Identity-Based Mediated RSA*

Dan Boneh[1], Xuhua Ding[2], and Gene Tsudik[2]

[1] Computer Science Department, Stanford University.
Email: dabo@cs.stanford.edu
[2] Department of Information and Computer Science, University of California, Irvine.
Email: {xhding,gts}@ics.uci.edu

Abstract. Identity-based encryption (IBE) [5] and digital signatures are important tools in modern secure communication. In general, identity-based cryptographic methods facilitate easy introduction of public key cryptography by allowing an entity's public key to be derived from some arbitrary identification value such as an email address or a phone number. Identity-based cryptography greatly reduces the need for, and reliance on, public key certificates.
Mediated RSA (mRSA) [4] is a simple and practical method of splitting RSA private keys between the user and the Security Mediator (SEM). Neither the user nor the SEM can cheat one another since each signature or decryption must involve both parties. mRSA allows fast and fine-grained control (revocation) over users' security privileges. However, mRSA still relies on public key certificates to derive public keys.
Current identity-based cryptographic methods do not support fine-grained revocation while mediated cryptography (such as mRSA) still relies on public key certificates to derive public keys. In this paper we present IB-mRSA, a variant of mRSA that combines identity-based and mediated cryptography. IB-mRSA is simple, secure and very efficient.

1 Introduction

In a typical public key 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 PKI model requires universal trust in certificate issuers (Certification Authorities or CAs). This also has some well-known side-effects such as 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 note that this assumption is not always realistic, especially, in wireless networks where connectivity is sporadic.

In contrast, identity-based cryptography changes the nature of obtaining public keys by constructing a one-to-one mapping between identities and public keys. Thus, identity-based cryptography greatly reduces the need for, and reliance on, public key certificates and certification authorities. Generally speaking, identity-based encryption and identity-based digital signatures are useful cryptographic tools that facilitate easy introduction of, and/or conversion to, public key cryptography by allowing a public key

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

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. However, one notable drawback is that current identity-based cryptographic methods do not support fine-grained revocation. (Revocation is typically done via Certificate Revocation Lists – CRLs or similar structures.)

Mediated RSA [4] is a simple and practical method of splitting RSA private keys between the user and the SEM. Neither the user nor the SEM can cheat one another since each signature or decryption operation must involve both parties. mRSA allows fast and fine-grained control (revocation) of users' security privileges. However, mRSA still relies on public key certificates to derive public keys.

Our goal in this paper is to combine the attractive features of identity-based cryptography with the fine-grained control of mRSA. To this end, we present IB-mRSA, a variant of mRSA that combines identity-based and mediated cryptography. IB-mRSA is simple, secure and very efficient.

Organization: The rest of this paper is organized as follows. In the next section we provide a brief overview of mediated RSA. Next, we describe IB-mRSA in Section 3 and analyze its security. Sections 5 and 6 discuss the implementation of IB-mRSA and its performance, respectively.

2 Overview of Mediated RSA

Mediated RSA (mRSA) involves a special entity, called a SEM an on-line partially trusted server. To sign or decrypt a message, Alice must first obtain a message-specific token from the SEM. Without this token Alice can not use her private key. To revoke Alice's ability to sign or decrypt, the administrator instructs the SEM to stop issuing tokens for Alice's public key. At that instant, Alice's signature and/or decryption capabilities are revoked. For scalability reasons, a single SEM serves many users. One of the mRSA's advantages is its transparency: SEM's presence is invisible to other users: in signature mode, mRSA yields standard RSA signatures, while in decryption mode, mRSA accepts plain RSA-encrypted messages.

The main idea behind mRSA is the splitting of an RSA private key into two parts as in threshold RSA [9]. One part is given to a user while the other is given to a SEM. If the user and the SEM cooperate, they employ their respective half-keys in a way that is functionally equivalent to (and indistinguishable from) standard RSA. The fact that the private key is not held in its entirety by any one party is transparent to the outside, i.e., to the those who use the corresponding public key. Also, knowledge of a half-key cannot be used to derive the entire private key. Therefore, neither the user nor the SEM can decrypt or sign a message without mutual consent.

We now provide a brief overview of mRSA functions. (For a detailed description and security analysis of mRSA, we refer the reader to [4].) The variant described below is the additive mRSA (+mRSA) as presented by Boneh, et al. in [4]. (There is also a very similar multiplicative mRSA (*mRSA) first proposed by Ganesan [8].) The first function `+mRSA.key` is used by the CA to set up the user's public/private key-pair and

split the private key into two shares. The other two, `+mRSA.sign` and `+mRSA.decr`, are the signing and decryption functions, respectively. We note that no encryption and signature verification functions are specified since they are identical to those in plain RSA.

Algorithm +mRSA.key (executed by CA)
Let k (even) be the security parameter

1. Generate random $k/2$-bit primes: p, q
2. $n \leftarrow pq$
3. $e \stackrel{r}{\leftarrow} Z^*_{\phi(n)}$
4. $d \leftarrow 1/e \bmod \phi(n)$
5. $d_u \stackrel{r}{\leftarrow} Z_n - \{0\}$
6. $d_{sem} \leftarrow (d - d_u) \bmod \phi(n)$
7. $SK \leftarrow d$
8. $PK \leftarrow (n, e)$

After computing the above values, CA securely communicates d_{sem} to the SEM and d_u – to the user. (See [4] for details.) The user's public key PK is released, as usual, in a public key certificate.

Protocol +mRSA.sign (executed by User and SEM)

1. USER: $h \leftarrow H(m)$
   where H() is a suitable hash function such as SHA-1 and $|H()| < k$
2. USER: send h to SEM.
3. In parallel:
   3.1 SEM:
   (a) If USER revoked return (ERROR)
   (b) $PS_{sem} \leftarrow h^{d_{sem}} \bmod n$
   (c) send PS_{sem} to USER
   3.2 USER:
   (a) $PS_u \leftarrow h^{d_u} \bmod n$
4. USER: $h' \leftarrow (PS_{sem} * PS_u)^e \bmod n$
5. USER: If $h' \neq h$ then return (ERROR)
6. $S \leftarrow (PS_{sem} * PS_u) \bmod n$
7. USER: return (h,S)

3 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.

Protocol +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: return (m)

The receiver's identity can be the receiver's email address, user id or any value unique to the receiver (essentially, an arbitrary string). To compute the receiver's encryption key, an efficient public mapping function $f()$ 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 among all users of a system. This modulus is assumed to be public and contained in a public key certificate issued, as usual, by some Certificate Authority (CA). To encrypt a message for a certain recipient (Bob), the sender (Alice) first computes $e_{Bob} = f(ID_{Bob})$ 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 our present context, we make an important assumption that:

> [IB-mRSA Assumption:] Throughout the lifetime of the system, the adversary is unable to compromise a SEM.

Obviously, without this assumption, IB-mRSA would offer no security: 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 that 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.

3.1 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 compute 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 prime. Then it computes $n = pq$, which is, in fact, a Blum integer. Since n is a Blum integer, a randomly chosen number in Z_n has negligible probability of not being relatively prime to $\phi(n)$. (See Section 4 for further discussion.) Hence, our mapping function f can be quite trivial.

The public exponent is set to be the email address represented as a binary string, padded with zeros on the left and with the rightmost bit set to one. This ensures that e_{Alice} is odd and, with overwhelming probability, relatively prime to $\phi(n)$. It is assumed that the email address is at most 8 bits shorter than the size of the RSA modulus. This is reasonable considering practical RSA modulus sizes. The currently recommended minimum RSA modulus size is 1024 bits. Consequently, our assumption translates into the email address being at most 127 characters (bytes) long. We observe that most email addresses tend to be under 30 characters.

In fact, it is largely unnecessary to use the *entire* email address as input to $f()$; it suffices to use only its leftmost component, i.e., the "username" portion. This is because the rest of the address is common to all email users in the domain. For example, if Alice's email address is: `Alice.Smith@ics.uci.edu` we can use "`Alice.Smith`" as input to $f()$. The common part "`ics.uci.edu`" can be assumed to be part of the domain-wide (organizational) certificate.

The complete IB-mRSA key generation proceeds as follows:

`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. $n \leftarrow pq$
3. For each user (Alice):
   (a) $k' \leftarrow k - |ID_Alice| - 8$
   (b) $e_{Alice} \leftarrow f(ID_{Alice}) = 0^{k'}||ID_{Alice}||00000001$
   (c) $d_{Alice} \leftarrow 1/e_{Alice} \bmod \phi(n)$
   (d) $d_{Alice,u} \stackrel{r}{\leftarrow} Z_n - \{0\}$
   (e) $d_{Alice,sem} \leftarrow (d - d_{Alice,u}) \bmod \phi(n)$

A domain- or system-wide certificate ($Cert_{org}$) is issued by the CA after completion of the key generation algorithm. This certificate contains all the usual fields normally found in RSA certificates with few exceptions discussed later in Section 4. In particular, it contains the common modulus n and (if applicable) the common part of

the email address for all users. User's and SEM's key shares are distributed securely as in the mRSA scheme described in Section 2.

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.

3.2 IB-mRSA Encryption

To encrypt a message, the sender needs the recipient's email address and its organization certificate.

Algorithm IB-mRSA.encr

1. $e \leftarrow IB - mRSA.key(Email)$
2. Retrieve n from organization certificate;
3. Let (e, n) be the public key and encrypt message m using standard RSA encryption with OAEP padding.

Note that the recipient's public key certificate is not required for the sender to encrypt. Since the key is derived from the receiver's unique identifier, the sender does not need a certificate to ensure that the intended receiver is the correct public key holder. Furthermore, instantaneous 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: the security administrator merely notifies the SEM and the revoked user is unable to decrypt any further messages.

3.3 Signature Protocol

IB-mRSA can be also used for signing messages. Since the signing protocol is the same as in mRSA, we do not provide a detailed description. Basically, a user computes its own half-signature while the SEM computes its half-signature. When the two parts are coalesced together, a full-blown RSA signatures is obtained.

The verification procedure is slightly different from mRSA (or plain RSA). In mRSA and RSA, the verifier obtains the public verification key, from the signer's public key certificate. The certificate can be obtained from the signature structure itself or from some public site. In ID-based signature schemes, the verifier computes the signer's public key using the signer's identity. In IB-mRSA, as mentioned before, we still need a domain certificate, which includes the common RSA modulus n. However, we should point it out that this certificate is not a normal public key certificate but a sort of an attribute certificate for the entire domain. The verification protocol is as follows:

Note that, since signature generation is the same as in mRSA, the binding signature semantic is preserved [4]. This means that the verifier can be sure that the signer's

Algorithm IB-mRSA.verify

1. Extract and verify domain certificate from signature or from a public site.
2. Obtain RSA modulus n from domain certificate;
3. $e \leftarrow IB - mRSA.key(Email)$;
4. Let (e, n) be the public key and verify the signature using standard RSA verification algorithm with OAEP padding.

private key was valid (believed by the SEM to be valid) at the time the signature was computed.

4 Security Analysis

In this section, we discuss some security issues unique to IB-mRSA. Clearly, the biggest security concern for IB-mRSA is the use of the common modulus for all users within the same domain.

SEM ***Security:*** Let us consider an attacker trying to decrypt a message sent to Alice or to forge Alice's signature on a certain message. Recall that the token sent by SEM back to Alice is $t = x^{d_{sem}} \bmod N$ for some value of x. The attacker sees both x and the token t. In fact, since there is no authentication of the user's request to the SEM, the attacker can obtain this t for any x of its choice. We claim that this information is of no use to the attacker. The reason is that, d_{sem} is just a random number in Z_n independent of the rest of the attacker's view. Using a simulation argument, we claim that any attack possible with the SEM can be mounted to attack standard RSA. One can simulate the SEM by picking a random integer $d_{sem} \in_R Z_n$ and thus use the attack on the SEM to mount an attack on standard RSA.

Suppose the attacker is able to compromise the SEM and expose the secret key d_{sem}. This enables the attacker to "un-revoke" previously revoked, or block possible future revocation of currently valid, certificates. However, knowledge of d_{sem} does not enable the attacker to decrypt or sign messages on behalf of its users. The reason is obvious: Note that Alice does not send 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.

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 one 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. Thus, in the context of IB-mRSA we have to assume that a SEM is a **fully** trusted party, as opposed to **semi-trusted** in mRSA.

One unclear issue has to do with the existence of attacks on a common modulus with a large number of public/private key pairs. With one public/private key pair, it is widely believed that RSA encryption (with OAEP padding) is secure against adaptive chosen ciphertext attack [2], [7]. More specifically, such encryption is secure even if the attacker is allowed to employ a decryption oracle at will. In IB-mRSA we introduce a new attack model:

> The adversary is allowed to freely choose e', and ask the decryption/signature oracle to decrypt/sign a chosen ciphertext/plaintext with d' any number of times. (Where d' is the private counterpart of e' and RSA modulus is fixed.)

We refer to this type of a decryption oracle as a *multi-key oracle*. This oracle reveals more information than a traditional decryption or signature oracle, however, it does not possess more secret information. The reason being that, knowing one RSA private/public key pair is equivalent to knowing the factorization of n, which allows one to compute a private exponent for any given public exponent. A SEM in IB-mRSA can be viewed as a multi-key oracle in a sense that "bad" users can act as attackers and use the SEM to decrypt/sign any number of chosen messages. Although no formal security proof is offered in this paper, we were unable to find any attacks in this model. (Albeit, it seems that the security of the *multi-key oracle* is somehow tied to the *Strong RSA* Assumption [6].)

Key Generation In Section 3.1, we directly use the numerical value of an email address as a public key. The main requirement for an integer to form a proper RSA public exponent e is that it should be relatively prime with $\phi(n)$. Clearly, not all ASCII strings satisfy this requirement.

Since we choose $n = (2p' + 1)(2q' + 1)$, we have $\phi(n) = 4p'q'$. Both p' and q' are large primes (ca. 511 bits) while a typical email address is typically at most 320 bits, which is much smaller than p' and q'. In addition, note that all email addresses are converted into odd integers in *Algorithm IBE mRSA.Key*. Thus the derived e (which is odd) is not divisible by any of: $\{2, 4, p', q'\}$. In other words, e is relatively prime with $\phi(n)$.

Even if a very long email address is used, the probability of $gcd(e, \phi(n)) > 1$ is negligible. The reason is that all odd integers which are not relatively prime with $4p'q'$ are: $\{p', 3p', \ldots, (4q' - 1)p'\}$ and $\{q', 3q', \ldots, (4p' - 1)q'\}$. There are $2(p' + q' - 1)$ such integers. Thus, for a random odd integer $r < \phi(n)$:

$$Pr[gcd(r, \phi(n)) > 1] = \frac{(p' + q' - 1)}{2p'q'}$$

which is overwhelmingly small.

As pointed out in [3], a secure RSA setting requires that the decryption key to be large enough. Usually it should be at least larger than $n^0.5$, which is relatively the size of p and q. This is guaranteed by choosing short email address. For example, if e is at most 320 bits, the corresponding d should be longer than 512 bits, in case of a 1024-bit modulus.

Forward Security One of the motivations for research on ID-based cryptosystems is to provide forward security [1]. A usual forward security technique involves using a time-variant parameter along with user identifier, such that the derived key changes with time, e.g., every day or every month.

IB-mRSA can provide forward security in the same manner. Rather than computing e from just an email address, current date can be appended. In this case, the key distribution process in Section 3.1 should be revised. Instead of getting the signed and encrypted key bundle when joining an organization, the user sends a request to the CA when it received the first encrypted message in the current period. This causes the CA to generate and distribute the appropriate keys.

This is clearly an expensive process since the CA should be on-line and ready for users' requests. In addition, the CA needs to compute all non-revoked keys in every period. Thus, forward secure IB-mRSA is more suitable for organizations where IB-mRSA is lightly used and there exists an secure channel between users and the CA to protect key generation requests.

Organization Certificate and User Certificate Once the common RSA modulus n and the mapping function $f()$ is chosen, the CA issues and publishes a domain (attribute) certificate which includes these two values. In addition, the certificate also contains the organization/domain name (e.g. `@ics.uci.edu`).

User certificates are, as mentioned before, optional in IB-mRSA. They actually bridge IB-mRSA with plain RSA and mRSA. With the individual certificates, IB-mRSA becomes fully compatible with normal mRSA. The sender can get the public key from the individual certificate if it trusts it, or rely on the domain certificate and receiver's email address to compute the public key.

5 Implementation

To test our assertions and gain experimental and practical experience, we implemented the IB-mRSA scheme. Our implementation includes a IB-mRSA library as well as a fully functional email plug-ins for Eudora[12] and Microsoft Outlook[10]. It is freely available from the following web address:

`http://sconce.ics.uci.edu/SUCSES`

This implementation, incidentally, is part of the of the SUCSES project code release and borrows much of the code from the mRSA package. We re-used the basic mRSA functions, including user and SEM bundle generation as well as decryption and signature procedures.

Our Eudora and Outlook plug-ins allow the sender to encrypt outgoing email to all clients in the same domain using only one domain (organizational) certificate. When the user is ready to send, the 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 1.

When an email message encrypted with IB-mRSA is received, an icon for IB-mRSA is dislayed in the message window. To decrypt the message, the user just clicks on the

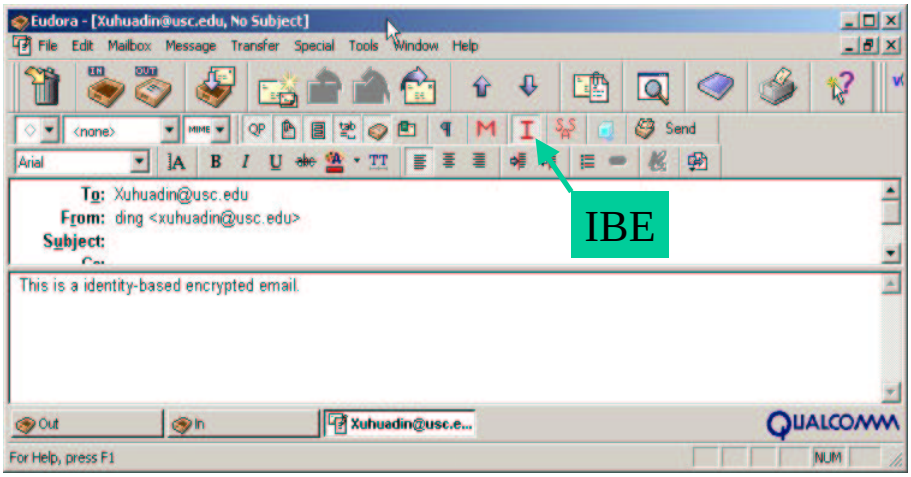


Fig. 1. Eudora IBE Plugin

IB-mRSA icon. The plug-in then contacts the user's SEM to get a partial decrypted message (provided the user is not revoked). This is basically the same process as plain mRSA [4].

6 Performance

When plain RSA is used for encryption, the public encryption exponent e is typically a small integer with only a few bits set to 1. One example is the popular OpenSSL toolkit [11] 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 identity-based systems, there is no such luxury of choosing special exponents. Therefore, e is a larger integer with likely higher number of 1 bits.

Keys	RSA Modulus 1Kb	RSA Modulus 2Kb	RSA Modulus 4Kb
$65,537$	2.3 ms	3.6 ms	11.7 ms
xhding@isi.edu	2.8 ms	9.5 ms	32.1 ms
Alice.Smith@ics.uci.edu	4.8 ms	15.5 ms	53.7 ms

Table 1. Performance Comparison of Different Encryption Keys

We ran some simple tests to measure the cost of IB-mRSA encryption for public keys derived from email addresses. The encryption was tested using Linux 2.4 version of OpenSSL on an 800MHz PIII workstation. In our tests, we used: 1) "default" encryption exponent $65,537$ and 2) two other exponents derived from different email addresses. For each key, we encrypted the same message one thousand times and obtained the average. The results are depicted in Table 1.

¿From the results in Table 1, we note that encryption with email address-derived keys does not introduce significant added overhead. Although the IB-mRSA encryption overhead increases as the RSA modulus size grows, it is somewaht negligible for 1024-bit modulus (which is currently the most common size).

The decryption cost for IB-mRSA is identical to mRSA. The performance of mRSA has been reported on by Boneh, et al. in [4]. For example, a 1024-bit mRSA decryption costs around 30ms 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 90ms that is needed for Boneh/Frnklin IBE decryption (for 1024 bits of security on the same hardware platform).

7 IB-mRSA versus Boneh/Franklin IBE

Recently, Boneh and Franklin developed a novel and elegant identity-based encryption system (IBE) based on Weil Pairing [5]. IBE represents a significant advance in cryptography since the problem that it solved has been open for many years. However, IBE does not 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 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. Therefore, the Public Key Generator (PKG, in IBE's parlance) must be on-line much of the time, or, at least, very often. Consequently, we observe that, when IBE is used to provide fine-grained revocation, a PKG is, for all practical purposes, equivalent to a SEM in mRSA and IB-mRSA.

Our conclusion is that, in functional terms, IB-mRSA seems to offer security equivalent to IBE (when the latter provides fine-grained revocation). This is because compromise of a PKG in IBE results in a total system break. The same happens upon a SEM compromise in IB-mRSA. Moreover, IB-mRSA offers some practical advantages over IBE.

First, IB-mRSA is fully compatible with plain RSA if (optional) individual certificates are used. Thus, it offers a smooth and natural transition from ID-based to normal cryptography. Also, IB-mRSA (which takes roughly 4-5 times less efficient than plain RSA) offers significantly better performance than IBE.

8 Future Work and Summary

In this paper, we described a simple, secure and efficient IB-mRSA scheme. IB-mRSA combines the convenience of identity-based encryption (thus greatly reducing the need for public key certificates) with the functionality of mediated RSA which provides fine-grained revocation.

IB-mRSA allows the sender (encryptor) to skip the costly checking of individual public key certificates. The tight control over users' security capability is inherited from

mediated RSA mechanism since the decryption protocol for IB-mRSA is essentially the same as in mRSA. Furthermore, IB-mRSA can be easily be extended to provide forward security.

Several issues remain for future work. First, we have not provided a formal proof that RSA (with OAEP padding) is secure against adaptive chosen ciphertext attack if the decryption oracle offers replies for multiple private keys with the same modulus. We also need to investigate alternative mapping functions in order to speed up the encryption with derived public keys.

References

1. R. Anderson. Invited lecture at the acm conference on computer and communication security (ccs'97), 1997.
2. 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. Final (revised) version appeared November 19, 1995. Available from `http://www-cse.ucsd.edu/users/mihir/papers/oaep.html`.
3. D. Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices of the American Mathematical Society (AMS)*, 46(2):203–213, 1999.
4. 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.
5. D. Boneh and M. Franklin. Identity-based encryption from the Weil Pairing. In J. Kilian, editor, *Advances in Cryptology – CRYPTO '2001*, number 2139 in Lecture Notes in Computer Science, pages 213–229. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 2001.
6. E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In B. S. Kaliski, Jr., editor, *Advances in Cryptology – CRYPTO '97*, number 1294 in Lecture Notes in Computer Science. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1997.
7. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is still alive! Record 2000/061, Cryptology ePrint Archive, Nov. 2000. Revised on Dec. 4, 2000. A revised version titled "RSA—OAEP is secure under the RSA Assumption".
8. 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.
9. P. Gemmel. An introduction to threshold cryptography. *RSA CryptoBytes*, 2(7), 1997.
10. Microsoft. Microsoft Outlook©, `http://www.microsoft.com`.
11. OpenSSL User Group. The OpenSSL Project Web Page, `http://www.openssl.org`.
12. Qualcomm. Qualcomm eudora mailer, `http://www.eudora.com`.