Simple Identity-Based Cryptography with Mediated RSA ⋆

Xuhua Ding and Gene Tsudik

Department of Information and Computer Science, University of California, Irvine.
Email: {xhding,gts}@ics.uci.edu

Abstract. Identity-based public key encryption facilitates easy introduction of public key cryptography by allowing an entity's public key to be derived from an arbitrary identification value, such as name or email address. The main practical benefit of identity-based cryptography is in greatly reducing the need for, and reliance on, public key certificates. Although some interesting identity-based techniques have been developed in the past, none are compatible with popular public key encryption algorithms (such as El Gamal and RSA). This limits the utility of identity-based cryptography as a transitional step to full-blown public key cryptography. Furthermore, it is fundamentally difficult to reconcile fine-grained revocation with identity-based cryptography.
Mediated RSA (mRSA) [9] is a simple and practical method of splitting a RSA private key between the user and a Security Mediator (SEM). Neither the user nor the SEM can cheat one another since each cryptographic operation (signature or decryption) involves both parties. mRSA allows fast and fine-grained control of users' security privileges. However, mRSA still relies on conventional public key certificates to store and communicate public keys. In this paper, we present IB-mRSA, a simple variant of mRSA that combines identity-based and mediated cryptography. Under the random oracle model, IB-mRSA with OAEP[7] is shown as secure (against adaptive chosen ciphertext attack) as standard RSA with OAEP. Furthermore, IB-mRSA is simple, practical, and compatible with current public key infrastructures.
Keywords: Identity-based Encryption, Mediated RSA, Revocation

1 Introduction

In a typical public key infrastructure (PKI) setting, a user's public key is explicitly encoded in a public key certificate which is, essentially, a binding between the certificate holder's identity and the claimed public key. This common model requires universal trust in certificate issuers (Certification Authorities or CAs). It has some well-known and bothersome side-effects such as the need for cross-domain trust and certificate revocation. The main problem, however, is the basic assumption that all certificates are public, ubiquitous and, hence, readily available to anyone. We observe that this assumption is not always realistic, especially, in wireless (or any fault-prone) networks where connectivity is sporadic.

⋆ This work was supported by DARPA contract F30602-99-1-0530.

In contrast, identity-based cryptography changes the nature of obtaining public keys by constructing a one-to-one mapping between identities and public keys. Identity-based cryptography thus greatly reduces the need for, and reliance on, public key certificates and certification authorities. In general, identity-based encryption and identity-based signatures are useful cryptographic tools that facilitate easy introduction of, and/or conversion to, public key cryptography by allowing a public key to be derived from arbitrary identification values such as email addresses or phone numbers. At the same time, identity-based methods greatly simplify key management since they reduce both: the need for, and, the number of, public key certificates.

The concept of identity-based public encryption was first proposed by Shamir[20] in 1984. For the following 16 years the progress in this area has been rather slow. However, recently, Boneh and Franklin developed an elegant Identity-Based Encryption system (BF-IBE) based on Weil Pairing on elliptic curves [10]. BF-IBE represents a significant advance in cryptography.

Nevertheless, an identity-based RSA variant has remained elusive for the simple reason that an RSA modulus n (a product of two large primes) can not be safely shared among multiple users. Another notable drawback of current identity-based cryptographic methods is lack of support for fine-grained revocation. Revocation is typically done via Certificate Revocation Lists (CRLs) or similar structures. However, IBE aims to simplify certificate management by deriving public keys from identities, which makes it difficult to control users' security privileges.

In this paper, we propose a simple identity-based cryptosystem developed atop some Mediated RSA (mRSA) by Boneh, et al. [9]. mRSA is a practical and RSA-compatible method of splitting an RSA private key between the user and the security mediator, called a SEM. Neither the user nor the SEM knows the factorization of the RSA modulus and neither can decrypt/sign message without the other's help. By virtue of requiring the user to contact its SEM for each decryption and/or signature operation, mRSA provides fast and fine-grained revocation of users' security privileges.

Built on top of mRSA, IB-mRSA blends the features of identity-based and mediated cryptography and also offers some practical benefits.[1] Like mRSA, it is fully compatible with plain RSA. With the exception of the identity-to-public-key mapping, it requires no special software for communicating parties. IB-mRSA also allows optional public key certificates which facilitates easy transition to a conventional PKI. More generally, IB-mRSA can be viewed as a simple and practical technique inter-operable with common modern PKIs. At the same time, IB-mRSA offers security comparable to that of RSA, provided that a SEM is not compromised. Specifically, it can be shown that, in the random oracle model, IB-mRSA with OAEP[7] is as secure – against adaptive chosen ciphertext attacks – as RSA with OAEP.

The rest of the paper is organized as follows. The next section gives a detailed description of IB-mRSA. The security analysis is presented in Section 3 and the performance analysis – in Section 4. In Section 5, IB-mRSA is compared with Boneh-Franklin's IBE. Finally, a brief description of the implementation is presented in the last section. Some further security details can be found in the Appendix.

[1] A very sketchy version of IB-mRSA was first presented in [8].

2 Identity-Based mRSA

The main feature of identity-based encryption is the sender's ability to encrypt messages using the public key derived from the receiver's identity and other public information. The identity can be the receiver's email address, user id or any value unique to the receiver; essentially, an arbitrary string. To compute the encryption key, an efficient (and public) mapping function $\mathcal{KG}$ must be set beforehand. This function must be a one-to-one mapping from identity strings to public keys.

The basic idea behind identity-based mRSA is the use of a single common RSA modulus n for all users within a system (or domain). This modulus is public and contained in a system-wide certificate issued, as usual, by some Certificate Authority (CA). To encrypt a message for a certain recipient (Bob), the sender (Alice) first computes $e_{Bob} = \mathcal{KG}(ID_{Bob})$ where ID_{Bob} is the recipient's identity value, such as Bob's email address. Thereafter, the pair (e_{Bob}, n) is treated as a plain RSA public key and normal RSA encryption is performed. On Bob's side, the decryption process is identical to that of mRSA.

We stress that using the same modulus by multiple users in a normal RSA setting is **utterly insecure**. It is subject to a trivial attack whereby anyone – utilizing one's knowledge of a single key-pair – can simply factor the modulus and compute the other user's private key. However, in the present context, we make an important assumption that: *Throughout the lifetime of the system, the adversary is unable to compromise a SEM.*

Obviously, without this assumption, IB-mRSA would offer no security whatsoever: a single SEM break-in coupled with the compromise of just one user's key share would result in the compromise of all users' (for that SEM) private keys. The IB-mRSA assumption is slightly stronger than its mRSA counterpart. Recall that, in mRSA, each user has a different RSA setting, i.e., a unique modulus. Therefore, to compromise a given user an adversary has to break into both the user and its SEM.

We now turn to the detailed description of the IB-mRSA scheme.

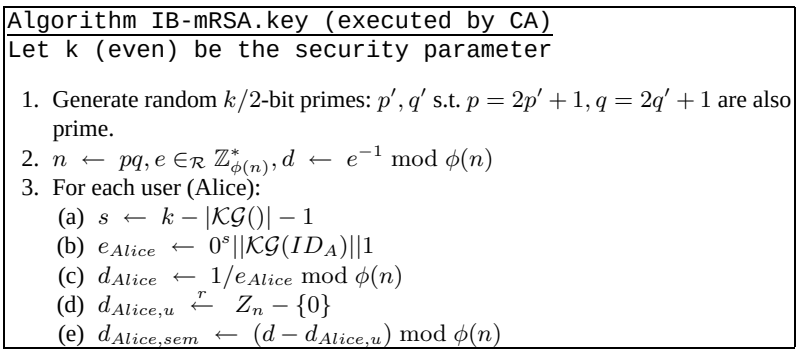
2.1 System Setting and User Key Generation

In the following, we use email addresses as unique identifiers of the public key owners in the system. However, as mentioned above, other identity types can be used just as well, e.g., Unix UIDs, HTTP addresses, physical addresses or even phone numbers. We use the notation ID_{Alice} to denote the user's (Alice) email address that will be used to derive the public exponent.

In the initialization phase, a trusted party (CA) sets up the RSA modulus for all users in the same system (organization or domain). First, CA chooses, at random, two large primes p' and q' such that $p = 2p' + 1$ and $q = 2q' + 1$ are also primes, and finally sets $n = pq$. We note that, since n is a product of two strong primes, a randomly chosen odd number in Z_n has negligible probability of not being relatively prime to $\phi(n)$. (See Section 3 for further discussion.) Hence, the mapping function $\mathcal{KG}$ can be quite trivial. (Our current implementation uses the popular $MD5$ hash function.)

The public exponent e_{Alice} is constructed as the output of $\mathcal{KG}(ID_Alice)$ represented as a binary string of the same length as the modulus, with the least significant bit

set. This ensures that e_{Alice} is odd and, with overwhelming probability, relatively prime to $\phi(n)$. The complete IB-mRSA key generation proceeds as in Figure 1.



Algorithm IB-mRSA.key (executed by CA)

Let k (even) be the security parameter

1. Generate random $k/2$-bit primes: p', q' s.t. $p = 2p' + 1, q = 2q' + 1$ are also prime.
2. $n \leftarrow pq, e \in_{\mathcal{R}} \mathbb{Z}^*_{\phi(n)}, d \leftarrow e^{-1} \bmod \phi(n)$
3. For each user (Alice):
   (a) $s \leftarrow k - |\mathcal{KG}()| - 1$
   (b) $e_{Alice} \leftarrow 0^s || \mathcal{KG}(ID_A) || 1$
   (c) $d_{Alice} \leftarrow 1/e_{Alice} \bmod \phi(n)$
   (d) $d_{Alice,u} \stackrel{r}{\leftarrow} Z_n - \{0\}$
   (e) $d_{Alice,sem} \leftarrow (d - d_{Alice,u}) \bmod \phi(n)$

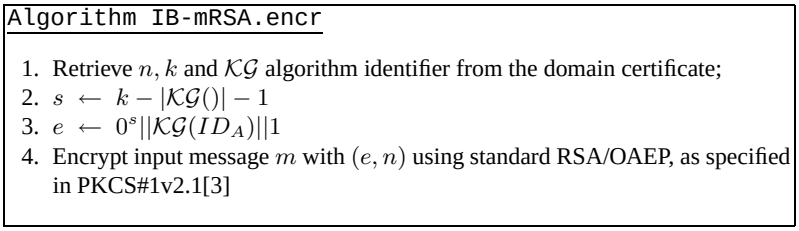
Fig. 1. IB-mRSA: User Key Generation

A domain- or system-wide certificate ($Cert_{org}$) is issued by the CA after completion of the key generation algorithm. This certificate contains almost all the usual fields normally found in RSA public key certificates with few exceptions, such as no real public key value is given. In particular, it mainly contains the common modulus n and (if applicable) the common part of the email address for all users, such as the domain name.

For the sake of compatibility with other (not identity-based) RSA implementations – including plain RSA and mRSA – the CA may, upon request, issue an individual certificate to a user. In most cases, however, an individual user certificate would not be needed, since not having such certificates is exactly the purpose of identity-based cryptography.

2.2 IB-mRSA Encryption

To encrypt a message, the sender needs only the recipient's email address and the domain certificate. The encryption algorithm is shown in Figure 2.



Algorithm IB-mRSA.encr

1. Retrieve n, k and $\mathcal{KG}$ algorithm identifier from the domain certificate;
2. $s \leftarrow k - |\mathcal{KG}()| - 1$
3. $e \leftarrow 0^s || \mathcal{KG}(ID_A) || 1$
4. Encrypt input message m with (e, n) using standard RSA/OAEP, as specified in PKCS#1v2.1[3]

Fig. 2. IB-mRSA: Encryption

Since the receiver's public key is derived from the receiver's unique identifier, the sender does not need a public key certificate to ensure that the intended receiver is the correct public key holder. Furthermore, fast revocation provided by mRSA obviates the need for the sender to perform any revocation checks. The decryption process is essentially the same as in mRSA. If a certain user needs to be revoked, the domain security administrator merely notifies the appropriate SEM and the revoked user is unable to decrypt any further messages.

2.3 IB-mRSA Decryption

IB-mRSA decryption is identical to that of mRSA. To make this paper self-contained, we borrow (from [9]) the protocol description in Figure 3. For a detailed description and security analysis of additive mRSA, we refer the reader to [9].[2]

Protocol IB-mRSA.decr (executed by User and SEM)

1. USER: $m' \leftarrow$ encrypted message
2. USER: send m' to SEM
3. In parallel:
   3.1 SEM:
   (a) If USER revoked return (ERROR)
   (b) $PD_{sem} \leftarrow m'^{d_{sem}} \bmod n$
   (c) Send PD_{sem} to USER
   3.2 USER:
   (a) $PD_u \leftarrow m'^{d_u} \bmod n$
4. USER: $M \leftarrow (PD_{sem} * PD_u) \bmod n$
5. USER: $m \leftarrow$ OAEP Decoding of M
6. USER: If succeed, return (m)

Fig. 3. IB-mRSA: Decryption

3 Security of Identity-based mRSA

We now examine the security of IB-mRSA/OAEP in a setting with n users. All users share a common RSA modulus N and each user (U_i) is associated with a unique identity ID_i, which is mapped into an RSA public exponent e_i via a mapping function $\mathcal{KG}$.

3.1 Security Analysis

In the following, we argue that if $\mathcal{KG}$ is an appropriate hash function, IB-mRSA/OAEP is semantically secure against adaptive chosen ciphertext attacks (CCA-2) in the random oracle model. We use the term *indistinguishability* which is a notion equivalent to semantic security. (See [6] for the relevant discussion.)

[2] There is also a very similar multiplicative mRSA (*mRSA) first proposed by Ganesan [13].

Our analysis is mainly derived from the results in [5], ([4] has similar results) where it was shown that a public-key encryption system in a multi-user setting is semantically secure against certain types of attacks if and only if the same system in a single-user setting is semantically secure against the same attack types.

IB-mRSA/OAEP is obviously an encryption setting with many users, although they do not physically possess their own private keys. To prove semantic security, we begin by asserting that IB-mRSA in single-user mode is equivalent to the standard RSA/OAEP, which is proven secure against CCA-2 under the random oracle [12]. Next, we apply the theorems in [5] with the condition that all users are honest. To remove this condition, we analyze the distribution of views of the system from users and outside adversaries. Furthermore we introduce an additional requirement for the key generation function (division-intractability) so that we can neglect the possibility of an attack from legitimate (inside) users, which is a problem unique to our setting. In the end, we argue for semantic security of IB-mRSA/OAEP.

We use $Succ_1^{IB}(t, q_d)$ to denote the maximum advantage of all adversary algorithms in polynomial time t, attacking IB-mRSA/OAEP with one user, $Succ_n^{IB}(t, q_d, q_e)$ for the setting with n users, and $Succ^R(t, q_d)$ for RSA/OAEP. In the above, q_d(q_e) denote the maximum number of decryption (encryption) queries allowed for each public key. Throughout the analysis, we consider semantic security against CCA-2 under the random oracle assumption. To conserve space, we omit mentioning them in the following discussion.

We begin with the following lemma.

Lemma 1. *IB-mRSA/OAEP system in a single-user setting is polynomially as secure as standard RSA/OAEP encryption, i.e.,*

$$Succ_1^{IB}(t, q_d) = Succ^R(t', q_d)$$

where c is constant value, $t' = t + c$.

The proof is in Appendix A.2. Basically, if there exists an algorithm breaking the security of IB-mRSA/OAEP in a single-user mode, we can build upon it an algorithm breaking standard RSA/OAEP with the same success probability and constant extra overhead. Of course, it is easy to see that breaking RSA/OAEP implies breaking IB-mRSA. Thus, we claim that they are equally secure.

For the multi-user setting, we cannot claim that IB-mRSA with n users is semantically secure by directly applying the security reduction theorem in [5]. The reason is that our system is not a typical case referred in [5]. Sharing a common RSA modulus among many users results in their respective trapdoors not being independent; consequently, there could be attacks among the users. Furthermore, users in IB-mRSA may have the incentive not only to attack other users, but also to attempt to break the underlying protocol so that they can bypass the mandatory security control of the SEM.

However, assuming for the moment, that all users are honest, we can obtain the following lemma derived from [5].

Lemma 2. *IB-mRSA/OAEP system with n users is semantically secure if all n are honest. More precisely,*

$$Succ_n^{IB}(t_n, q_d, q_e) \leq q_e n Succ_1^{IB}(t_1, q_d)$$

where $t_1 = t_n + O(\log(q_e n))$

When all users are honest, there are clearly no attacks. Thus, IB-mRSA with multi-user can be considered as an example of encryption system in [5] where each user has an independent trapdoor. We adapt the original proof in [5] in order to claim security against CCA-2 since no user actually knows its own trapdoor in IB-mRSA. See Appendix A.3 for details.

Unfortunately, in a real application, all users cannot be assumed to be trusted. To remove this condition in Lemma 2, we have to examine both the information an inside user can observe and the operations an inside user can perform.

For a given entity (user or set of users) we use an informal term "system view" to refer to the distribution of all inputs, outputs, local state information as well as scripts of interactions with decryption oracles, encryption oracles, and the SEM. The system view for an outside attacker is denoted as:

$$V_1 ::= Pr\{N, (e_0, \ldots e_n), \Gamma_O, \Gamma_E, \Gamma_D, \Gamma_{SEM}\}$$

while the system view for a set of users is:

$$V_2 ::= Pr\{N, (e_0, \ldots, e_n), \{d_{u_i}\}, \Gamma_O, \Gamma_E, \Gamma_D, \Gamma_{SEM}, \Gamma_{d_u,n}\}$$

where $\{d_{u_i}\}$ is the set of user key-shares; $\Gamma_O, \Gamma_E, \Gamma_D$ are three scripts recording all queries/answers to the random oracle, encryption oracles and decryption oracles, respectively; Γ_{SEM} is the script recording all requests/replies between all users and the SEM; $\Gamma_{d_u,n}$ is the script recording all n users' computation on ciphertexts with their own secret key-share d_{u_i}. We claim in Lemma 3, that being an IB-mRSA user does not afford one extra useful information as compared to an outside adversary.

Lemma 3. *Under the adaptive chosen ciphertext attack, the system view of the outside adversary (V_1), is polynomially indistinguishable from the combined system view (V_2) of a set of malicious insiders, in the random oracle model.*

Proof. See Appendix A.4 for details. □

Thus far, we have shown that insider adversaries do not gain advantages over outsiders in terms of obtaining extra information. However, we also need to consider the privileged operations that an insider can make. In IB-mRSA, each user is allowed to send legitimate decryption queries to its SEM. In conventional proofs, the adversary is not allowed to query its oracle for the challenge ciphertext. However in our case, an inside adversary can manipulate a challenge ciphertext (intended for decryption with d_i) into another ciphertext that can be decrypted with its own key d_j and legally decrypt it with the aid of the SEM.[3]

[3] A simple example is as follows. Suppose $e_i = 3 * e_j$. Then, given $c = m^{e_j} \bmod n$, User U_i can compute $c' = c^3 = m^{e_i} \bmod n$, which can be decrypted by U_i with the help from its SEM. The notion of non-malleability does not capture this attack since it is defined under a single fixed private/public key pair.

We now have to consider the probability of such attacks in our setting. More generally, let $e_{a_0}, \ldots, e_{a_v}$ be the set of public keys of v malicious users, and $E_v = \prod_{a_i} e_{a_i}$. They may attempt to use some function f, which takes a challenge $c = m^x \bmod n$ as input and outputs ciphertext $c' = m^{E_v}$. We offer the following lemma to address the conditions for the existence of such f.

Lemma 4. *Given two RSA exponents x, y and modulus n, let f be a polynomial time complexity function s.t. $f(m^x) = m^y \bmod n$. Such f exists iff $x|y$.*

Proof. See Appendix A.5 for details. □

According to Lemma 4, we require negligible probability of obtaining a user's public key which is a factor of the product of a set of others. A similar requirement appears in a signature scheme by Gennaro et al. in [14]. They introduce the notion of division intractability for a hash function. Informally, a hash function H is **Division intractable** if it is infeasible to find distinct $(X_1, \ldots, X_n, Y)$ in its domain, such that $H(Y)|\prod_i(H(X_i))$. Denoting $Pr^{div}(H)$ as the probability that H fails to hold this property, we have the following proposition regarding the security of IB-mRSA in a multi-user setting.

Proposition 1. *IB-mRSA/OAEP encryption offers equivalent semantic security to RSA/OAEP against adaptive chosen ciphertext attacks in the random oracle model, if the key generation function is division intractable.*

In summary, Lemma 3 and Lemma 4 enable us to remove the condition of Lemma 2 where all users are assumed to be honest, by requiring the key generation function to be division intractable. Thus, we can reduce the security of IB-mRSA/OAEP in multi-user setting into single-user, which is as secure as standard RSA/OAEP according to Lemma 1.

3.2 The Public Key Mapping Function

The key generation function $\mathcal{KG}$ in IB-mRSA is a hash function H. To ensure the security of the scheme, H must satisfy the following requirements.

AVAILABILITY OF PUBLIC KEYS: The output of H should have an overwhelming probability of being relatively prime to $\phi(n)$. Obviously, for the inverse (private key) to exist, a public exponent can not have common factors with $\phi(n)$.

Note that in Section 2 the RSA modulus n is set to $n = p * q$ and p, q are chosen as strong primes $p = 2p' + 1$, $q = 2q' + 1$ where both p' and q' are also large primes. Considering $\phi(n) = 2^2 p'q'$ with only three factors 2, p, q, the probability of the output from H being co-prime to $\phi(n)$ is overwhelming on the condition that the output is an odd number, because finding an odder number not co-prime to $4p'q'$ is equivalent to find p' or q' and consequently factoring n.

COLLISION RESISTANCE: H should be a collision-resistant function, i.e., given any two distinct inputs ID_1, ID_2, the probability of $H(ID_1) = H(ID_2)$ should be negligible. In other words, no two users in the domain can share the same public exponent.

DIVISION RESISTANCE: As discussed in Section 3.1, division intractability of H is essential to the security of IB-mRSA. Gennaro et al. analyzed the probability of division for hash functions in [14].

Moreover, Coron and Naccache showed in [11] that the number of necessary hash-value to find a division relation among a hash function's outputs is sub-exponential to its digest size k: $exp(\sqrt{2\log 2}/2 + o(1)\sqrt{k \log k})$.

They suggested using 1024-bit hash functions to get a security level equivalent to 1024-bits RSA. However, such a strong hash function is not needed in our case. As a point of comparison, the GHR signature scheme [14] needs a division-intractable hash function to compute message digests, where an adaptive adversary can select any number of inputs to the underlying hash function. IB-mRSA needs a hash function to compute digests from users' identities. In any domain, the number of allowed identities is certainly much fewer compared to the number of messages in [14].

digest size in bits	$\log_2$ complexity (in # of operations)
128	36
160	39
192	42
1024	86

Table 1. Estimated complexity of the attack for variable digest sizes.

To help select the best hash size for our purposes, we quote from the experiments by Coron and Naccache [11] in Table 1. Taking the first line as an example, an interpretation of the data is that, among at least 2^{36} hash digests, the probability of finding one hash value dividing another is non-negligible. In IB-mRSA setting, the typical personnel of an organization is on the order of $2^{10} \sim 2^{17}$. Consequently, the possible number of operations is far less than 2^{36}. Hence, we can safely use MD5 or SHA-1 as the mapping function (H).

3.3 SEM Security

Suppose that the attacker is able to compromise the SEM and expose the secret key d_{sem}, however, without collusion with any user. This only enables the attacker to "un-revoke" previously revoked, or block possible future revocation of currently valid, certificates. The knowledge of d_{sem} does not enable the attacker to decrypt or sign messages on behalf of the users. The reason is obvious: note that Alice never sends her partial results to her SEM. Thus, the attacker's view of Alice can be simulated in the normal RSA setting, where the attacker just picks a random number as d_{sem} and make computations on the ciphertext, messages to sign and signatures generated by Alice.

3.4 Security of Common Modulus

As mentioned earlier, using a common RSA modulus is clearly unacceptable in plain RSA setting. In the mediated RSA architecture, sharing a modulus is feasible since no

party knows a complete private/public key-pair. In fact, no coalition of users is able to compute a public/private key-pair. The only way to "break" the system appears to be by subverting a SEM and colluding with a user. Thus, in the context of IB-mRSA we need to assume that a SEM is a **fully** trusted party, as opposed to **semi-trusted** in mRSA [9].

4 Performance Analysis

When plain RSA is used for encryption, the public encryption exponent e is typically a small integer with only a few 1-bits. One example is the popular OpenSSL toolkit [17] which uses $65,537$ as the default public key value for RSA certificates. Encryption with such small exponents can be accelerated with specialized algorithms for modular exponentiation. However, in IB-mRSA setting, there is no such luxury of choosing special exponents and a typical public exponent is a relatively large integer with (on the average) half of the bits set to 1.

Keys	RSA Modulus 1Kb	RSA Modulus 2Kb	RSA Modulus 4Kb
$65,537$	2 ms	4 ms	12 ms
128-bit key	7 ms	20 ms	69 ms
160-bit key	8 ms	25 ms	88 ms

Table 2. IB-mRSA Encryption: Performance Comparison of Different Encryption Keys.

We ran some simple tests to assess the cost of IB-mRSA encryption for public keys derived from email addresses. The encryption was tested using OpenSSL on an 800MHz PIII workstation. In the tests, we used: 1) "default" encryption exponent $65,537$ and 2) two other exponents of length 128-bit and 160-bit. For each key, we randomly set half of the bits. The results are depicted in Table 2.

From the results in Table 2, we see that encryption with a randomized key does introduce overhead, especially when the RSA modulus size grows. However, it is rather negligible for the 1024-bit case, which is currently the most popular modulus size.

The decryption cost for IB-mRSA is identical to mRSA. The performance of mRSA has been reported on by Boneh, et al. in [9]. For example, a 1024-bit mRSA decryption costs around 35ms on an 800 MHz PIII, as compared to 7.5ms for plain RSA on the same platform. We note that this is still much cheaper than 40ms that is needed for Boneh/Franklin IBE decryption (for 1024 bits of security on a even more powerful hardware platform).

5 IB-mRSA versus Boneh/Franklin IBE

We now provide a detailed comparison of BF-IBE and IB-mRSA. The comparison is done along several aspects, including: practicality, revocation, security and cost of key generation.

Practicality and Performance: Although BF-IBE and IB-mRSA have similar architectures, the underlying cryptographic primitives are completely different. Compared to the elliptic curve primitives used in BF-IBE, IB-mRSA is much easier to deploy since RSA is currently the most popular public key encryption method. Recall that IB-mRSA is fully compatible with standard RSA encryption. Moreover, if optional individual certificates are used, IB-mRSA is fully compatible with current PKI-s. Thus, it offers a smooth and natural transition from normal ID-based to public key cryptography.

In addition, IB-mRSA offers better performance than BF-IBE. As seen from the comparison in Table 3, IB-mRSA is noticeably faster than BF-IBE in both key generation and message encryption.

	BF-IBE	IB-mRSA
Private Key Generation	3ms	$<$ 1ms
Encryption Time	40ms	7ms
Decryption Time	40ms	35ms

Table 3. Performance Comparison of BF-IBE (on PIII 1GHz) and IB-mRSA (on PIII 800MHz) with 1024-bit security.

Revocation: BF-IBE does not explicitly provide revocation of users' security capabilities. This is natural since it aims to avoid the use of certificates in the course of public key encryption. On the other hand, revocation is often necessary and even imperative.

The only way to obtain revocation in normal IBE is to require fine-grained time-dependent public keys, e.g., public keys derived from identifiers combined with time- or date-stamps. This has an unfortunate consequence of having to periodically re-issue all private keys in the system. Moreover, these keys must be (again, periodically) securely distributed to individual users. In contrast, IB-mRSA inherits its fine-grained revocation functionality from mRSA [9]. IB-mRSA provides per-operation revocation, whereas, BF-IBE provides periodic revocation, which clearly has coarser granularity. Essentially, IB-mRSA allows revocation to commence at any time while BF-IBE revokes users by refusing to issue new private keys. However, BF-IBE does not prevent the type of an attack whereby an adversary who compromises a previous or current key can use them to decrypt previously encrypted messages. This can be a serious attack in some settings, such as military applications.

Trusted Third Parties: Both SEM in IB-mRSA and PKG in BF-IBE are trusted third parties. However, the difference in the degree of trust is subtle. A SEM is fully trusted since its collusion with any user can result in a compromise of all other users' secret keys, due to the shared RSA modulus. Nonetheless, a compromise of a SEM alone does not result in a compromise of any users' secret keys. A PKG is a real TTP since it knows all users' secrets, thus, a compromise of a PKG results in a total system break. While a PKG can also be a CA at the same time, a SEM can never be allowed to play the role of CA.

If BF-IBE is used to provide fine-grained revocation, frequent key generation and secure key distribution are expensive procedures. Although a PKG is not required to be on-line all of the time, in practice, it must be constantly available since users do not all request their current private keys at the same time. Therefore, as the revocation interval in BF-IBE gets smaller, the on-line presence of a PKG becomes more necessary.

6 Implementation

We implemented IB-mRSA for the purposes of experimentation and validation. The implementation is publicly available at `http://sconce.ics.uci.edu/sucses`. The software is composed of three parts:

1. CA and Admin Utilities: domain certificate, user key generation, (optional) certificate issuance and revocation interface.
2. SEM daemon: SEM process as described in Section 2
3. Client libraries: IB-mRSA user functions accessible via an API.

The code is built on top of the popular OpenSSL [17] library. OpenSSL incorporates a multitude of cryptographic functions and large-number arithmetic primitives. In addition to being efficient and available on many common hardware and software platforms, OpenSSL adheres to the common PKCS standards and is in the public domain. The SEM daemon and the CA/Admin utilities are implemented on Linux, while the client libraries are available on both Linux and Windows platforms.

In the initialization phase, a CA initializes the domain-wide cryptographic setting, namely (n, p, q, p', q') and selects a mapping function (currently defaulting to MD5) for all domain clients. The set up process follows the description in Section 2. For each user, two structures are exported: 1) SEM *bundle*, which includes the SEM's half-key d_i^{SEM}, and 2) user *bundle*, which includes d_i^u and the entire server bundle.

The server bundle is in PKCS#7[1] format, which is basically a RSA envelope signed by the CA and encrypted with the SEM's public key. The client bundle is in PKCS#12[2] format, which is a shared-key envelope also signed by the CA and encrypted with the user-supplied key which can be a pre-set key, a password or a passphrase. (A user is not assumed to have a pre-existing public key.)

After issuance, each user bundle is distributed in an out-of-band fashion to the appropriate user. Before attempting any IB-mRSA transactions, the user must first decrypt and verify the bundle. A separate utility program is provided for this purpose. With it, the bundle is decrypted with the user-supplied key, the CA's signature is verified, and, finally, the user's half-key are extracted and stored locally.

To decrypt a message, the user starts with sending an IB-mRSA request, with the SEM bundle piggybacked. The SEM first check the status of the client. Only when the client is deemed to be a legitimate user, does the SEM process the request using the bundle contained therein. As mentioned earlier, in order to encrypt a message for an IB-mRSA, that user's domain certificate needs to be obtained. Distribution and management of domain certificates is assumed to be done in a manner similar to that of normal certificate, e.g., via LDAP or DNS.

6.1 Emailer client plug-in

To demonstrate the ease of using IB-mRSA we implemented plug-ins for the popular Eudora[19] and Outlook[16] mailers. The plug-ins allow the sender to encrypt outgoing emails to any client in the common domain using only one domain (organizational) certificate. When ready to send, the sender's plug-in reads the recipient's email address and looks up the organization certificate by using the domain name in the email address. A screen snapshot of the Eudora plug-in is shown in Figure 4.

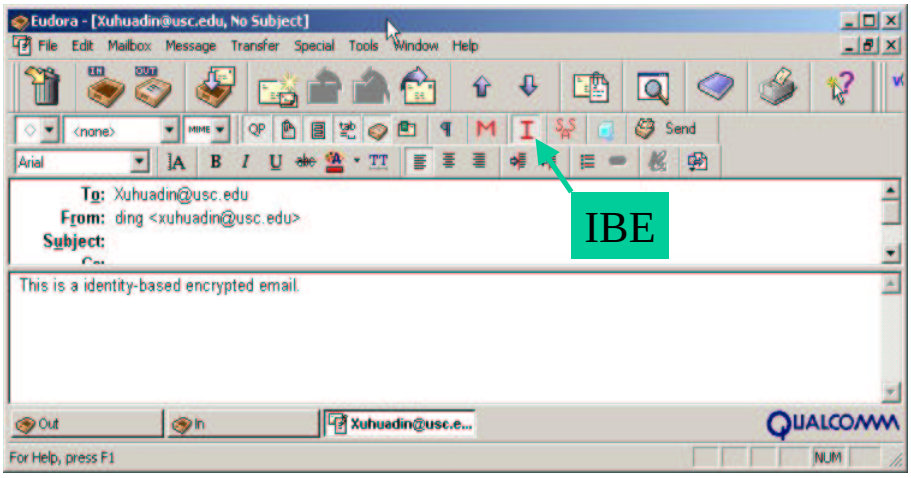


Fig. 4. Eudora IBE Plugin

When an email message encrypted with IB-mRSA is received, an icon for IB-mRSA is displayed in the message window. To decrypt the message, the user just clicks on the IB-mRSA icon. The plug-in then contacts the user's SEM to get a partially decrypted message (if the user is not revoked). This is basically the same process as in mRSA.

7 Summary and Future Work

We described IB-mRSA, a practical and secure identity-based encryption scheme. It is compatible with standard RSA encryption and offers fine-grained control (revocation) of users security privileges.

Several issues remain for future work. It is unclear whether IB-mRSA can be shown secure under the standard model (our argument utilizes the random oracle setting). Moreover, we need a more formal analysis of semantic security. Another issue relates to IB-mRSA performance. Using a hash function for public key mapping makes encryption more expensive than RSA since the public exponent is random (and on the average half of the bits are set). We need to investigate alternative mapping functions that can produce more "efficient" RSA exponents.

8 Acknowledgements

Some of the initial ideas for this work emanated from discussions with Dan Boneh whose contribution is gratefully acknowledged. Thanks also go to the anonymous reviewers for their useful comments.

References

1. PKCS #7: Cryptographic message syntax standard. Technical note, RSA Laboratories, Nov. 1993. Version 1.5.
2. PKCS #12: Personal information exchange syntax. Technical note, RSA Laboratories, 1999. Version 1.0.
3. PKCS#1v2.1: Rsa cryptography standard. Technical note, RSA Laboratories, 2002. Version 2.1.
4. O. Baudron, D. Pointcheval, and J. Stern. Extended notions of security for multicast public key cryptosystems. In *27th International Colloquium on Automata, Languages and Programming (ICALP '2000)*, number 1853 in Lecture Notes in Computer Science. Springer-Verlag, Berlin Germany, July 2000.
5. M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Preneel [18], pages 259–274.
6. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, number 1462 in Lecture Notes in Computer Science, pages 26–45. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1998.
7. M. Bellare and P. Rogaway. Optimal asymmetric encryption — how to encrypt with RSA. In A. D. Santis, editor, *Advances in Cryptology – EUROCRYPT '94*, number 950 in Lecture Notes in Computer Science, pages 92–111. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1995.
8. D. Boneh, X. Ding, and G. Tsudik. Identity based encryption using mediated rsa. In *3rd Workshop on Information Security Application*, Jeju Island, Korea, Aug. 2002. KIISC.
9. D. Boneh, X. Ding, G. Tsudik, and C. M. Wong. A method for fast revocation of public key certificates and security capabilities. In *10th USENIX Security Symposium*, Washington, D.C., Aug. 2001. USENIX.
10. D. Boneh and M. Franklin. Identity-based encryption from the Weil Pairing. In Kilian [15], pages 213–229.
11. J.-S. Coron and D. Naccache. Security analysis of the gennaro-halevi-rabin signature scheme. In Preneel [18], pages 91–101.
12. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is secure under the rsa assumption. In Kilian [15], pages 260–274.
13. R. Ganesan. Augmenting kerberos with pubic-key cryptography. In T. Mayfield, editor, *Symposium on Network and Distributed Systems Security*, San Diego, California, Feb. 1995. Internet Society.
14. 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*, number 1592 in Lecture Notes in Computer Science, pages 123–139. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1999.
15. J. Kilian, editor. *Advances in Cryptology – CRYPTO '2001*, number 2139 in Lecture Notes in Computer Science. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 2001.

16. Microsoft. Microsoft Outlook©, http://www.microsoft.com.
17. OpenSSL User Group. The OpenSSL Project Web Page, http://www.openssl.org.
18. B. Preneel, editor. *Advances in Cryptology – EUROCRYPT '2000*, number 1807 in Lecture Notes in Computer Science, Brugge, Belgium, 2000. Springer-Verlag, Berlin Germany.
19. Qualcomm. Qualcomm eudora mailer, http://www.eudora.com.
20. A. Shamir. Identity-based cryptosystems and signature schemes. In G. Blakley and D. Chaum, editors, *Advances in Cryptology – CRYPTO '84*, number 196 in Lecture Notes in Computer Science, pages 47–53. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1985.

A Proof of Security

A.1 Notations and Attack Model

Throughout the appendix, we use the following notations

- $\mathcal{KG}$: Key Generation Function
- $\mathcal{PE}()$: IB-mRSA/OAEP encryption system.
- $\mathcal{DO}_{d_i}$: Decryption oracle with private key d_i
- $\mathcal{RO}$:Random oracle
- N: The common RSA modulus
- e_i/d_i: the i-th user's public key/private key
- n: The number of users in $\mathcal{PE}$
- q_e:The number of encryptions allowed to be performed by each user
- q_d: The maximum number of decryption queries the adversary can ask

Under the notion of indistinguishability of security, the adversary $\mathcal{A}$ takes the public key and outputs two equal length messages m_0, m_1. Then, it gets a challenge ciphertext C, computed by an encryption oracle which secretly picks $b \in_{\mathcal{R}} \{0,1\}$ and encrypts m_b. $\mathcal{A}$ is challenged to output b with a probability non-negligibly greater than $1/2$. In CCA attack model, $\mathcal{A}$ is allowed to send queries to a decryption oracle, with the restriction that $\mathcal{A}$ is not allowed to query on the challenge ciphertext c.

A.2 Proof of Lemma 1

Proof. The lemma means that if there exists an attack algorithm $\mathcal{B}$ with polynomial time complexity, breaking the security of IB-RSA/OAEP with success probability ϵ, then there exists an attack algorithm $\mathcal{F}$ with the same polynomial degree of running time, breaking RSA/OAEP with the same success probability; and vice versa.

The reverse direction is obvious. For any $\mathcal{F}$ that can break the indistinguishability of standard RSA, it breaks IB-mRSA in single-user mode. Thus we have $Succ^R(t', q_d) \leq Succ_1^{IB}(t, q_d)$. Now we show $Succ_1^{IB}(t, q_d) \leq Succ^R(t', q_d)$.

Let $\mathcal{B}$ be the polynomial algorithm attacking on the indistinguishability of $\mathcal{PE}(\mathcal{KG}, N, 1)$ containing the single user U_0 and its public key e_0 and secret bundle d_{u_0}. By allowing $\mathcal{B}$ to know d_{u_0}, we model the concern that the user in the system may be malicious. We construct $\mathcal{F}$ as the adversary algorithm against the standard RSA/OAEP $(\hat{N}, \hat{e})$ and analyze its success probability and time complexity. Replacing $\mathcal{KG}$ function in $\mathcal{PE}$ by a random oracle and acting as the random oracle and decryption oracle for $\mathcal{B}$, $\mathcal{A}$ runs $\mathcal{F}$ as follows.

Experiment $\mathcal{F}^{RSA}(\hat{N}, \hat{e}, \mathcal{DO}_d, \mathcal{RO})$

Select two random messages (m_0, m_1) with equal length. The encryption oracle EO_e secretly selects a random bit $b \in_{\mathcal{R}} \{0, 1\}$ and encrypts m_b into the ciphertext c. Given c, $\mathcal{F}$ runs the following to determine b.

1. Generate a random number r and a string id;
2. Initialize $\mathcal{PE}(\mathcal{KG}, N)$ with single-user setting by $N \leftarrow \hat{N}$, For user $ID_0 \leftarrow id$;
3. $\mathcal{PE}$ queries its random oracle ($\mathcal{F}$) for e_0;
4. $e_0 \leftarrow \hat{e}$;
5. Initialize $\mathcal{B}$ with (m_0, m_1, c) and the target system $\mathcal{PE}(\mathcal{KG}, \hat{N})$ and user public key e_0, user bundle r;
6. Run $\mathcal{B}$. The number of decryption queries is bounded by q_d:
   (a) For all $\mathcal{B}$'s random oracle queries on OAEP encoding/decoding, $\mathcal{F}$ forwards them to $\mathcal{RO}$ and hands the answers back;
   (b) For all $\mathcal{B}$'s decryption oracle queries, $\mathcal{F}$ forwards them to $\mathcal{DO}_d$, and hands the answers back;
   (c) For $\mathcal{B}$'s requests c to SEM (remember that the adversary might be inside the system): $\mathcal{F}$ queries $\mathcal{DO}_d$ on c. On getting the reply $c^d \bmod n$, $\mathcal{F}$ hands back $c^d/c^r \bmod n$ as the reply from SEM to $\mathcal{B}$.
7. $\mathcal{B}$ halts outputting a bit b';
8. Return b';

Clearly, if $\mathcal{B}$'s output b' equals b, $\mathcal{F}$ successfully discovers b. This holds for all polynomial algorithm $\mathcal{B}$. Thus we have $Succ_1^{IB}(t, q_d) \leq Succ^R(t', q_d)$. As for the time complexity of $\mathcal{F}$, the steps 1∼5 and steps 7,8 take constant time, in that the cost is independent of the security parameter, and step 6 runs in time t . Hence, the overall time for $\mathcal{F}$ is $t + c$, which leads us to the conclusion of $Succ_1^{IB}(t, q_d) = Succ^R(t', q_d)$. □

A.3 Proof of Lemma 2

Proof. If all users in $\mathcal{PE}$ are considered trusted, we do not need to consider all attacks originated from the inside users. The $\mathcal{PE}$ is therefore a IB-mRSA/OAEP in multi-user setting, whose security for single-mode is proved in Lemma 1.

To show the polynomial reduction, the proof in [5] constructs an attack algorithm $\mathcal{B}$ for single-setting. $\mathcal{B}$ calls another algorithm $\mathcal{A}$, which can break the multi-user setting. In order to argue the security in CCA2 model, $\mathcal{B}$ has to simulate the decryption oracles for $\mathcal{A}$. This is simple in their case, where $\mathcal{B}$ can invoke key generation function to obtain all needed public/private key pairs. Unfortunately, this is not the case in IB-mRSA setting, since the key generation will not give $\mathcal{B}$ the private keys. We slightly revise the original proof.

Still, $\mathcal{A}$ targets at a multi-use setting with public keys $\{N, e_0, \ldots, e_n\}$. However, we construct $\mathcal{B}$, targeting at $\{N, e = \prod_{i=0}^{n}(e_i)\}$. The algorithm for $\mathcal{A}$'s decryption query is shown below:

Decryption oracle simulator ($\mathcal{B}, \mathcal{A}, N, e_0, \ldots, e_n, e = \prod_{i=0}^{n} e_i$)
$\mathcal{B}$ simulates decryption oracle for $\mathcal{A}$ with the help from its own oracle $\mathcal{DO}_d$.
(d is the corresponding secret key to (n, e).)

1. $\mathcal{A} \rightarrow \mathcal{B}$: (c, e_i),
2. $\mathcal{B} \rightarrow \mathcal{DO}_d$: c
3. $\mathcal{B} \leftarrow \mathcal{DO}_d$: $c' = c^d \bmod n$
4. $\mathcal{B}$ computes $b = \frac{e}{e_i}$
5. $\mathcal{A} \leftarrow \mathcal{B}$: $a = c'^b \bmod n$

One can easily check that the answer a is exactly c^{1/e_i}. Thus, the proof for Theorem 4.1 in [5] still holds, which also proves this lemma.
□

A.4 Proof of Lemma 3

Proof. (Sketch) No secret channel is assumed either in IB-mRSA protocol execution or in the attack model. Thus, the outsider observes everything that the insider does, except for $\{d_{u_i}\}$ and $\Gamma_{d_u,n}$. However, an outsider can simulate $\Gamma_{d_u,n}$ with the help of a random oracle and the decryption oracles.

Note that a user's key-share is nothing but a random number derived from an idealized hash function, which can be replaced by the random oracle RO. The outsider can query RO and obtain a set of random values $\{r_i\}$ with the same distribution as $\{d_{u_i}\}$. For each ciphertext c in $\Gamma_{d_u,n}$ (encrypted with e_{u_i}) the adversary constructs $\Gamma'_{d_u,n}$ by computing $c^{d_{u_i}} = c^{d_i}/c^{r_i}$, where c^{d_i} is obtained from the decryption oracle $\mathcal{DO}_{d_i}$. All c, d_{u_i},r_i are random integers. (Note that c is also random since OAEP encoding is applied before exponentiation). Thus, $Pr\{\Gamma_{d_u,n}\} = Pr\{\Gamma'_{d_u,n}\}$, which leads to V_1 and V_2 having the same distribution □

A.5 Proof of Lemma 4

Proof. We show that $x|y$ is a sufficient and necessary condition for the existence of f.
SUFFICIENCY: if $x|y$, i.e. $\exists k \in \mathcal{N}$, s.t. $y = kx$. We construct f as

$$f : a \rightarrow a^k \bmod n$$

One can easily check that f is the desired function.
NECESSITY: Suppose there exists a function f satisfying the requirement, while $y = kx + r$, where $k, r \in \mathcal{N}$ and $1 \leq r \leq x$. Given $c = m^x \bmod n$, we can compute $c_1 = f(c) = m^y \bmod n$. Suppose $g = gcd(x, y)$, i.e $\exists a, b \in \mathcal{Z}$, s.t. $ax + by = g$. Thus, in polynomial time, we can get m^g by computing $c^a c_1^b \bmod n$. If we let $x = hg$, we have actually constructed a polynomial-time algorithm, which, taking $c = (m^g)^h \bmod n$ and h, n as input, outputs $c^{1/h} \bmod n$ without knowing the factorization of n. (Note that x is relatively prime to $\phi(n)$, which implies that h is also relatively prime to $\phi(n)$ and is a valid RSA public key exponent.) However, this contradicts the RSA assumption. □