Eng. & Technology, Vol.25, No.4, 2007

Secure E-Mail System Using S/Mime and Ib-Pkc

Dr. Sufyan T. Faraj* **Mohammed T. Ibrahem****

Received on: 4/4/2006
Accepted on: 26/11/2006

Abstract

Although e-mail security solutions have been introduced for more than two decades, most of the e-mail messages are sent nowadays without being secured by any of these techniques. This is due to the complexity of using these secure e-mail systems and protocols. The complexity mainly arises from the difficulty associated with managing certificates and public keys. The main objective of this study was to find a solution that can make secure e-mail systems easier to use while maintaining the same level of security. This paper proposes a secure e-mail system that is based on the S/MIME standard where the public key and signature algorithms have been replaced by their Identity-Based Cryptography analogue algorithms.

Using Identity-Based Cryptography has eliminated the need for digital certificates, and provided a solution to the usability problem present in the existing secure e-mail systems. Users can determine the public key of the recipient without having to contact any trusted third party, and can start encrypting or verifying messages as long as they have the public system parameters that can be publicly available. Users need to contact the Private Key Generator (PKG) only once in order to retrieve their private key before being able to decrypt or sign messages.

Keywords: Secure E-mail, S/MIME, Identity-Based Cryptography, IBE, IBS, Security, Usability.

الخلاصة

على الرغم من مرور فترة طويلة نسبيا على إنشاء أنظمة البريد الالكتروني الآمن، إلا أن معظم الرسائل يتم إرسالها بدون تأمينها بواسطة أي من هذه التقنيات. و يعود سبب عدم استخدام أي من هذه التقنيات إلى كون استخدامها يتسم بالتعقيد بسبب صعوبة التعامل مع الشهادات الرقمية و المفاتيح العامة. الهدف الرئيسي لهذا البحث هو إيجاد وسيلة لتسهيل التعامل مع أنظمة البريد الالكتروني الآمن مع الحفاظ على نفس مستوى الأمن. حيث تم تصميم نظام بريد الكتروني آمن يعتمد على ملحقات بريد الإنترنت الآمن متعدد الأغراض (S/MIME), و ذلك بعد أن تم إدخال خوارزميات التشفير و التوقيع الرقمي المعتمدة على الهوية (IB-PKC) بدل الخوارزميات المعتمدة في الأنظمة الحالية. لقد تم تحقيق أهداف البحث عن طريق النظام المقترح، فمع استخدام التشفير المعتمد على الهوية (IB-PKC) انتفت الحاجة لاستعمال الشهادات الرقمية، و وفر النظام المقترح حلا لمشكلة صعوبة استخدام أنظمة البريد الالكتروني الآمن. حيث أن مستخدمي النظام بإمكانهم تحديد المفتاح العام للمستلم بدون الحاجة للاتصال بأي طرف ثالث موثوق (TTP). كما أن بإمكان المستخدمين البدء بتشفير الرسائل و التحقق من تواقيع الرسائل على شرط أن يحصلوا على معلومات النظام العامة و

* Dean of College of IT, Nahrain Univ., Baghdad-IRAQ.
** Dept.of Computer Eng., College of Eng., Univ. of Baghdad, Baghdad-IRAQ.

التي تكون متاحة لتحميلها من قبل أي شخص. و لا يحتاج مستخدمو النظام إلى الاتصال بمولد المفاتيح الخاصة (PKG) إلا مرة واحدة فقط للحصول على مفتاحهم الخاص ليتمكنوا من فــك تشفير الرسائل الموجهة لهم و إنشاء رسائل موقعة رقميا.

1. Introduction

E-mail is one of the most popular activities on the Internet. The ease of communicating over e-mail makes it the optimum communication medium for many people and businesses. In order for e-mail to be used in important communications, secure e-mail systems were developed. E-mail Security solutions such as PEM [1], PGP [2] and S/MIME [3] have been in place for more than a decade, and have been implemented by most e-mail clients. However, most e-mail messages are sent without being secured by any of the available solutions. This is mainly because of the complexity of using these secure e-mail systems and protocols.

In 1999, a study [4] was made to evaluate the PGP usability, and it proved scientifically that e-mail encryption is too hard. Twelve test participants were involved in that study, and were asked to create keys and to send digitally signed and encrypted messages. Only one third of them were able to use PGP to correctly sign and encrypt an e-mail message. Furthermore, one quarter of them accidentally exposed the secret they were meant to protect in the process, by sending it in e-mail they thought they had encrypted but had not. But while the usability failings found in PGP 5.0 can certainly explain the failure of PGP 5.0 in the marketplace, these failings can't explain the similar failure of every other secure messaging system that implements public key cryptography [5]. Such a failure can be explained by a common usability failure in the underlying certification model used by these systems [5].

The PGP key certification model is of a user-centric type that allows individuals to create their own public/private key pairs and then optionally have those public keys certified by one or more individuals, but this comes at the cost of added responsibility on the part of end-user. On the other hand, S/MIME was constructed with an eye toward integration with the Public Key Infrastructure (PKI). As such, S/MIME allows the use of certificates for both signing and encrypting e-mail, along with the integration with Certificate Revocation Lists (CRL) and cryptographically signed return receipts. Current PKI have several aspects that attracted criticism and controversy, some of which are: Difficulty in retrieving keys and certificates, certificate processing complexity, costly certificates, and naming semantics problem [6]. Several solutions have been made in order to reduce this complexity; some of them were in the form of commercial security appliances that manages

the key processing for the users, while the others were academic solutions like “Safe Staging for Computer Security” [7] and “Enabling E-mail Confidentiality through the use of Opportunistic Encryption” [8].

In Identity-Based Public Key Cryptography (IB-PKC), the public keys are constructed from users' publicly available identities that are uniquely associated with them (like their e-mail addresses). The motivation for such a scheme was to simplify key management and eliminate the use of certificates. Since the introduction of the first successful Identity-Based Encryption (IBE) scheme, replacing the current PKI services with their equivalent IB-PKC techniques has become a very popular research area. This ranges from making some kind of combination between the PKI services and the IB-PKC algorithms to totally replacing some PKI services with their IB-PKC alternatives. Several studies have been made on the subject of securing e-mails with the IB-PKC principles. In 2002, a secure e-mail system was developed that relied totally on IBE, where a Microsoft Outlook Add-in has been programmed. It allows the user to encrypt outgoing e-mail messages and decrypt incoming e-mail messages [9]. However, it doesn’t provide signing services, and didn’t use any existent secure e-mail standard. Another example of such research is the “Identity-Based Mediated RSA” [10] which is a simple variant of Mediated RSA that combines identity-based and mediated cryptography. In IB-Mediated RSA, users’ RSA public key can be computed publicly from their e-mail address so that no individual certificate is needed. A Microsoft Outlook Add-in has been developed to demonstrate this method. This approach also does not make use of any existing secure e-mail standard.

Also a "Lightweight Signatures for E-mail" was proposed in [11] which is a mechanism for Internet-wide distribution of identity-based public keys for the purpose of e-mail authentication. Each e-mail domain becomes a master authority for an Identity-Based scheme of its choosing and generates a unique master key pair (*MPK*, *MSK*). Each *MPK* is distributed via the Domain Name System (DNS). It was later updated by Lightweight E-mail Signatures (LES) [12] which is an extension to DomainKeys Identified Mail (DKIM), a mechanism by which domains are made cryptographically responsible for the e-mail they send. While this solution works well for e-mail authentication, e-mail encryption exhibits a different threat model that requires special treatment. "Lightweight Encryption for E-mail" [13] was proposed to strengthen the security model by key splitting and distributed key generation. These latter approaches can be deployed with client or server updates.

However, they were not concerned with specifying a standard secure e-mail message format as they were mainly describing methods for deploying an Identity-Based PKI.

In order for two e-mail users to communicate securely, it is necessary for them to use programs that are compatible with each other. Hence secure e-mail systems that utilize IB-PKC algorithms need to make use of existing e-mail security standards in order to be widely adopted and make secure messaging easier to use. In this paper, a secure e-mail system is proposed that is based on S/MIME standard where the public key and signature algorithms have been replaced by their IB-PKC analogue algorithms. We believe that our approach would result in a better usability of secure e-mail systems. The rest of this paper is organized as follows: Section 2 briefly introduces S/MIME and Identity-Based Cryptography. Some important design considerations are outlined in Section 3. The Specifications of messages used in the proposed system are described in Section 4, while the overall system architecture is presented in Section 5. Then, some notes on the implementation and performance are given in Section 6. Also, some possible extensions to existing system implementation are highlighted in Section 7. Finally, some important points are concluded in Section 8.

2. Preliminaries

This section briefly presents the basic building blocks of our approach to develop a better usability secure e-mail system. These are: S/MIME and IB-PKC.

2.1 S/MIME

MIME (Multipurpose Internet Mail Extensions) extends the format of Internet Mail to allow non-ASCII textual messages, non-textual messages, multi-part message bodies, and non-ASCII information in message headers. Messages in MIME format can contain files of different types. The file type is described in a "content-type" header that is used by the recipient e-mail program to determine how to handle that file. MIME also defines a set of content transfer encodings which can be used to represent 8-bit binary data using characters from the 7-bit ASCII character set.

S/MIME (Secure/Multipurpose Internet Mail Extensions) provides a consistent way to send and receive secure MIME data. Based on the MIME standard, S/MIME provides the following cryptographic security services for electronic messaging applications: data confidentiality (using encryption), authentication, message integrity and non-repudiation of origin (using digital signatures). S/MIME messages are made up of MIME bodies and Cryptographic Message Syntax (CMS) objects. There are several CMS content types. Of

these, only four content types are currently used by S/MIME:

- Data: Sending agents must use the "id-data" content type identifier to identify the "inner" MIME message content.
- SignedData: This content type is used to apply a digital signature to a message or, in a degenerate case where there is no signature information, to convey certificates.
- EnvelopedData: It is used to apply data confidentiality to a message. A sender needs to have access to a public key for each intended message recipient to use this service.
- CompressedData: This content type is used to apply data compression to a message. It is only used to reduce message size.

2.2 Identity-Based Cryptography

The concept of Identity-Based Public Key Cryptography (IB-PKC) was first proposed by Shamir in 1984 [14], where it was shown that the authenticity problem in public key cryptography (PKC) can be solved without the use of certificates. What makes IB-PKC differs from the ordinary PKC is that the public key can be an ordinary string. For example, if Alice wants to encrypt a message to Bob at "bob@foo.com", then she simply encrypts the message using "bob@foo.com" as the public key. Thus, any user in the system can send encrypted e-mail messages to anyone, even if the recipient hasn't yet registered and created his/her own private key. Once a recipient receives the encrypted message, that recipient contacts the Private Key Generator (PKG) in order to request his/her private key that is needed for decryption, and it's the only time in which this user has to contact any Trusted Third Party (TTP) until he/she needs to update the private key later.

The primary advantage of IB-PKC is that a system participant can make a public key without having to contact any TTP. Moreover, it can be used for managing user credentials, delegation of decryption keys, or for implementing short lived public keys. It has also been used to build forward-secure encryption schemes.

While efficient solutions for Identity-Based Signature (IBS) schemes were quickly found, most IBE schemes proposed since 1984 were unsatisfactory because they were too computationally intensive, they required tamper resistant hardware, or they were not secure if users colluded. The first efficient and secure IBE scheme did not appear until 2001 when Boneh and Franklin [15] presented their IBE scheme. The Boneh-Franklin IBE scheme (BF-IBE) is based on bilinear maps between groups. The BF-IBE scheme has chosen ciphertext security in the random oracle model assuming a variant of the

computational Diffie-Hellman problem. Moreover, its performance is comparable to the performance of ElGamal encryption.

After that, several other identiy-based schemes were proposed based on the BF-IBE scheme. For example, the Cha-Cheon IBS scheme (CC-IBS) [16] shared the same system parameters with the BF-IBE scheme. Combining these two schemes yields a complete solution for of an Identity-Based Public Key system. In an IBE scheme there are four algorithms:

- *Setup*: takes as input a security parameter and outputs params (system parameters) and the master-key. The system parameters must include the description of the message space M and the ciphertext space C. The system parameters would be publicly known while the master-key is known only to the Private Key Generator (PKG).
- *Extract*: takes as input the system parameters (params), the master-key and an arbitrary string ID $\in \{0,1\}^*$ and outputs the private key d_{ID} corresponding to the public key ID.
- *Encrypt*: takes as input the system parameters (params), a public key ID and a plaintext $M \in M$ and outputs the corresponding ciphertext.
- *Decrypt*: takes as input the system parameters (params), a private key d_{ID} and a ciphertext $C \in C$ and outputs the corresponding plaintext.

IBS schemes also have four algorithms. However, *Encrypt* and *Decrypt* are replaced by the algorithms *Sign* and *Verify*. The algorithm *Setup* is run by the PKG. The PKG also runs the algorithm *Extract* at the request of a user who wishes to obtain the private key corresponding to some string. The user should prove to the PKG that he/she is the legitimate owner of this string. The algorithms *Encrypt*, *Decrypt*, *Sign*, and *Verify* are run by the users to encrypt, decrypt, sign, and verify messages.

3. The Proposed System Design Considerations

The proposed secure e-mail system should securely transmit e-mail messages, be easy to use, make use of the existing secure e-mail standards, and it should be applied without making significant changes to the structure of the network. In order to achieve the previous goals, some decisions had to be made before designing have the system:

- The first question to be answered is whether to apply security to both the e-mail client and server, or just one of them. Any change in the e-mail

servers is not recommended, since this implies that all the e-mail servers around the world should be updated to implement the new changes. Hence, this design would apply security to e-mail clients only, and this will allow any organization to apply this system without having to modify the underlying network architecture.

- The analysis of the current PKI [6] shows that different aspects of the PKI technology have attracted criticism, and shows that most of these aspects are related directly to the digital certificates' management complexity. On the other hand, IB-PKC provides a comparable security and an equivalent functionality, and does not need any digital certificates. Thus, IB-PKC represents an excellent replacement to the PKI technology, and it will be adopted in the design of this system.
- Distributing private keys to the users of the system is a main concern when designing any secure e-mail system. There are three options to choose from:

  1. A method that is based on the E-mail Based Identification and Authentication (EBIA) [17] approach. It involves developing an e-mail-based application that would be able to receive private key request e-mail messages, and respond to them.
  2. Providing private keys manually by generating them using a private key extraction tool, and distributing them using disks, for example.
  3. Designing a web page that a user can use to input his/her public key and a password, and then a tool generates the private key, encrypt it with the password, and send it back to the user.

  The first method seems to be the most appropriate because the PKG design would be similar to the secure e-mail client design, and would simplify the process of generating private keys.
- Proposing a totally new secure e-mail standard and ignoring the currently available solutions is not a practical approach, since any new standard should pass many tests to be considered secure and eligible to be implemented and used widely. Two options arise when considering an existing

secure e-mail solution, and they are S/MIME and OpenPGP. S/MIME is chosen to be the base standard for this design, because it has been designed to be a standard with a neatly designed structure and a proven security. Moreover, it is based on MIME that will allow users to benefit from all the advantages of electronic mail today.

- The cryptographic algorithms to be used should have the best security, efficiency, and usability characteristics compared to other available candidates. Thus, the AES would be used for the conventional encryption, the Boneh-Franklin IBE scheme [15] for the public-key algorithm, the Cha-Cheon IBS scheme [16] for the Signature algorithm, and the SHA-1 for the secure hash function. The Boneh-Franklin IBE scheme (BF-IBE) and the Cha-Cheon IBS scheme (CC-IBS) have been chosen together because they have a similar initial steps. The public system parameters of both algorithms can be combined to compose general public system parameters [16].
- The only needed information for any participant to start using the system is the public system parameters. These parameters can be stored in a file that would be publicly available. They are needed for using the Identity-Based Cryptography algorithms. These parameters are the result of combining BF-IBE scheme and CC-IBS scheme public system parameters (where the master key is *s*, and it is chosen randomly provided that $s \in Z_p^*$):
  1. G_1, G_2, $\hat{e}$, p: G_1 and G_2 are cyclic groups of prime order p together with a bilinear map $\hat{e}$: $G_1 \times G_1 \rightarrow G_2$ corresponding to this security parameter.
  2. P: a random generator $P \in G_1$.
  3. P_{pub}: $P_{pub} = s*P$.
  4. Cryptographic hash functions.
- In IB-PKC, the public key can consist of three parts (e-mail address || current-date || clearance-level) to add extra functionality. However, for the first version of our implementation, the public key consists of the e-mail address only. The main reason for this is to demonstrate the basic idea without adding excessive work and increasing the

complexity of the system. It is possible to append the additional strings that represent the additional functionality at a later time without changing the main structure of the design (see Section 7).

4. Secure Message Specifications

Since S/MIME is the basis for this system, the message format would follow the S/MIME message specifications [18] and the CMS standard [19] with slight modifications to some fields in order to add the functionality of the Identity-Based Cryptography. In this section, the modifications that have been applied to the CMS EnvelopedData and SignedData types are described. The CompressedData doesn't need to be modified. The Data content type also doesn't need to be modified since it only exists to contain the message content (like MIME body part) that is going to be processed by S/MIME. However, before continuing, it is important to notice that:

- The CMS Version of the current S/MIME is 3.1, while in our design it is considered 5; in order to differentiate it from the existing S/MIME standard that depends on a different set of public-key encryption algorithms.
- The Digest algorithms and the Message encryption algorithm are exactly as specified in the S/MIME specification. However, the Digital signature algorithms and the Session key encryption algorithms are replaced by IB-PKC algorithms. The Cha-Cheon IBS Scheme is going to be the Digital signature algorithm, and the Boneh-Franklin IBE Scheme is going to be the Session key encryption algorithm.
- All the optional fields in the S/MIME message specification will be discarded in our first implementation; as they are mostly related to Digital Certificates, and they won't be used in our system.
- There can be more than one recipient for an encrypted message, and more than one signer for a signed message according to the S/MIME Specifications. However, in order to simplify the explanation of the system; only one sender and one receiver are assumed in the following description of the SignedData and EnvelopedData types. (Section 7 explains how to handle multi-recipient encrypted messages and multi-signer signed messages)
- Sending agent is software that creates S/MIME CMS objects, MIME body parts that contain CMS objects,

or both. Receiving agent is software that interprets and processes S/MIME CMS objects, MIME body parts that contain CMS objects, or both. S/MIME agents are user software that is a receiving agent, a sending agent, or both.

4.1 EnvelopedData

The EnvelopedData content type consists of an encrypted content of any type and encrypted content-encryption key for a recipient. Fig. 1 shows the structure of the EnvelopedData content type after neglecting the optional fields. The EnvelopedData content type is composed of:

- *CMSVersion*: it must be set to 5.
- *RecipientInfo*: Recipient information is represented in this field. It identifies the way that a receiving e-mail client can obtain the session-key in order to decrypt the message. CMS provides five options to choose from in this field. The OtherRecipientInfo type is chosen in order to make our own structure to fill this field. The OtherRecipientInfo type allows key management techniques beyond the pre-defined ones. The OtherRecipientInfo type consists of:
  - *OBJECT IDENTIFIER*: identifies the key management technique. Here it is set to "IBC" in order to denote the use of Identity-Based Cryptography.
  - *CMSVersion*: it must be set to 5.
  - *KeyEncryptionAlgorithmIdentifier*: identifies the key-encryption algorithm used to encrypt the content-encryption key with key-encryption key. It's always set to "BF-IBE" (Boneh-Franklin IBE Scheme).
  - *EncryptedKey*: it is the result of encrypting the content-encryption key for the recipient.
- *EncryptedContentInfo*: Encrypted content Information is represented in this field, and it consists of:
  - *ContentType*: indicates the type of content, and it can be a content of any type.
  - *ContentEncryptionAlgorithmIdentifier*: identifies the content-encryption algorithm used to encrypt the content.
  - *EncryptedContent*: it is the result of encrypting the content.

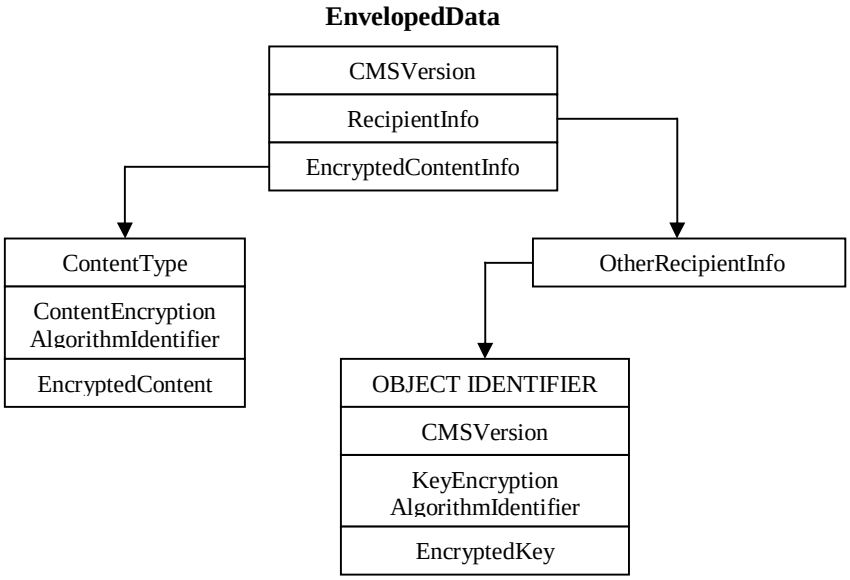

Fig. 1: EnvelopedData Structure.

4.2 SignedData

The SignedData content type consists of a content of any type and one or more signature values. Fig. 2 shows the structure of the SignedData content type after neglecting the optional fields. The SignedData content type is composed of:

- *CMSVersion*: it must be set to 5.
- *DigestAlgorithmIdentifier*: identifies the message digest algorithm employed by the signer.
- *EncapsulatedContentInfo*: This field represents the signed content, consisting of a content type identifier and the content itself:
  - *ContentType*: indicates the type of content, and it can be a content of any type.
  - *Content*: This is the content itself, represented as an octet string.
- *SignerInfo*: Signer information is represented in this field. And it consists of:
  - *CMSVersion*: it must be set to 5.
  - *SignerIdentifier*: this field should specify the signer's certificate, and thereby the signer's public key. However, we won't need it since the IBC is used, and the sender's e-mail address is the public key. Since this field is not optional, it can not be omitted. An arbitrary value will be put in this field, and the CMS

Version is going to be an indicator that the public key is the sender's e-mail address (when CMS Version is set to 5).

- *DigestAlgorithmIdentifier* : identifies the message digest algorithm employed by the signer.
- *SignatureAlgorithmIdentifier*: identifies the signature algorithm used by the signer to generate the digital signature. It's always set to "CC-IBS" (Cha-Cheon IBS Scheme).
- *SignatureValue*: it's the result of digital signature generation using the message and the signer's private key.

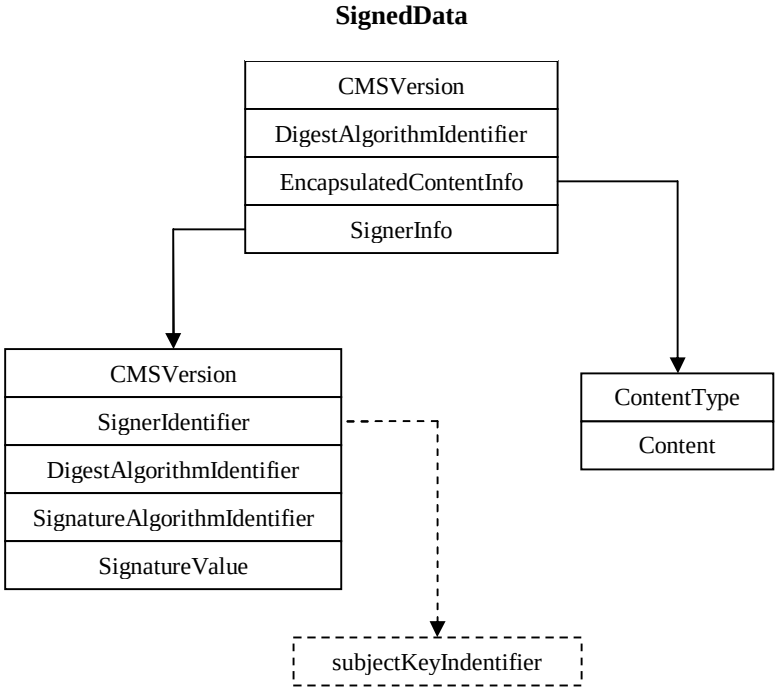

Fig. 2: SignedData Structure

5. SYSTEM ARCHITECTURE

The proposed system is composed of three parts: the Secure E-mail Client, the Private Key Generator (PKG), and an E-mail Server as shown in Fig. 3. The e-mail server is represented in dotted lines since it is not affected by our system approach, and any existing e-mail server would work well when integrated with system. Thus, there is no need to re-design the e-mail server.

5.1 Secure E-mail Client

Secure E-mail clients are mainly used for viewing incoming e-mail messages, composing outgoing

e-mail messages, managing messages' folders, and applying and managing e-mail security services. The modules needed by this application can be identified as follows (see Fig. 4):

- E-mail message module that should provide the ability to create and parse e-mail messages according to the Internet message format and the MIME.
- S/MIME module should contain a set of functions that provide ability to encrypt and/or sign MIME entities, and the ability to decrypt and/or verify MIME entities. It has to comply with the modifications to the S/MIME that were mentioned in section 4.
- Cryptography module should provide a set of functions that would be required to carry out the cryptographic operations needed by the secure message specification described in section 4.
- SMTP module should contain a set of functions responsible for sending e-mail messages using the SMTP protocol.

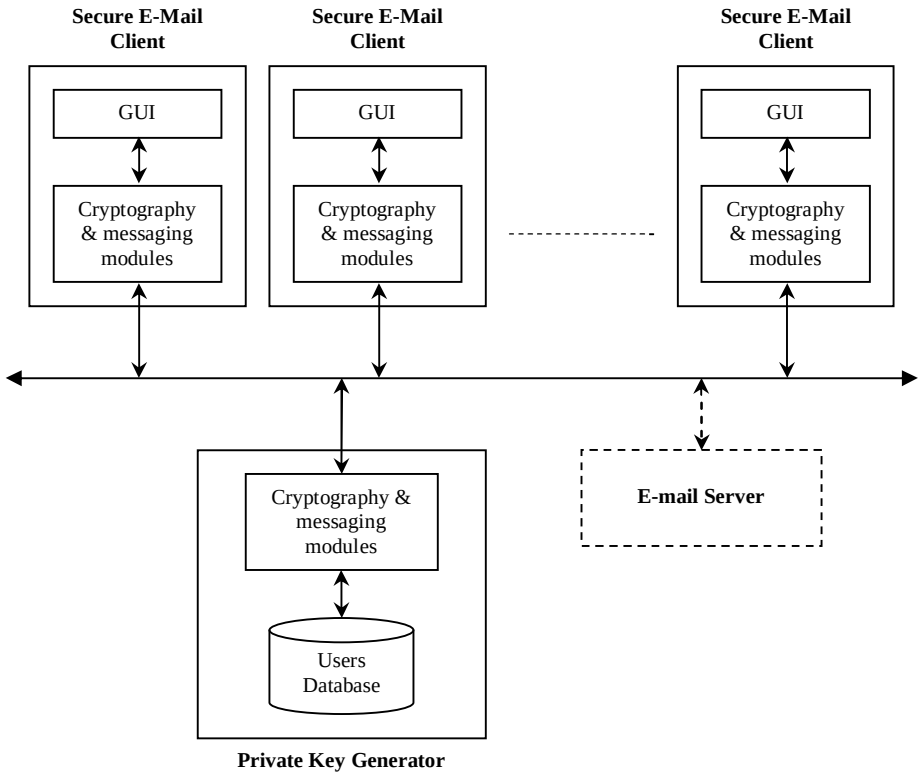

Fig. 3: System Architecture.

- POP3 module should contain a set of functions responsible for receiving e-mail messages using the POP3 protocol.
- Database module that should provide all the necessary data structures and the functions needed to interact with them. This can include an options' file structure, a folders' indexing structures (to enhance the performance of the application by reducing the time needed to search for the e-mail message files and scan them each time a folder is selected), and a contacts' information structure.
- Private Key management module that should provide a set of functions responsible for requesting the private key corresponding to the user's public key.
- An easy-to-use graphical user interface (GUI) that would allow the user to easily compose and view messages, apply security services to them, send and receive messages, manage the messages' folders, and manage all the other necessary operations needed by the application.

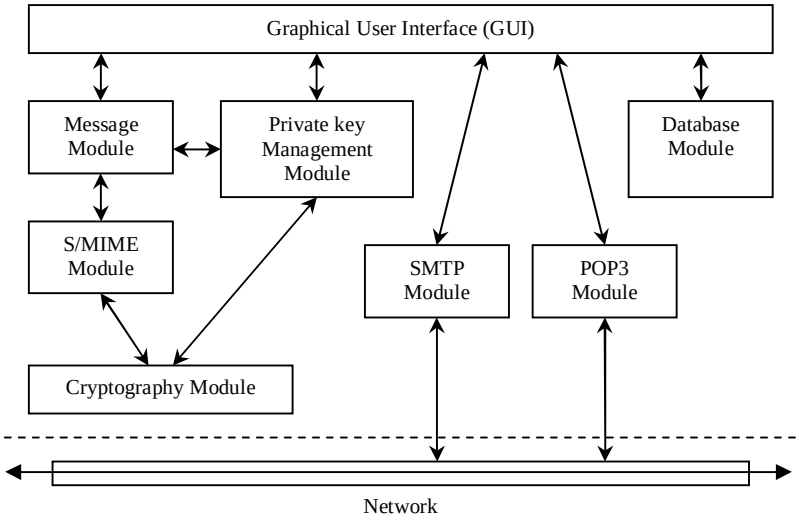

Fig. 4: Secure E-mail Client Architecture.

5.2 Private Key Generator (PKG)

PKG is the part of the system representing the trusted authority that would perform the first two steps needed by any system that uses the Identity-Based Cryptography, setup and extract. It first runs the setup algorithm to

generate the public system parameters, and the master key, and it does that only once. Then the PKG runs the extract algorithm for every request to generate the private key out of the public key (e-mail) supplied. The extraction process is carried out according to the protocol that is described in Subsection 5.3, where a request would be received from the e-mail client using a special e-mail message format. Then the PKG generates the private key, and sends it back to the e-mail client. To accomplish these tasks, the modules needed by the PKG can be identified as follows:

- Cryptography module should provide a set of functions that would be required to run the setup and extract algorithm according to the IBE Scheme, and to carry out the cryptographic operations needed by the secure message specification described in section 4.
- E-mail message module that should provide the ability to create and parse e-mail messages according to the Internet message format and the MIME.
- S/MIME module should contain a set of functions that provide the ability to encrypt and/or sign MIME entities, and the ability to decrypt and/or verify MIME entities. It has to comply with the modifications to the S/MIME that were mentioned in section 4.
- SMTP module should contain a set of functions responsible for sending an e-mail message using the SMTP protocol. POP3 module should contain a set of functions responsible for receiving an e-mail message using the POP3 protocol.
- Message Management module should be executed synchronously with the rest of the application. It checks the outbox folder periodically to see whether there are messages to be sent. It also checks the e-mail server for messages to receive them and store them in the inbox folder.
- Database module that should provide all the necessary data structures and the functions needed to interact with them. This can include an options' file structure, details of the rejected requests, a list of the public keys that has their private keys extracted, and a list of the allowed public keys (e-mail addresses) and domain names.
- Private Key extraction module should contain a set of function that would be required to check for any new request, check the e-

mail address to see if it's in the allowed list, extract the private key, send it back to client that requested it, and record all the necessary information in the data structures.

- An easy-to-use graphical user interface (GUI) that would allow the user to easily configure the PKG and view the rejected requests, the executed request, and the application's log file.

5.3 Private Key Distribution Method

This subsection explains the method that the e-mail client can use to extract the private key associated with its public key (e-mail address). It's an e-mail dependent method that is based on the E-mail Based Identification and Authentication (EBIA) method. EBIA uses an e-mail address as a universal identifier and the ability to receive e-mail at that address as a kind of authenticator [17]. Mailing lists and forums subscription confirmations, Web password reminders, and e-commerce notifications all use this method. In our system, private keys, which are very sensitive data, are distributed using EBIA. This is why EBIA needs to be modified by the use of fine cryptographic principles in order to maintain its security.

This key distribution method is carried out between the Private Key management module in the e-mail client and the Private Key extraction module in the PKG as shown in Fig.6. Before being able to extract its private key, the e-mail client should obtain the public system parameters, a common information file that should be provided by the PKG. The user should also provide the e-mail address of the PKG. The steps needed to accomplish this task are (see Fig. 5):

1. The secure e-mail client sends a request to the PKG:
   a. It generates a random session key (e.g, AES key), and save it in a file called "session.key". The file is copied and stored for future use.
   b. The session key file is then encapsulated in a MIME entity. The MIME entity is processed into a CMS object of type EnvelopedData (Subsection 4.1) where the public key used is the e-mail address of the PKG.
   c. A message is composed that contains the generated "EnvelopedData" object. The subject of this message is "Request".
   d. Then the secure e-mail client puts this message in the outbox folder ready to be sent.
2. The PKG responds to the request:

a. The PKG receives the message, checks its subject line. If the subject line contains "Request", the PKG processes the "EnvelopedData" object to decrypt the encrypted "session.key" file using its private key.
b. Then the PKG takes the e-mail address present in the “From:” field of the message, extracts the corresponding private key, encrypts it using the session key, and stores the encrypted private key in a file called "encprivate.key".
c. The private key file is then encapsulated in a MIME entity. The MIME entity is processed into a CMS object of type SignedData (Subsection 4.2) where the private key used to sign the content is that of the PKG.
d. The PKG composes a message that contains the generated “SignedData” object, and sends it back to the sender.

3. The Secure e-mail client receives the private key:
   a. The secure e-mail client receives a message from the PKG e-mail address, and the Private Key management module verifies that the message is really from the PKG using the "SignedData" object and the PKG's public key (e-mail address).
   b. The encrypted private key file ("encprivate.key") is then decrypted using the stored session key.
   c. The private key is ready to use.

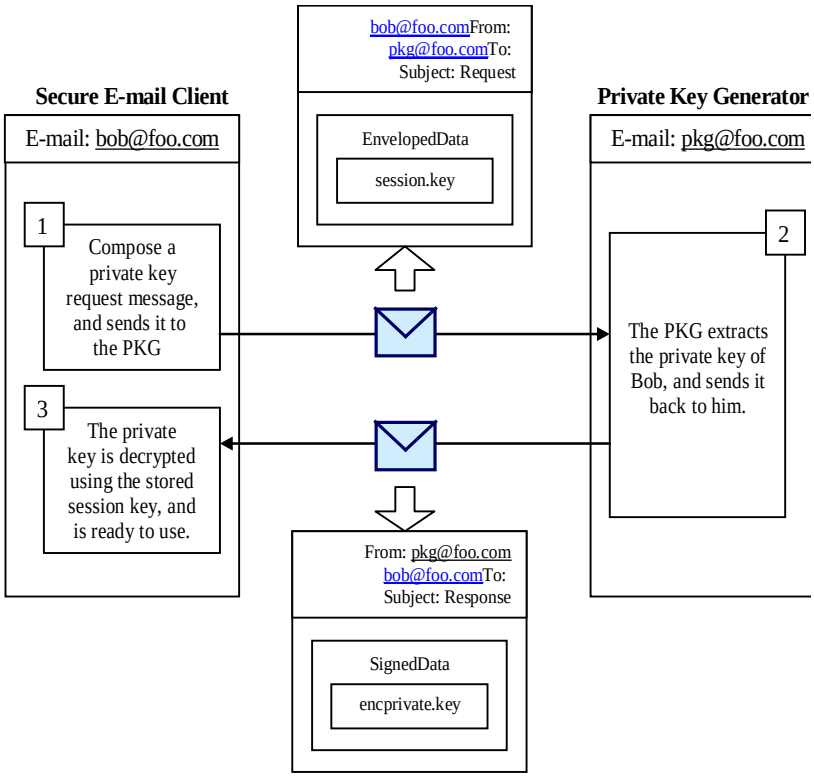

Fig. 5: Private Key Distribution Method

6. IMPLEMENTATION AND PERFORMANCE

The language used in programming this system is C/C++, and the compiler used is the Mircosoft Visual C++. Two applications were developed: the secure e-mail client and the Private Key Generator (PKG). Some modules were needed by both application, and they are the cryptography, S/MIME, E-mail Message, SMTP, and POP3 modules. On the other hand, some modules were specific to each one of the two applications. Some open source libraries were used like the SFL library [20] and the MIRACL library [21] in order to speed up the development time, to avoid increasing the number of bugs in the applications, and to ensure that the best performance has been achieved since these libraries have been thoroughly tested by many developers around the world.

The system was tested on a PC with Intel Celeron 2GHz processor, 256 MB of RAM, 100Mbps Fast Ethernet PCI adapter, and has Windows XP Professional installed. The secure e-mail client uses an average of 9 MB of RAM, while the PKG uses an average of 5.6 MB of RAM. A code profiler has been used to

measure the time needed to create encrypted or signed e-mail messages. These timing results are shown in Fig. 6 and Fig. 7 for encrypted and signed messages respectively. It is obvious that encrypting messages requires a higher time than signing them given the same message size.

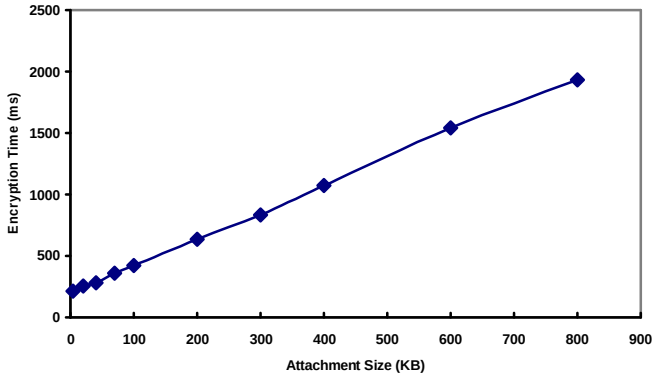

Fig. 6: Encryption Time Graph.

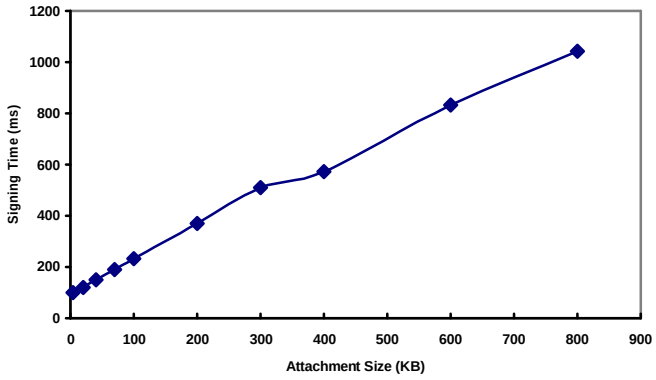

Fig. 7: Signing Time Graph.

7. POSSIBLE EXTENSIONS

In this section we describe possible extensions to our system. These extensions allow preparing encrypted messages with multiple recipients (Subsection 7.1) and signed messages with multiple signers (Subsection 7.2). The previous two extensions permits the use of public keys that are not confined to e-mail addresses like (e-mail address || current-date || clearance-level). Subsection 7.3 presents the approach for Identity-Based PKI described in [12,13], and shows how to integrate it with our system.

7.1 Multiple-Recipient Encrypted Message

Another field should be added to the "OtherRecipientInfo" field (see Subsection 4.1) to make it possible to send encrypted data to more than one recipient. This field would contain the recipient's public key, and can be called "RecipientPublicKey". The content is encrypted using a random session key as usual. For every recipient, a separate "RecipientInfo" field is prepared that contains the encrypted session key and the public key that was used to encrypt the session key.

7.2 Multiple-Signer Signed Message

The "SignerInfo" type (see Subsection 4.2) contains a field that identifies the public key of the signer. It's the "SignerIdentifier" field, and it provides two alternatives for specifying the signer's public key. They are the "issuerAndSerialNumber" and the "subjectKeyIdentifier". The "issuerAndSerialNumber" identifies the signer's certificate by the issuer's distinguished name and the certificate serial number. The "subjectKeyIdentifier" identifies the signer's certificate by the X.509 subject Key Identifier extension value. Both of these choices identify the signer's certificate (and thereby the signer's public key). However, certificates are not used in our system, so another way should be found to identify the signer's public key.

Thus, if more than one signer wants to sign the content, a third option should be added to the "SignerIdentifier" field. This option field can be called "SignerPublicKey" and contains the signer's public key. For every signer, a separate "SignerInfo" field is prepared that contains the signature value and the public key of the person that used his/her private key to generate this signature. The problem with this approach is that it violates the CMS standard since it's not permitted to have user-created alternative choices in the "SignerIdentifier" field.

7.3 Identity-Based PKI

Using a single PKG from which all users request their private keys may not be practical in some situations. Each institution may want to have its own PKG and manage its users' keys. Moreover, a PKG's secret key must not exist on a single machine since it can be used to decrypt all the messages of this PKG's users. The Lightweight PKI, described in [12] and [13], can be used to solve these problems. In this Identity-Based PKI, each domain has its own PKG that extracts private keys for its users. The Master Public Key (*MPK*) of a domain is distributed via the DNS, as a TXT record associated with the hostname of the domain's Mail Exchange (MX) record.

In fact, a domain should maintain several PKGs that independently generate master key

shares (MPK_i, MSK_i). The Master Public Key shares are combined into a single *MPK* that is distributed via the DNS. Each server with Master Secret Key MSK_i individually sends Alice secret key share $SK_{Alice,i}$. Alice then can combine all shares into a single private key SK_{Alice}. EBIA is used to distribute these secret key shares [13].

The integration of our system with the Identity-Based PKI, described above, can be carried out without modifications to our system. However, it's better to apply the changes presented in Subsections 7.1 and 7.2 first in order to exploit all the capabilities of the Identity-Based PKI, like allowing a user to extract his private key from a domain different than his original domain.

In [12], EBIA is used to distribute secret key shares, while our system uses a modified version of EBIA (Subsection 5.3) to distribute private keys. Our modified EBIA is more secure and can be used in the distribution of the secret key shares. A user can send request e-mail messages to all the PKG servers of a single domain. Each one of these PKGs receives a different session key in the request message, generates a secret key share, encrypts it with the received session key, and sends it back the user. The user then decrypts these secret key shares using the stored session keys. After that he/she can combine these secret key shares to form his/her private key. Thus, all the secret key shares would never be sent without encryption.

8. Conclusions

Our system provides a solution to the usability problem present in the existing e-mail security solutions. Anyone can start using the system after getting a copy of the common system parameters that can be publicly provided. When a user wants to get his/her private key, the only thing needed is the e-mail address of the PKG. The system is compatible with the S/MIME protocol. In fact, it’s based on S/MIME to benefit from the structure of S/MIME that has gone through extensive testing and improvement for several years now. This also decreases the time and effort needed to add this system’s functionality in the existing e-mail applications that implement S/MIME. The changes made by this system need to be applied on the e-mail clients only, while the e-mail servers stay intact. These changes to the e-mail clients don’t need to be integrated into them. They can be provided in the form of Add-Ins to these applications.

In addition, as a future work, the PKG in our system can be enhanced further. The enhancement my include finding a method that enables users who have different PKGs to use the system while maintaining the same level of security and usability. This has been partially covered in Subsection 7.3.

References

1. Linn, J., “RFC 989: Privacy Enhancement for Internet Electronic Mail, Part I: Message Encipherment and Authentication Procedures”, Internet Engineering Task Force (IETF), February 1987.
2. Atkins, D., Stallings, W. and Zimmermann, P., "RFC 1991: PGP Message Exchange Formats", Internet Engineering Task Force (IETF), August 1996.
3. Dusse, S., Hoffman, P., Ramsdell, B., Lundblade, L. and Repka, L., “RFC 2311: S/MIME Version 2 Message Spcification”, Internet Engineering Task Force (IETF), March 1998.
4. Whitten, A. and Tygar, J., “Why Johnny can’t encrypt: A usability evaluation of PGP 5.0”, *The 8th USENIX Security Symposium*, pages 169 – 184, 1999.
5. Garfinkel, S., “Creating Systems that are Simultaneously Usable and Secure”. PhD thesis, the Massachusetts Institute of Technology (MIT), May 2005.
6. Linn, J., and Branchaud, M., “An Examination of Asserted PKI Issues and Proposed Alternatives”, *In Proceedings of the 3rd Annual PKI R&D Workshop*, pages 34-47, March 2004.
7. Whitten, A. and Tygar, J., “Safe staging for computer security”, *In Proceedings of the 2003 Workshop on Human-Computer Interaction and Security Systems*, April 2003. "http://www.andrewpatrick.ca/ CHI2003/HCISEC/".
8. Garfinkel, S., “Enabling e-mail Confidentiality through the use of Opportunistic Encryption”, *In The 2003 National Conference on Digital Government Research, National Science Foundation*, pages 173-176, 2003.
9. Baldwin, M., “Identity Based Encryption from the Tate Pairing to Secure E-mail Communications”. Master of Engineering Thesis, University of Bristol, May 2002.
10. Ding, X. and Tsudic, G., “Simple Identity-Based Cryptography with Mediated RSA”, *Cryptographer's Track RSA Conference*, 2003.
11. Adida, B., Chau, D., Hohenberger, S. and Rivest, R., "Lightweight Signatures for Email", *a preliminary version presented in the DIMACS Workshop on Theft in E-Commerce*, April 2005.
12. Adida, B., Chau, D., Hohenberger, S. and Rivest, R., "Lightweight Email Signatures", February 2006. Available at" http://theory.lcs.mit.edu/~rivest/publications.html".
13. Adida, B., Hohenberger, S. and Rivest, R., "Lightweight Encryption for Email", *In Proceedings of Usenix's Symposium on Reducing Unwanted Traffic on the*

Internet, pages 93-99, July 2005.

14. Shamir, A., "Identity-Based Cryptosystems and Signature Schemes", *Proceedings of Crypto '84*, pp. 47-53, 1984.
15. Boneh, D. and Franklin, M., "Identity Based Encryption from the Weil Pairing", *Proceedings of Crypto 2001*, Lecture Notes in Computer Science (LNCS) 2139, pp 213 - 229, Springer-Verlag, 2001.
16. Cha, J. and Cheon, J., "An Identity-Based Signature from Gap Diffie-Hellman Groups", *Practice and Theory in Public Key Cryptography-PKC 2003*. Also Cryptology ePrint Archive 2002/018, 2002.
17. Garfinkel, S., "Email-based identification and authentication: An alternative to PKI?", *Security & Privacy Magazine*, 1:20–26, Nov. - Dec. 2003.
18. Ramsdell, B., "RFC 3851: Secure/Multipurpose Internet Message Extensions (S/MIME) Version 3.1 Message Specification", Internet Engineering Task Force (IETF), July 2004.
19. Housley, R., "RFC 3852: Cryptographic Message Syntax (CMS)", Internet Engineering Task Force (IETF), July 2004.
20. SFL, S/MIME Freeware Library, "http://www.digitalnetgov.com /hot/ sfl_home.htm".
21. MIRACL, Multiprecision Integer and Rational Arithmetic C/C++ Library, "http://indigo.ie/~mscott/".