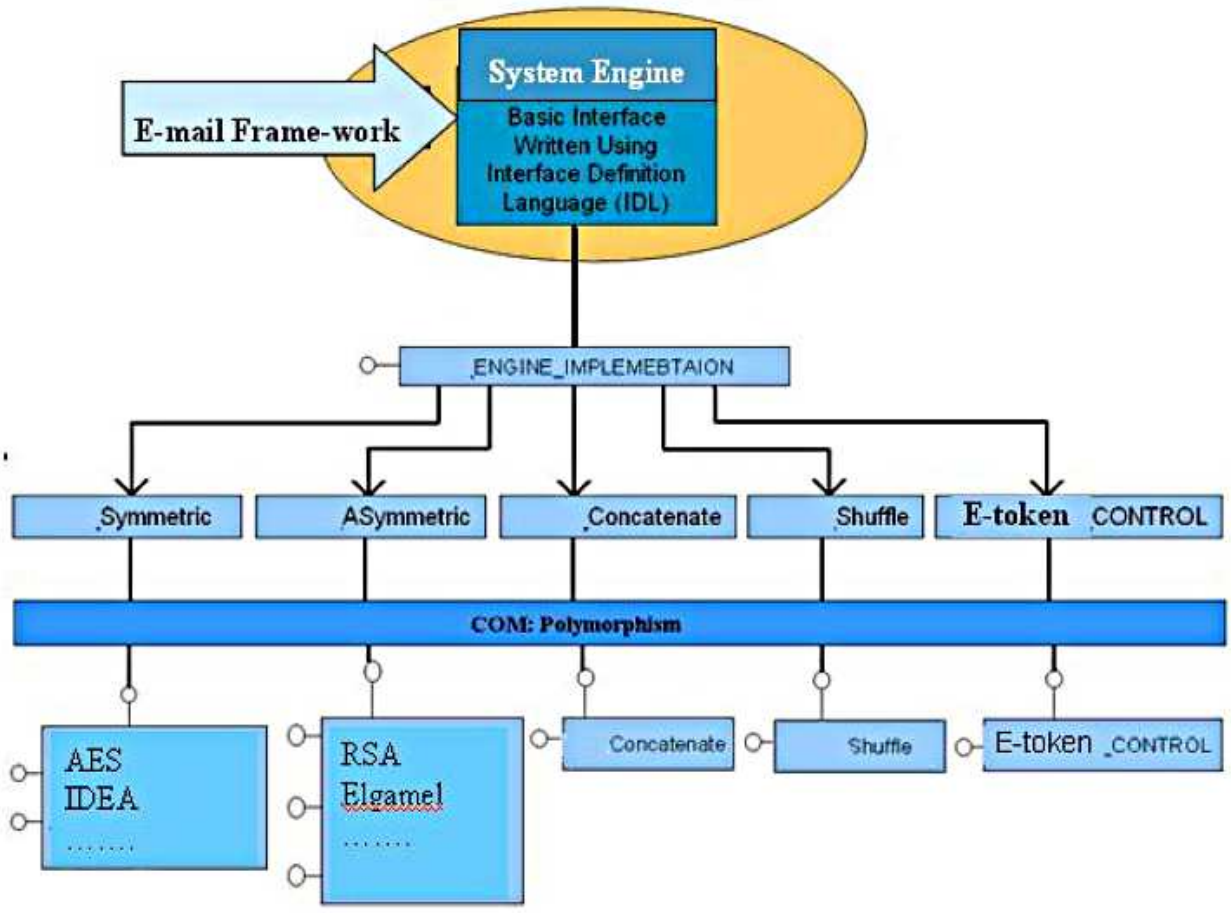

Figure 9: COM$^+$ architecture in the proposed system model

Session key Component	Signature	Encrypted Message

Figure 10: Common format for PGP message

(sender) side complexity point of view is only Big O(1), and for the DPM either shuffling or deshuffling processes the complexity is Big O(n) that n is the size of the message so grouping the complexity O(1)+O(n). The result of the complexity is Big O(n) as additional complexity of the original PGP complexity.

4 Working System

This section describes simple example to the entire, working system in operation. Assuming sender A sends a message to receiver B. The next steps are used to exchange the message using the proposed system.

At Sender *A*:

1) Sender *A* use XML Web service Request/Response Vectors to obtain both symmetric and asymmetric algorithms through AOM (modifications b, d) and their keys by KOM (modifications c, e).
2) Sender *A* signs the message using his private key obtained from e-token by (modification a) and concatenate it to the message.
3) Sender *A* secures the message (that contains the signature of sender) by symmetric encryption using the session key (symmetric key) obtained in Step 1 (algorithm and its key).
4) Sender *A* encrypts the session key using B's private key/algorithm obtained at Step 1.
5) Sender *A* concatenate encrypted session key + encrypted message + XMLRV together to generate the secured message.
6) Sender *A* shuffles the whole mixture (modification f) and hides the encryption seed inside the resulted shuffled message.
7) Sender *A* sends the message to receiver *B* on the WAN.

At Receiver *B*:

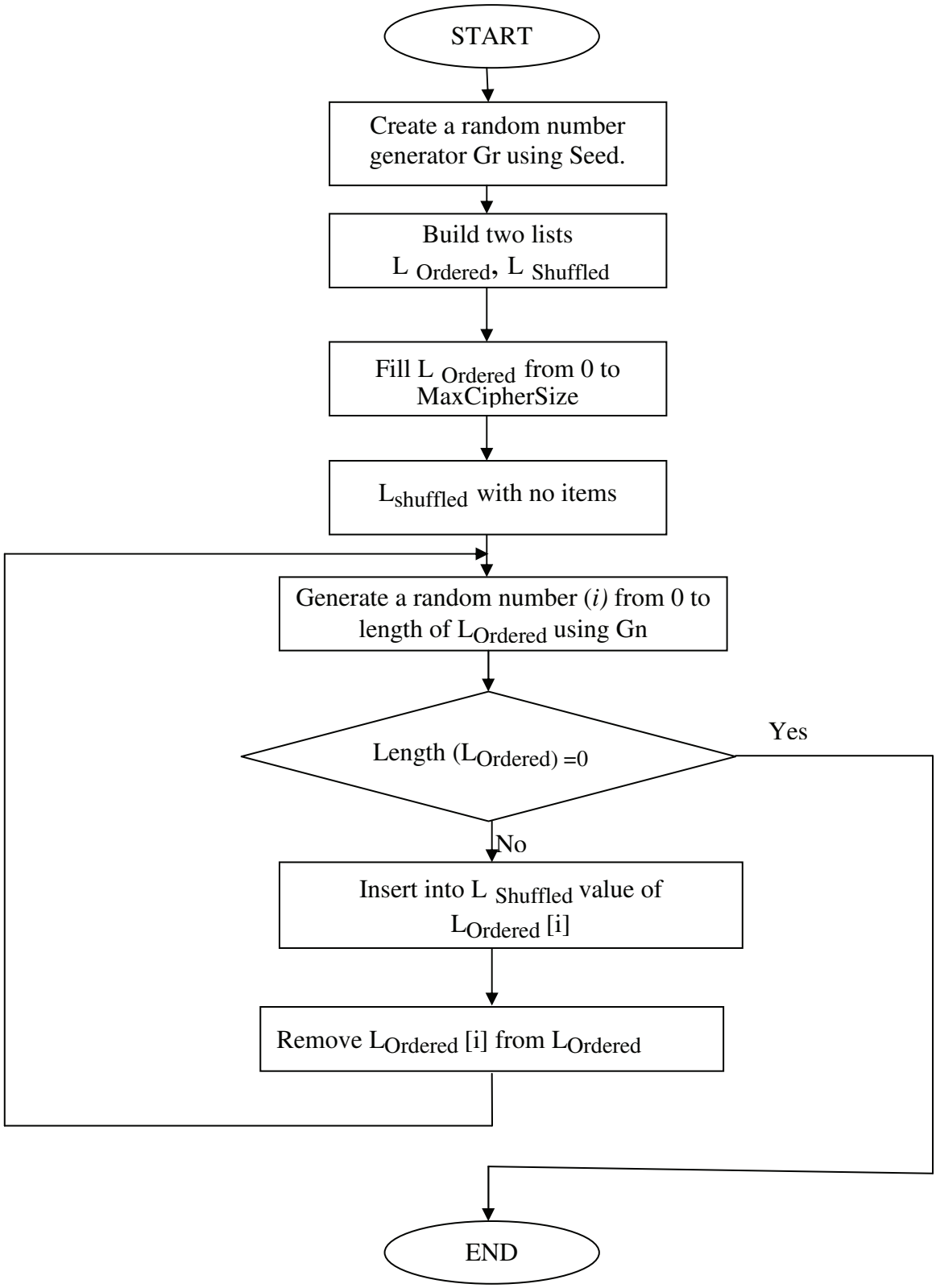

Figure 11: Shuffling algorithm

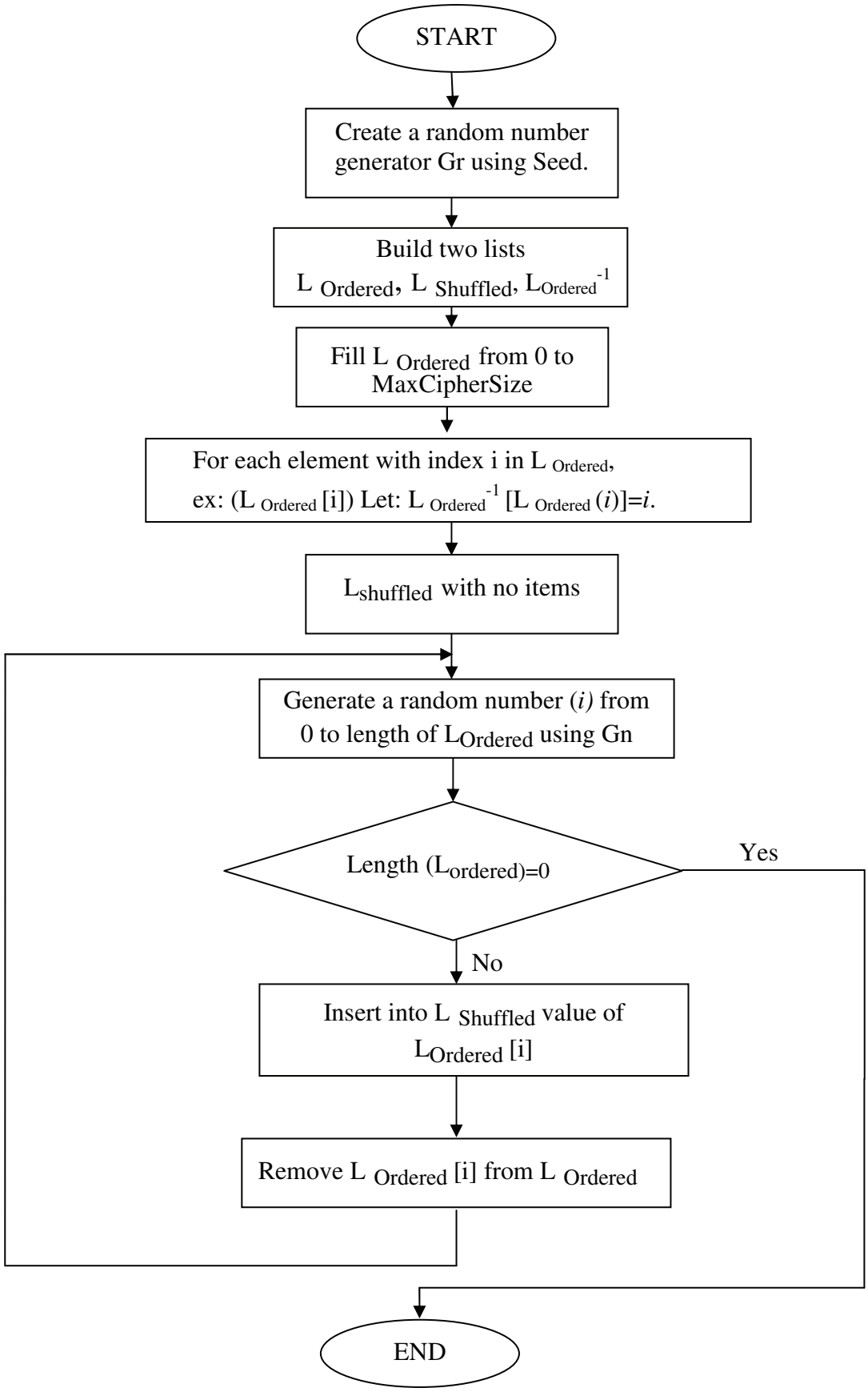

Figure 12: Deshuffling algorithm

1) Receiver B extracts the shuffling seed and deshuffles the message.

2) Receiver B splits the message to extract the three individual parts of the message (encrypted session key, encrypted message, XMLRV).

3) Receiver B uses extracted XMLRV to obtain the algorithms used in securing the message at sender's side.

4) Receiver B uses his private key to decrypt the session key used to encrypt the message.

5) Receiver B uses the session key obtained in step 4 to decrypt the message.

6) Receiver B generates the message, decrypt the sender A signature using A's public key and compares it with the received message to check sender A's Signature.

There are a few optional security service extensions:

1) Security labels attributes included in a signed message specifies the security classification of the signed content. They can be based on the X.411 recommendation (unmarked, unclassified, restricted, confidential, secret, and top-secret) or on an organization's own security policy. Based on this information, the mail agent can decide whether or not to show the contents of the message to the user.

2) Mail list management when sending encrypted messages to a large number of recipients, the mail client would have to encrypt the session key using every recipient's public key. With the mail list management extension, the user would only encrypt the message once, using the mailing list agent's public key. The mailing list agent would then encrypt and forward the message to every intended recipient.

3) Mail list management also includes expansion attributes to prevent mail loops within mailing list.

5 Conclusion

Public-key based protocols, with a mature PKI, will enable all the involved entities such as users, network operators and service providers to have full range of state-of the-art security features including non-repudiation services, mutual authentication and anonymity [26]. This system uses public key technique without overloading the network. It is a hybrid encryption system; where in normal condition the original algorithms and protocols work, only when a flaw occurs, the solution is triggered.

To prove the effectiveness of the proposed system, we have implemented it using RSA as public key algorithm; then measured the time taken for each operation. It is found that the time taken to apply the ATPKSS is very small (with maximum of 47 msec); which is a very few price to pay in terms of ensuring security. In order to apply our system both mobile stations and network entities must change to include digital certificates and a public key algorithm.

The software implementation the RSA algorithm can be mounted in both mobile stations and Network HLR/VLR. The computations will be faster if the algorithm is implemented as a hardware chip. In this case, it will require changing the mobile equipment to include it. And due to the evolution in mobile equipment industry, it is possible to do that. Also we presented a new end-to-end approach where a key agreement phase takes place between the network and the two users first. Then in the second phase the initiating user sends a built-in phone sub-key.

The original built in phone key is the root of the sessions' sub-keys. Each time a new session is opened, a function is applied to the root key to obtain a sub-key. The data is transferred between the two users afterwards encrypted by this key only or a combination key between the phone key and the network session key. This will address the problem of attacking the Service provider itself. Finally, this work especially selects XML Web services that it can work in any heterogeneous environments and it is a platform independent and language independent. All these features make this work scalable to any kinds of networks such as Internet.

For further research; we suggest testing the additional asymmetric encryption algorithms and see whether the time measured is decreased or not.

References

[1] M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," *IEEE Transactions on Software Engineering,* vol. 22, no. 1, pp. 6-15, 1996.

[2] N. Asokan, M. Schunter, and M. Waidner, "Optimistic protocols for fair exchange," *Proceedings of AMC Conference on Computer and Communications Security (CCS'97)*, pp. 7-17, ACM Press, New York, 1997.

[3] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures," *IEEE Journal of Selected Areas Communications*, vol. 18, no. 4, pp. 591-606, 2000.

[4] G. Ateniese, C. N. Rotaru, "Stateless-recipient certified email system based on verifiable encryption," *Proceedings of CT-RSA'02*, LNCS 2271, pp. 182-199, Springer-Verlag, 2002.

[5] G. Ateniese, B. d. Medeiros, and M. T. Goodrich, "TRICERT: A distributed certified email scheme," *Proceedings of Symposium on Network and Distributed Systems Security (NDSS'01)*, 2001.

[6] A. Bahreman and J. D. Tygar, "Certified electronic mail," *Proceedings of Symposium on Network and Distributed System Security (NDSS'94)*, pp. 3-19, 1994.

[7] F. Bao, R. H. Deng, and W. Mao, "Efficient and practical fair exchange protocols with off-line TTP," *Proceedings of IEEE Symposium on Security and Privacy*, pp. 77-85, Los Alamitos, 1998.

[8] E. E. Schultz, "Research on usability in information security," *Computer Fraud & Security*, vol. 2007, no. 6, pp. 8-10, June 2007.

[9] J. L. F. Gomila, M. P. Capella, and L. H. Rotger, "An efficient protocol for certified electronic mail," *Proceedings of Information Security Workshop (ISW'00)*, LNCS 1975, pp. 237-248, Springer-Verlag, Berlin, 2000.

[10] C. Galdi and R. Giordano, "Certified email with temporal authentication: An improved optimistic protocol," *Proceedings of International Conference on Trust and Privacy in Digital Business (TrustBus'04)*, LNCS 3184, pp. 181-190, Springer-Verlag, Berlin, 2004.

[11] S. Gurgens, C. Rudolph, and H. Vogt, "On the security of fair nonrepudiation protocols," *Proceedings of Information Security Conference (ISC'03)*, LNCS 2851, pp. 193V207, Springer-Verlag, Berlin, 2003.

[12] R. Hunt, "Technological infrastructure for PKI and digital certification," *Computer Communications*, vol. 24, no. 14, pp. 1460-1471, Sep. 15, 2001.

[13] K. Imamoto and K. Sakurai, "A certified email system with receiver's selective usage of delivery authority," *Proceedings of Indocrypt'02*, LNCS 2551, pp. 326-338, Springer-Verlag, Berlin, 2002.

[14] S. M. Kaback, "A patent searcher looks at the American Inventors Protection Act," *World Patent Information*, vol. 24, no. 4, pp. 263-268, Dec. 2002.

[15] S. Kremer, O. Markowitch, "Selective receipt in certified email," *Proceedings of Indocrypt'01*, LNCS 2247, pp. 136-148, Springer-Verlag, Berlin, 2001.

[16] S. Kremer, O. Markowitch, and J. Zhou, "An intensive survey of fair nonrepudiation protocol," *Computer Communications*, vol. 25, no. 17, pp. 1606-1621, 2002.

[17] J. O. Kwon, I. R. Jeong, and D. H. Lee, "A forward-secure e-mail protocol without certificated public keys," *Information Sciences*, vol. 179, no. 24, pp. 4227-4231, 2009.

[18] P. Machanick, "A distributed systems approach to secure Internet mail," *Computers & Security*, vol. 24, no. 6, pp. 492-499, 2005.

[19] S. Micali, "Simple and fast optimistic protocols for fair electronic exchange," *Proceedings of 22nd Annual ACM Symp. on Principles of Distributed Computing (PODC'03)*, pp. 12-19, New York, 2003.

[20] R. E. Nemr, I. A. S. Ismail, and S. H. Ahmed, "Action-triggered public-Key cryptography for GSM systems with phone dependent end-to-end encryption," *ICGST-CNIR Journal*, vol. 5, no. 2, pp. 64-70, June 2006. (http://www. Icgst.com)

[21] News, "Over two-thirds of UK companies have suffered security breaches, but response is patchy," *Computer Fraud & Security*, vol. 2009, no. 7, pp. 1-2, July 2009.

[22] M. Omar, Y. Challal, and A. Bouabdallah, "Reliable and fully distributed trust model for mobile ad hoc networks," *Computers & Security*, vol. 28, no. 3-4, pp. 199-214, May-June 2009.

[23] J. A. Onieva, J. Zhou, and J. Lopez, "Enhancing certified email service for timeliness and multicast," *Proceedings of 4th International Network Conference (INC'04)*, pp. 327-336, Plymouth, UK, 2004.

[24] R. Oppliger, "Certified mail: The next challenge for secure messaging," *Communications of the ACM*, vol. 47, no. 8, pp. 75-79, 2004.

[25] M. H. Shao, G. Wang, and J. Zhou, "Some common attacks against certified email protocols and the countermeasures," *Computer Communications*, vol. 29, pp. 2759-2769, 2006.

[26] K. Shim and Y. R. Lee, "Security flaws in authentication and key establishment protocols for mobile communications," *Applied Mathematics and Computation*, vol. 169, no. 1, pp. 62-74, Oct. 2005.

[27] W. Stallings, *Cryptography and Network Security Principals and practices*, Fourth Edition.

[28] B. Schneier and J. Riordan, "A certified email protocol," *Proceedings of Annual Computer Security Applications Conference (ACSAC'97)*, pp. 232-238, Los Alamitos, 1997.

[29] W. Technologies, *Email Monitoring in the Workplace a Simple Guide to Employers*, Robert Muckle, July 2003.

[30] X. Gu, J. Yangc, J. Lana, and Z. Caod, "Huffman-based join-exit-tree scheme for contributory key management," *Computers & Security*, vol. 2, no. 8, pp. 29-39, 2009.

[31] Y. Zhuang, X. Ji, and L. Wang, "Investigation on Implementation Method of Web Services Security Proxy," *GESTS International Transactions on Computer Science and Engineering*, vol. 19, no. 1, pp. 172-182, 2005.

[32] C. Zhou, L. T. Chia, and B. S. Lee, "DAML-QoS ontology for Web services," *Proceedings of IEEE International Conference on Web Services*, pp. 472-479, 2004.

[33] J. Zhou and D. Gollmann, "A fair non-repudiation protocol," *Proceedings of IEEE Symposium on Security and Privacy*, pp. 55-61, Los Alamitos, 1996.

[34] J. Zhou and D. Gollmann, "An efficient non-repudiation protocol," *Proceedings of 10th Computer Security Foundations Workshop (CSFW'97)*, pp. 126-132, Los Alamitos, 1997, 2005.

Tarek Salah Sobh received his B.Sc. degree in computer engineering from Military Technical College, Cairo, Egypt in 1987. Both M.Sc. and Ph.D. degrees from Computer and System Engineering Department, Faculty of Engineering, Al-Azhar University, Cairo, Egypt. He has managed, designed and developed several packages for business applications and security systems. He has authored/co-authored of many refereed journal/conference papers and booklet. Some of the articles are available in the ScienceDirect Top 25 hottest

articles. His research of interest includes distributed systems, knowledge discovery, database system design and development, data mining, information fusion, software engineering, intelligent systems, networks and computer security, and network management.

Mohamed Ibrahiem Amer received his B.Sc. degree in electrical engineering from Military Technical College, Cairo, Egypt in 2002. He received his M.Sc. degree in computer engineering from Military Technical College, Cairo, Egypt in 2009. He has designed and developed several packages for business applications and security systems. His research of interest includes database system design and development, data mining, software engineering, networks and computer security, and FPGA.