Multiplex Encryption: A Practical Approach to Encrypting Multi-Recipient Emails

Wei Wei[1], Xuhua Ding[2], and Kefei Chen[1]

[1] Computer Science Department, Shanghai JiaoTong University
Email: www_weiwei, kfchen@sjtu.edu.cn
[2] School of Information Systems, Singapore Management University
Email: xhding@smu.edu.sg

Abstract. Efficiently protecting the privacy of multi-recipient emails is not as trivial as it seems. The approach proposed by S/MIME is to concatenate all ciphertexts. However, it suffers from poor scalability due to its linear computation and communication cost. In this paper, we propose a new practical and secure scheme, called *multiplex encryption*. By combining the ideas of identity-based mediated RSA and re-encryption mix network, our framework only requires constant computation and communication cost to encrypt a multi-recipient email.

1 Introduction

The electronic mail is one of the most popular media for data exchange. People send and read emails from their computers, PDAs or even cellular phones. A huge volume of information are transferred via emails within an organization or across organizations. Among them, those with confidential data are encrypted such that only the intended receivers are able to access the content while an adversary obtains no information about the protected data. With the support of the public key infrastructure, those emails are encrypted under the recipients' public keys. According to S/MIME[14], the current standard for secure emails proposed by IETF's S/MIME Working Group, a confidential email is encapsulated by an ”encryption envelop”. Specifically, the content is encrypted by a randomly selected symmetric key, which is encrypted under the recipient's public key.

It seems trivial to extend the one-to-one email encryption method to multi-recipient emails, in which case the same email is sent to a number of receivers by using *carbon-copy*. In S/MIME approach, the sender produces different encryption envelops for different recipients. All envelops are pushed into a stack. The receiver seeks and decrypts the envelop intended for him from the stack. Unfortunately, this approach has several drawbacks. First, the computation cost at the sender's end is linear to the number of recipients. Note that users usually expect immediate email forwarding. Therefore, the delay incurred by encryption offsets the user friendliness of emails, since it stalks email delivery. Secondly, the length of the message to transfer is linear to the number of recipients, which contradicts the motivation of using carbon-copy. In a SMTP transaction for carbon-copy, the sender first interacts with the SMTP server to confirm every recipient's address in the carbon copy list. Then, only one copy of email, instead of multiple copies, is forwarded to the SMTP server. Although the same protocol applies to sending an encrypted multi-recipient email, the message to forward is essentially a concatenation of multiple ciphertexts. Therefore, the communication cost is not saved. Lastly, not the least, the concatenation of ciphertext ruins an important feature of carbon copy. Carbon-copy is used not only for the purpose of cost saving, but usually as a means to imply that the same message is equally read by all the intended recipients. For instance, a sales representative sends carbon-copy of a draft of contract to both his supervisor and his clients, so that all related entities receive consistent information about the contract terms; a committee chair expresses his opinion to all committee members by sending a carbon-copy email so that everyone on the committee receive the same statement. We observe that this feature is not preserved when carbon-copy emails are encrypted. This is due to the fact that each recipient only decrypts his own part and has no knowledge of the results from others' decryption.

Encryption of multi-recipient emails is in fact surprisingly challenging. Though the communication model is similar to multicast, the approach to multicast security is infeasible for emails. Multicast encryption requires stable group membership and a key establishment phase. For emails, the recipients are chosen when the sender composes the email. Therefore, the grouping of recipients is ephemeral. Furthermore, it is impractical to require all recipients to be online to agree on a key. An ideal solution would be to design a new public key encryption scheme such that the sender constructs an encryption key $\mathcal{PK}$ from a set of public keys $\{PK_0, \cdots, PK_n\}$. The ciphertext produced by $\mathcal{PK}$ can be decrypted by corresponding private keys $\{SK_0, \cdots, SK_n\}$ while the notion of semantic security still preserves. However, it is an open question whether such an encryption scheme exists.

In this paper we take a systematic approach to encrypting multi-recipient emails. We present a *multiplex encryption* system for multi-recipient emails. Built on top of mediated RSA[5], our scheme enables the sender to multiplex several ciphertexts into one so that the encryption cost and the length of ciphertext is independent of the number of recipients. In the recipient's end, our scheme introduces a partially trusted server which, functioning as *demultiplexor*, translates the ciphertext and forwards the results to the corresponding recipients respectively. Note that the server is not *fully* trusted in the sense that the server is unable to decrypt the ciphertext by itself.

The rest of the paper is organized as follows. In next section, we show the related work. The details of multiplex encryption are presented in Section 3. We discuss its security in Section 4 and its performance in Section 5. A summary is provided in Section 6, which concludes the paper.

2 Related Work

Our scheme is built on top of identity-based mediated RSA[5]. In [5], a user's RSA public key is derived from its identity, e.g. an email address, and the corresponding RSA private key is split into two shares. One share is secretly owned by the user and the other is secretly owned by a partially trusted server. To produce a signature or decrypt a ciphertext, the user has to collaborate with the server. The main motivation of their work is for fine-grained control over users' security capabilities.

Public key encryption in multi-user setting was studied in [1], which discussed the security relation between single-user setting and multi-user setting. In [9], Kurosawa shorten the ciphertext by a half for ElGamal and Crammer-Shoup encryption in multi-user setting. Bellare et. al. further investigated the security of random reuse in [2] to provide a general test of extending standard encryption schemes for multiple users.

The function of our de-multiplexer is similar to re-encryption mix servers[7] in terms of computation model. In a re-encryption mix network, the mix server transforms a ciphertext into another one in order to provide source and destination anonymity against traffic analysis.

3 An Multiplex Encryption System

3.1 Architecture

The multiplex encryption system comprises three types of entities: a Certification Authority (CA), an encryption de-multiplexer, denoted by $\mathcal{DM}$, and a group of email users denoted by $U_1, U_2, \cdots, U_n$. The CA governs the other two types of participants and generates keys for $U_1, \cdots, U_n$. All the users are in the same email domain, which is served by $\mathcal{DM}$. Practically, a de-multiplexer can be a customized SMTP or POP3 server, where users retrieve their emails. Specifically, $\mathcal{DM}$ plays the role of both mail delivery agent and security mediator as in [5]. $\mathcal{DM}$ is *partially* trusted, in the sense that it is trusted to execute the protocol honestly and does not collude with any malicious users. We observe that establishing a *fully* trusted party would resolve the problem of multi-recipient encryption in a trivial fashion. However, a fully trusted party (TTP) is usually unrealistic or undesirable due to its high security risk, i.e. compromising TTP alone will expose all users' secrets. We stress that $\mathcal{DM}$ is not a TTP since a compromised or malicious $\mathcal{DM}$ is not able to decrypt any user's encrypted emails.

Similar to other public key encryption schemes, our construction has three algorithms KeyGen, Enc and Dec, as shown in following sections.

3.2 Key Generation

CA initializes the system parameters and generates RSA keys for all users.

First, CA chooses two 512-bit random safe primes p and q such that $p = 2p' + 1$ and $q = 2q' + 1$, where p', q' are also primes. It sets $N = pq$ and $e = 3$ or 65537. CA then selects a collision resistent and division intractable[8] hash function $\mathcal{H}()$, for instance SHA-256. $p, q, p'q'$ are kept secret while (N, e) and $\mathcal{H}$ are public.

For $1 \leq i \leq n$, user U_i's public key e_i is derived from its email address ID_i, exactly in the same fashion as in [5]. Specifically,

$$e_i = \mathcal{H}(ID_i) \| 0 \cdots 01$$

, where ”$\|$” denotes concatenation of two binary strings. Note that the least significant bit is set to 1 to ensure that e_i is odd and has overwhelming probability of being co-prime to $\phi(N)$. CA generates $d_{s,i}$ and $d_{u,i}$ such that $ee_i d_{s,i} d_{u,i} = 1 \bmod \phi(N)$. Then $d_{u,i}$ is securely delivered to U_i while $d_{s,i}$ is securely transferred to $\mathcal{DM}$. The details are shown in Figure 1.

Let $\mathcal{DM}$ choose for himself a public key *pk* and a private key *sk* for a public key scheme which is semantically secure under adaptive chosen ciphertext attack. $\mathcal{DM}$'s encryption setting is independent of the users' RSA setting. An encryption on message m using *pk* is denoted by $\mathcal{E}_{pk}(m)$ while a decryption on ciphertext c using *sk* is denoted by $\mathcal{D}_{sk}(c)$.

```
Algorithm KeyGen: Generating keys for $U_i$ (executed by CA).
Let $t$ be a public security parameter.

1. $s \leftarrow t - |\mathcal{H}()| - 1$
2. $e_i \leftarrow \mathcal{H}(ID_i)\|0^s\|1$, where $0^s$ denotes a $s$ bit binary string of 0-s.
3. $d_{s,i} \xleftarrow{r} \mathbb{Z}^*_{\phi(N)}$, i.e. $d_{s,i}$ is a random number in $\mathbb{Z}^*_{\phi(N)}$.
4. $d_{u,i} \leftarrow (ee_i d_{s,i})^{-1} \bmod \phi(N)$
5. $d_{u,i}$ and $d_{s,i}$ are securely distributed to $U_i$ and $\mathcal{DM}$ respectively.
```

Fig. 1. User Key Generation Algorithm

3.3 Multi-recipient Email Encryption

When an email is only delivered to one recipient, its encryption is exactly the same as in identity-based mRSA[5]. The sender derives the recipient's public key from his email address as in Figure 1; then encrypts the email using a normal RSA encryption defined in PKCS#1[12]. Note that the sender does not need to load the recipient's public key certificate by virtue of identity-based RSA encryption.

To encrypt a multi-recipient email, the sender executes the Algorithm Enc shown in Figure 2.

```
Algorithm Enc: Encrypting a multi-recipient mail $m$ for user $U_1, \cdots, U_k$
(executed by the sender)

1. Employ $(e, N)$ as an RSA public key and encrypt the mail using
   RSA with OAEP padding[4, 6] as defined in PKCS#1:
   $C \leftarrow \text{OAEP-ENC}(m)^e \bmod N$
2. Encrypt $C$ using $\mathcal{DM}$'s public key:
   $C' \leftarrow \mathcal{E}_{pk}(C\|ID_1\| \cdots \|ID_k)$.
3. Prepare S/MIME headers and send $C'$ to $\mathcal{DM}$ with $U_1, \cdots, U_k$ being on the carbon-copy list.
```

Fig. 2. Multi-recipient encryption algorithm

It is optional for U_i to sign the email using his private key, depending upon whether message integrity is in consideration. U_i specifies all intended recipients' email address in his carbon-copy list as well as in the ciphertext C'. One ciphertext is sent to all recipients.

3.4 Multi-recipient Email Decryption

When a recipients, U_j, comes to fetch the mail C', $\mathcal{DM}$ helps him to decrypt the message. The basic idea is to combine the concept of identity-based mediated RSA[5] and re-encryption mix server. $\mathcal{DM}$ first decrypts C' into C which is the RSA encryption of m under the public key (e, N). For user U_j, $1 \leq j \leq k$, $\mathcal{DM}$ translates C into the ciphertext corresponding to U_j without knowing m. The details of the decryption algorithm Dec for U_j and $\mathcal{DM}$ are shown in Figure 3.

```
Algorithm Dec: Decrypting a multi-recipient mail $C'$ for $U_1, \cdots U_k$
Mail Server:
  1. Decrypt $C'$ into $C$ by computing:
         $C \| ID_{r_1} \cdots \| ID_{r'_k} = \mathcal{D}_{sk}(C')$
  2. Construct a recipient set $\mathcal{R} = \{ID_{r_1}, \cdots, ID_{r'_k}\}$ for email $C'$. If $\mathcal{R}$
     does not match the carbon-copy list in the SMTP header, abort.
     Otherwise,
  3. On receiving $U_j$'s request for retrieving email $C'$, check if $ID_j \in \mathcal{R}$.
     If not, the request is rejected. Otherwise,
  4. Compute $U_j$'s public key $e_j$ by computing
     $e_j = \mathcal{H}(ID_j \| 0 \cdots 01)$
  5. Translate $C$ for $U_j$ by computing
     $\hat{C} = C^{e_j d_{s,j}} \bmod N$
  6. Send $< C', \hat{C}, ID_{r_1} \cdots \| ID_{r'_k} >$ to $U_j$.
User $U_j$:
  1. Decrypt $\hat{C}$ by computing
     $\hat{m} = \hat{C}^{d_{u,j}} \bmod N$
  2. Decode the OAEP encoding of $\hat{m}$ to get message $m$.
     $m \leftarrow$ OAEP-DEC$(m)$
```

Fig. 3. Multi-recipient decryption algorithm

Note that OAEP decoding is involved in U_j's step. $\mathcal{DM}$ should not, and actually is unable to, execute OAEP decoding since $\hat{C}$ is still a ciphertext.

In Figure 4, we present a conceptual comparison between our multiplex encryption protocol and S/MIME's approach to multi-recipient encryption. In our scheme, the sender U_i only sends one ciphertext to k recipients while in S/MIME's approach, the sender has to send a concatenation of k ciphertexts. In our approach, only two encryptions are needed, while in S/MIME approach, k public key encryptions are required.

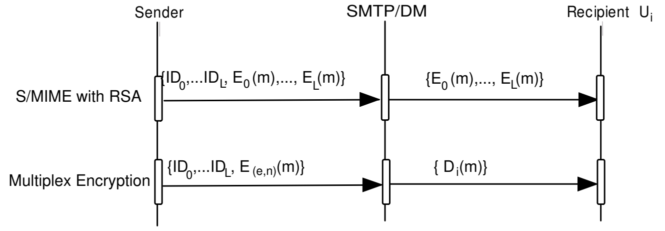

Sender
SMTP/DM
Recipient U_i
S/MIME with RSA
$\{ID_0,\ldots ID_L, E_0(m),\ldots, E_L(m)\}$
$\{E_0(m),\ldots, E_L(m)\}$
Multiplex Encryption
$\{ID_0,\ldots ID_L, E_{(e,n)}(m)\}$
$\{D_i(m)\}$

Fig. 4. Comparison Between Multiplex Encryption and Concatenated Encryptions

4 Security Discussion

Now we proceed to analyze the security of our multi-recipient encryption system. From the cryptography perspective, our construction is a variant of identity-based mediated RSA. In [5], the authors discussed several security issues, including the semantic security, the issue of public key generation, the issue of sharing RSA modulus etc. Their observations remain valid for our construction. Both their scheme and our encryption system share the same notion of semantic security. Besides those, we further discuss several security issues at the system level.

Malicious Users

For an email m intended for a set of users, $\mathcal{R} = \{U_0, \cdots, U_t\}$, we consider whether a malicious user Eve, $Eve \notin \mathcal{R}$, is able to retrieve m. Note that if Eve knows the ciphertext $C = m^e \bmod N$, she can trivially decrypt it by computing $C' = \mathcal{E}_{pk}(C \| Eve)$ and sending C' to $\mathcal{DM}$. However, Eve is unable to obtain C since C is encrypted under $\mathcal{DM}$'s public key during email forwarding. Eve can not successfully mount a replay attack either, because the identities of intended recipients are encrypted together with C using $\mathcal{E}_{pk}$. Note that $\mathcal{E}_{pk}(\cdot)$ is semantically secure under CCA2. According to [3], it is non-malleable under CCA2 as well. Therefore, she is unable to construct a ciphertext containing both her identity and message m without prior knowledge of m.

It is reasonable to assume that the channels between users and $\mathcal{DM}$ are authentic, as all email retrieval protocols require user authentication, e.g. asking for user id and password. Therefore, it is infeasible for Eve to impersonate another user in $\mathcal{R}$ to retrieve C. The message returned from $\mathcal{DM}$ in Figure 3 is partially decrypted. However, since d_s^i is a random number chosen from $\mathbb{Z}^*_{\phi(N)}$, Eve gets no information about m.

Malicious De-multiplexer

A malicious $\mathcal{DM}$ may attempt to compromise the secrecy of an encrypted multi-recipient email. $\mathcal{DM}$ only has knowledge of a set of random numbers $d_{s,0}, d_{s,1}, \cdots, d_{s,n}$, which is exactly the same as in ID-based mediated RSA[5]. Given a ciphertext C, $\mathcal{GM}$ is unable to obtain any information on m from $C^{d_{s,i}} \bmod N$ because he is not capable to distinguish the distribution of $c^{d_{s,i}} \bmod N$ and $c^r \bmod N$ where r is a random number. Therefore, the presence of a malicious $\mathcal{DM}$ alone would not place a threat to the semantic security of our encryption.

Collusion Between De-multiplexer And Users

As in [5], our construction is threatened by the collusion attack between a dishonest de-multiplexer and a malicious user. The collusion will destroy the whole system's security since they can collaboratively recover the factorization of N. In practice, an organization may alleviate the problem by further splitting $d_{s,i}$ to several parts or by deploying a proactive threshold RSA scheme so that it is more challenging for the adversary to compromise several servers at the same time.

Implication of Carbon Copy

In Section 1, we argue that S/MIME's approach does not preserve the consistency implication of carbon-copy since all recipients essentially access different ciphertexts.

With multiplex encryption, this feature of email carbon copy is saved because the recipients decrypt a common ciphertext, though in their own ways. It is in the same fashion as an email in plaintext sent to multiple recipients. When two recipients U_0, U_1 result in different messages m_0, m_1 for the same encrypted email C', the discrepancy can be resolved in a court following the steps below:

1. Given C', $\mathcal{DM}$ presents C and the randomness used in the encryption, which verifies that C is the plaintext of C' under $\mathcal{DM}$'s public key.
2. $\mathcal{DM}$ presents $\hat{C}_0 = C^{ed_{s,0}} \bmod N$ and $\hat{C}_1 = C^{ed_{s,1}} \bmod N$.
3. U_0 runs RSA decryption on $\hat{C}_0$ using their private keys and obtains the random seed s_0 used for OAEP. U_1 does the same on $\hat{C}_1$ and obtains his random seed s_1. Both s_0 and s_1 are presented to the arbitrator.
4. The arbitrator verifies whether C is the RSA-OAEP encryption of m_0 with random seed s_0 or of m_1 with seed s_1, under public key e. Note that only one of them is the valid plaintext and random seed pair.

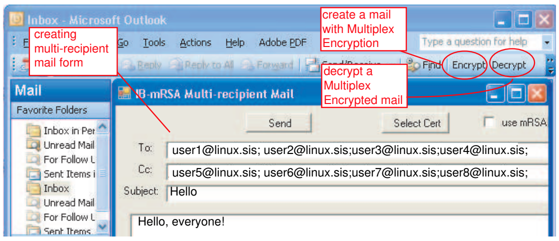
5 Performance

5.1 Implementation

To validate our algorithms and evaluate the performance, we implemented a prototype of the multiplex encryption system by using OpenSSL[10]. It includes

1. CA and Admin Utilities: This is for certificate issuance and revocation. It is implemented on a Linux platform.
2. De-multiplexer $\mathcal{DM}$: It includes the general functions of SMTP and POP3 protocols. It receives an encrypted email for multi-recipients. In email retrieval phase, it transforms the encrypted email properly for the requesting client. $\mathcal{DM}$ is implemented as a daemon running on a Linux host.
3. Outlook[11] plug-in: This is implemented as a Windows Dynamic Link Library(DLL) which provides encryption and decryption function for Outlook users.

A screen snapshot of the Outlook 2003 plug-in is shown in Fig. 5.



To: user1@linux.sis; user2@linux.sis;user3@linux.sis;user4@linux.sis;

Cc: user5@linux.sis; user6@linux.sis;user7@linux.sis;user8@linux.sis;

Subject: Hello

Hello, everyone!

Fig. 5. Outlook 2003 Plug-in

5.2 Performance Analysis

We conducted a series of experiments to evaluate the computation and communication cost of our scheme. $\mathcal{DM}$ daemon was running on a Linux PC with an Intel Pentium IV 2.00GHz processor. The client was on a Windows PC with 1.6GHz Centrino processor. Both of them had 1GByte memory. We ran two groups of experiments: one with 1024bit RSA modulus and the other with 2048bit RSA modulus. In each group of experiments, we measured the performance for 1 to 10 recipients, respectively.

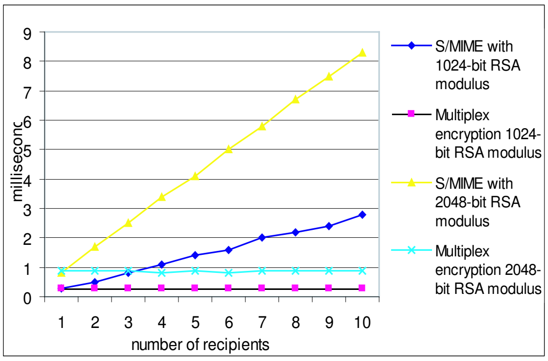


Fig. 6. Computation Cost for Multi-recipient Email Encryption

Figure 6 shows the RSA encryption time cost in milliseconds for the sender to encrypt an email for a number of recipients using two multi-recipient encryption approaches. The X axis indicates the number of recipients and the Y axis is for the encryption time. Not surprisingly, the two lines for S/MIME approach rise up linearly with respect to the number of recipients. In contrast, the encryption time using multiplex encryption almost remains unchanged though the number of recipients increases. It is straightforward to conclude from the protocol description in Figure 2 that the computation cost is independent of the number of recipients.

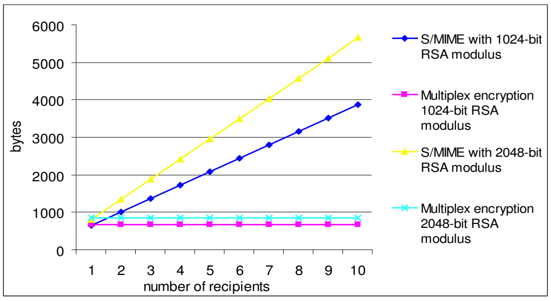


Fig. 7. Communication Cost for Encrypted Multi-recipient Emails

In Figure 7, we compare the communication cost for encrypted multi-recipient email delivery using two encryption approaches. We measure the cost in terms of the number of bytes to send by the sender. The X axis denotes the number of the recipients and the Y axis denotes the data length in bytes. Since the ciphertexts of each recipient are concatenated together in S/MIME approach, its data length grows linearly with the number of recipients, as shown by the two rising lines in Figure 7. In contrast, only one ciphertext is sent in multiplex encryption, where the length is determined by the RSA modulus used in encryption, and therefore is almost constant.

Nonetheless, the benefit of our scheme is at the cost of additional computation load at the both the server end and the recipients' sides. For every recipient, $\mathcal{DM}$ needs to perform one RSA decryption operation. Inevitably, a recipient performs one RSA decryption as well. However, since neither $\mathcal{DM}$ nor the recipient knows the factorization of N, they are not able to take the advantage of Chinese Remainder Theorem (CRT), which usually facilitates RSA decryption three times faster when used in standard RSA settings. The partial decryption time for $\mathcal{DM}$ and a recipient are shown in Table 1. We also list the standard RSA decryption time (with CRT) on the same hosts to demonstrate the additional cost.

RSA Modular size (bit)	$\mathcal{DM}$	Recipient	Standard RSA Decryption ($\mathcal{DM}$)	Standard RSA Decryption (User)
1024	13	13	5	4
2048	83	79	27	26

Table 1. Decryption Time Cost for $\mathcal{DM}$ and a Recipient (in ms)

We argue that the cost at both $\mathcal{DM}$ and the recipient sides is deserved. First, when the number of recipient is large, the benefit from the sender side compensates the cost. Second, users usually expect immediate email delivery. Minimizing the delivery delay keeps the user-friendliness of email application. On the other hand, the email retrieval takes place periodically and is not a real-time application. Therefore, slightly longer delay in retrieval does not affect the recipients. Moreover, to improve the performance, the partial decryption at $\mathcal{DM}$ end can be computed during idle time, instead of being triggered by the recipient's request.

6 Summary

Protecting the privacy of multi-recipient emails is not trivial as it seems. The approach proposed by S/MIME suffers from poor scalability due to its linear computation and communication cost. In this paper, we propose a new practical and secure scheme, called *multiplex encryption*, by combining the ideas of identity-based mediated RSA and re-encryption mixnet. Our framework only requires constant number of RSA encryption operations and constant ciphertext length for multi-recipient email encryption. We also implemented a prototype for experiment purpose.

Our scheme has a few drawbacks. First, it introduces a partially trusted third party. Though compatible with the email application architecture, deploying of our scheme still requires changes on email servers. Moreover, it limits all users to be in the same email domain. Secondly, the security relies on an assumption that $\mathcal{DM}$ does not collude with any users. For a large, heterogenous organization, such assumption might not hold. Our future work will investigate the feasibility of a new encryption paradigm without the involvement of a third party.

References

1. M. Bellare, A. Boldyreva and S. Micali: Public-key encryption in a multi-user setting: security proofs and improvements. In Eurocrypto'2000.
2. M. Bellare, A. Boldyreva and J. Staddon: Multi-recipient encryption schemes: security notions and randomness re-use. In PKC'2003.
3. M. Bellare, A. Sahai: Non-Malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-based Charaterization. In Crypto'99
4. M. Bellare and P. Rogaway: Optimal asymmetric encryption – how to encrypt with RSA.In A.D. Santis, editor, *Advances in Cryptology - EUROCRYPT '94*
5. X. Ding, G. Tsudik: Simple identity-based cryptography with mediated RSA. In 2003 RSA Conference, *Cryptographers' Track*, April 2003.
6. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is secure under the rsa assumption. In *Advances in Cryptology - CRYPTO '2001*
7. P. Golle, M. Jakobsson, Ari Juels and P. Syverson: Universarl Re-encryption for Mixnets. In *CT-RSA 2004*
8. R. Gennaro, S. Halevi, and T. Rabin: Secure hash-and-sign signatures without the random oracle.In J. Stern, editor, *Advances in Cryptology – EUROCRYPT '99*
9. K. Kurosawa: Multi-recipient Public-Key Encryption with Shortened Ciphertext. In *5th International Workshop on Practice and Theory in Public Key Cryptosystems*, PKC 2002.
10. OpenSSL User Group. The OpenSSL Project Web Page, http://www.openssl.org/
11. Microsoft.Microsoft Outlook©, http://www.microsoft.com
12. PKCS#1 v2.1: RSA Cryptography Standard. RSA Laboratories, June 2002
13. PKCS#12: Personal information exchange syntax. RSA Laboratories, 1999. Version 1.0.
14. S/MIME: Secure/Multipurpose Internet Mail Extensions. The S/MIME Working Group: http://www.ietf.org/html.charters/smime-charter.html .