How to Leak a Secret: Theory and Applications of Ring Signatures

Ronald L. Rivest[1], Adi Shamir[2], and Yael Tauman[1]

[1] Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139,

[2] Computer Science department, The Weizmann Institute, Rehovot 76100, Israel.

Abstract. In this work we formalize the notion of a *ring signature*, which makes it possible to specify a set of possible signers without revealing which member actually produced the signature. Unlike group signatures, ring signatures have no group managers, no setup procedures, no revocation procedures, and no coordination: any user can choose any set of possible signers that includes himself, and sign any message by using his secret key and the others' public keys, without getting their approval or assistance. Ring signatures provide an elegant way to leak authoritative secrets in an anonymous way, to sign casual email in a way that can only be verified by its intended recipient, and to solve other problems in multiparty computations.

Our main contribution lies in the presentation of efficient constructions of ring signatures; the general concept itself (under different terminology) was first introduced by Cramer et al. [CDS94]. Our constructions of such signatures are unconditionally signer-ambiguous, secure in the random oracle model, and exceptionally efficient: adding each ring member increases the cost of signing or verifying by a single modular multiplication and a single symmetric encryption. We also describe a large number of extensions, modifications and applications of ring signatures which were published after the original version of this work (in Asiacrypt 2001).

Keywords: signature scheme, ring signature scheme, signer-ambiguous signature scheme, group signature scheme, designated verifier signature scheme.

1 Introduction

The general notion of a *group signature scheme* was introduced in 1991 by Chaum and van Heyst [CV91]. In such a scheme, a trusted group manager predefines certain groups of users and distributes specially designed keys to their members. Individual members can then use these keys to anonymously sign messages on behalf of their group. The signatures produced by different group members look indistinguishable to their verifiers, but not to the group manager who can revoke the anonymity of misbehaving signers.

In this work we formalize the related notion of *ring signature schemes*. These are simplified group signature schemes that have only users and no managers (we call such signatures "ring signatures" instead of "group signatures" since rings are geometric regions with uniform periphery and no center). Group signatures are useful when the members want to cooperate, while ring signatures are useful when the members do not want to cooperate. Both group signatures and ring signatures are *signer-ambiguous*, but in a ring signature scheme there are no prearranged groups of users, there are no procedures for setting, changing, or deleting groups, there is no way to distribute specialized keys, and there is no way to revoke the anonymity of the actual signer (unless he decides to expose himself). Our only assumption is that each member is already associated with the public key of some standard signature scheme such as RSA. To produce a ring signature, the *actual signer* declares an arbitrary set of *possible signers* that must include himself, and computes the signature entirely by himself using only his secret key and the others' public keys. In particular, the other possible signers could have chosen their RSA keys only in order to conduct e-commerce over the internet, and may be completely unaware that their public keys are used by a stranger to produce such a ring signature on a message they have never seen and would not wish to sign.

The notion of ring signatures is not completely new, but previous references do not crisply formalize the notion, and propose constructions that are less efficient and/or that have different, albeit related, objectives. They tend to describe this notion in the context of general group signatures or multiparty constructions, which are quite inefficient. For example, Chaum et al. [CV91]'s schemes three and four, and the two signature schemes in Definitions 2 and 3 of Camenisch's paper [Cam97] can be viewed as ring signature schemes. However the former schemes require zero-knowledge proofs with each signature, and the latter schemes require as many modular exponentiations as there are members in the ring. Cramer et al. [CDS94] show how to produce witness-indistinguishable interactive proofs. Such proofs could be combined with the Fiat-Shamir technique to produce ring signature schemes. Similarly, DeSantis et al. [SCPY94] show that interactive SZK for random self-reducible languages are closed under monotone boolean operations, and show the applicability of this result to the construction of a ring signature scheme (although they don't use this terminology).

The direct construction of ring signatures proposed in this paper is based on a completely different idea, and is exceptionally efficient for large rings (adding only one modular multiplication and one symmetric encryption per ring member both to generate and to verify such signatures). The resultant signatures are unconditionally signer-ambiguous and secure in the random oracle model. This model, formalized in [BR93], assumes that all parties have oracle access to a truly random function.

There have been several followup papers on the theory and applications of ring signatures. We summarize these results in Section 7.

2 Definitions and Applications

2.1 Ring signatures

Terminology: We call a set of *possible signers* a *ring*. We call the ring member who produces the actual signature the *signer* and each of the other ring members a *non-signer*.

We assume that each possible signer is associated (via a PKI directory or certificate) with a public key P_k that defines his signature scheme and specifies his verification key. The corresponding secret key (which is used to generate regular signatures) is denoted by S_k. The general notion of a ring signature scheme does not require any special properties of these individual signing schemes, but our simplest construction assumes that they use trapdoor one-way permutations (such as the RSA functions) to generate and verify signatures.

A ring signature scheme is defined by two procedures:

- **ring-sign**$(m, P_1, P_2, \ldots, P_r, s, S_s)$ which produces a ring signature σ for the message m, given the public keys $P_1, P_2, \ldots, P_r$ of the r ring members, together with the secret key S_s of the s-th member (who is the actual signer).
- **ring-verify**(m, σ) which accepts a message m and a signature σ (which includes the public keys of all the possible signers), and outputs either *true* or *false*.

A ring signature scheme is *set-up free*: The signer does not need the knowledge, consent, or assistance of the other ring members to put them in the ring; all he needs is knowledge of their regular public keys. Different members can use different independent public key signature schemes, with different key and signature sizes. Verification must satisfy the usual soundness and completeness conditions, but in addition we want the signatures to be *signer-ambiguous* in the sense that a signature should leek no information about the identity of the signer. This anonymity property can be either *computational* or *unconditional*. Our main construction provides unconditional anonymity in the sense that even an infinitely powerful adversary with access to an unbounded number of chosen-message signatures produced by the same ring member cannot guess his identity with any advantage, and cannot link additional signatures to the same signer.

Note that the size of any ring signature must grow linearly with the size of the ring, since it must list the ring members; this is an inherent disadvantage of ring signatures as compared to group signatures that use predefined groups.

2.2 Leaking secrets

To motivate the title for this paper, suppose that Bob (also known as "Deep Throat") is a member of the cabinet of Lower Kryptonia, and that Bob wishes to leak a juicy fact to a journalist about the escapades of the Prime Minister, in such a way that Bob remains anonymous, yet such that the journalist is convinced that the leak was indeed from a cabinet member.

Bob cannot send to the journalist a standard digitally signed message, since such a message, although it convinces the journalist that it came from a cabinet member, does so by directly revealing Bob's identity.

It also doesn't work for Bob to send the journalist a message through a standard "anonymizer" [Ch81,Ch88,GRS99], since the anonymizer strips off all source identification and authentication: the journalist would have no reason to believe that the message really came from a cabinet member at all.

A standard group signature scheme does not solve the problem, since it requires the prior cooperation of the other group members to set up, and leaves Bob vulnerable to later identification by the group manager, who may be controlled by the Prime Minister.

The correct approach is for Bob to send the story to the journalist (through an anonymizer), signed with a ring signature scheme that names each cabinet member (including himself) as a ring member. The journalist can verify the ring signature on the message, and learn that it definitely came from a cabinet member. He can even post the ring signature in his paper or web page, to prove to his readers that the juicy story came from a reputable source. However, neither he nor his readers can determine the actual source of the leak, and thus the whistleblower has perfect protection even if the journalist is later forced by a judge to reveal his "source" (the signed document).

2.3 Designated verifier signature schemes

A designated verifier signature scheme is a signature scheme in which signatures can only be verified by a single "designated verifier" chosen by the signer. It can be viewed as a "light signature scheme" which can authenticate messages to their intended recipients without having the nonrepudiation property. This concept was first introduced by Jakobsson, Sako and Impagliazzo at Eurocrypt 96 [JSI96].

A typical application is to enable users to authenticate casual emails without being legally bound to their contents. For example, two companies may exchange drafts of proposed contracts. They wish to add to each email an authenticator, but not a real signature which can be shown to a third party (immediately or years later) as proof that a particular draft was proposed by the other company.

One approach would be to use zero knowledge interactive proofs, which can only convince their verifiers. However, this requires interaction and is difficult to integrate with standard email systems and anonymizers. We can use non-interactive zero knowledge proofs, but then the authenticators become signatures which can be shown to third parties. Another approach is to agree on a shared secret symmetric key k, and to authenticate each contract draft by appending a message authentication code (MAC) for the draft computed with key k. A third party would have to be shown the secret key to validate a MAC, and even then he wouldn't know which of the two companies computed the MAC. However, this requires an initial set-up procedure to generate the secret symmetric key k.

A designated verifier scheme provides a simple solution to this problem: company A can sign each draft it sends, naming company B as the designated verifier. This can be easily achieved by using a ring signature scheme with companies A and B as the ring members. Just as with a MAC, company B knows that the message came from company A (since no third party could have produced this ring signature), but company B cannot prove to anyone else that the draft of the contract was signed by company A, since company B could have produced this draft by itself. Unlike the case of MAC's, this scheme uses public key cryptography, and thus A can send unsolicited email to B signed with the ring signature without any preparations, interactions, or secret key exchanges. By using our proposed ring signature scheme, we can turn standard signature schemes into designated verifier schemes, which can be added at almost no cost as an extra option to any email system.

3 Efficiency of our Ring Signature Scheme

When based on Rabin or RSA signatures, our ring signature scheme is particularly efficient:

- signing requires one modular exponentiation, plus one or two modular multiplications for each non-signer.
- verification requires one or two modular multiplications for each ring member.

In essence, generating or verifying a ring signature costs the same as generating or verifying a regular signature plus an extra multiplication or two for each non-signer, and thus the scheme is truly practical even when the ring contains hundreds of members. It is two to three orders of magnitude faster than Camenisch's scheme, whose claimed efficiency is based on the fact that it is 4 times faster than earlier known schemes (see bottom of page 476 in his paper [Cam97]). In addition, a Camenisch-like scheme uses linear algebra in the exponents, and thus requires all the members to use the same prime modulus p in their individual signature schemes. One of our design criteria is that the signer should be able to assemble an arbitrary ring without any coordination with the other ring members. In reality, if one wants to use other users' public keys, they are much more likely to be RSA keys, and even if they are based on discrete logs, different users are likely to have different moduli p. The only realistic way to arrange a Camenisch-like signature scheme is thus to have a group of consenting parties.

4 The Proposed Ring Signature Scheme (RSA version)

Suppose that Alice wishes to sign a message m with a ring signature for the ring of r individuals A_1, A_2, $\ldots$, A_r, where the signer Alice is A_s, for some value of s, $1 \leq s \leq r$. To simplify the presentation and proof, we first describe a ring signature scheme in which all the ring members use RSA [RSA78] as their individual signature schemes. The same construction can be used for any other trapdoor one way permutation, but we have to modify it slightly in order to use trapdoor one way functions (as in, for example, Rabin's signature scheme [Rab79]).

4.1 RSA trapdoor permutations

Each ring member A_i has an RSA public key $P_i = (n_i, e_i)$ which specifies the trapdoor one-way permutation f_i of $\mathbf{Z}_{n_i}$:

$$f_i(x) = x^{e_i} \pmod{n_i} \ .$$

We assume that only A_i knows how to compute the inverse permutation f_i^{-1} efficiently, using trapdoor information (i.e., $f_i^{-1}(y) = y^{d_i} (\bmod\ n_i)$, where $d_i = e_i^{-1} (\bmod\ \phi(n_i))$ is the trapdoor information). This is the original Diffie-Hellman model [DH76] for public-key cryptography.

Extending trapdoor permutations to a common domain

The trapdoor RSA permutations of the various ring members will have domains of different sizes (even if all the moduli n_i have the same number of bits). This makes it awkward to combine the individual signatures, and thus we extend all the trapdoor permutations to have as their common domain the same set $\{0,1\}^b$, where 2^b is some power of two which is larger than all the moduli n_i's.

For each trapdoor permutation f over $\mathbf{Z}_n$, we define the extended trapdoor permutation g over $\{0,1\}^b$ in the following way. For any b-bit input m define nonnegative integers q and r so that $m = qn + r$ and $0 \leq r < n$. Then

$$g(m) = \begin{cases} qn + f(r) & \text{if } (q+1)n \leq 2^b \\ m & \text{else.} \end{cases}$$

Intuitively, g is defined by using f to operate on the low-order digit of the n-ary representation of m, leaving the higher order digits unchanged. The exception is when this might cause a result larger than $2^b - 1$, in which case m is unchanged. If we choose a sufficiently large b (e.g. 160 bits larger than any of the n_i's), the chance that a randomly chosen m is unchanged by the extended g becomes negligible. (A stronger but more expensive approach, which we don't need, would use instead of $g(m)$ the function $g'(m) = g((2^b - 1) - g(m))$ which can modify all its inputs). The function g is clearly a permutation over $\{0,1\}^b$, and it is a one-way trapdoor permutation since only someone who knows how to invert f can invert g efficiently on more than a negligible fraction of the possible inputs.

4.2 Symmetric encryption

We assume the existence of a publicly defined symmetric encryption algorithm E such that for any key k of length l, the function E_k is a permutation over b-bit strings. Here we use the ideal cipher model which assumes that all the parties have access to an oracle that provides truly random answers to new queries of the form $E_k(x)$ and $E_k^{-1}(y)$, provided only that they are consistent with previous answers and with the requirement that E_k be a permutation. It was shown in [BSS02] that the ideal cipher model can be reduced to the random oracle model without almost any efficiency loss.[1] For simplicity we use the ideal cipher model in this presentation.

4.3 Hash functions

We assume the existence of a publicly defined collision-resistant hash function h that maps arbitrary inputs to strings of length l, which are used as keys for E. We model h as a random oracle. (Since h need not be a permutation, different queries may have the same answer, and we do not consider "h^{-1}" queries.)

4.4 Combining functions

We define a family of keyed "combining functions" $C_{k,v}(y_1, y_2, \ldots, y_r)$ which take as input a key k, an initialization value v, and arbitrary values y_1, y_2, $\ldots$, y_r in $\{0,1\}^b$. Each such combining function uses E_k as a sub-procedure, and produces as output a value z in $\{0,1\}^b$ such that given any fixed values for k and v, we have the following properties.

1. **Permutation on each input:** For each $s \in \{1, \ldots, r\}$, and for any fixed values of all the other inputs y_i, $i \neq s$, the function $C_{k,v}$ is a one-to-one mapping from y_s to the output z.
2. **Efficiently solvable for any single input:** For each $s \in \{1, \ldots, r\}$, given a b-bit value z and values for all inputs y_i except y_s, it is possible to efficiently find a b-bit value for y_s such that $C_{k,v}(y_1, y_2, \ldots, y_r) = z$.
3. **Infeasible to solve verification equation for all inputs without trapdoors:** Given k, v, and z, it is infeasible for an adversary to solve the equation

$$C_{k,v}(g_1(x_1), g_2(x_2), \ldots, g_r(x_r)) = z \tag{1}$$

for x_1, x_2, $\ldots$, x_r, (given access to each g_i, and to E_k) if the adversary can't invert any of the trapdoor functions g_1, g_2, $\ldots$, g_r.

For example, the function

$$C_{k,v}(y_1, y_2, \ldots, y_r) = y_1 \oplus y_2 \oplus \cdots \oplus y_r$$

[1] It was shown in [LR88] that the ideal cipher model can *always* be reduced to the random oracle model (with some efficiency loss).

(where $\oplus$ is the exclusive-or operation on b-bit words) satisfies the first two of the above conditions, and can be kept in mind as a candidate combining function. Indeed, it was the first one we tried. But it fails the third condition since for any choice of trapdoor one-way permutations g_i, it is possible to use linear algebra when r is large enough to find a solution for $x_1, x_2, \ldots, x_r$ without inverting any of the g_i's. The basic idea of the attack is to choose a random value for each x_i, and to compute each $y_i = g_i(x_i)$ in the easy forward direction. If the number of values r exceeds the number of bits b, we can find with high probability a subset of the y_i bit strings whose XOR is any desired b-bit target z. However, our goal is to represent z as the XOR of all the values $y_1, y_2, \ldots, y_r$ rather than as a XOR of a random subset of these values. To overcome this problem, we choose for each i *two* random values x'_i and x''_i, and compute their corresponding $y'_i = g_i(x'_i)$ and $y''_i = g_i(x''_i)$. We then define $y'''_i = y'_i \oplus y''_i$, and modify the target value to $z' = z \oplus y'_1 \oplus y'_2, \ldots \oplus y'_r$. We use the previous algorithm to represent z' as a XOR of a random subset of y'''_i values. After simplification, we get a representation of the original z as the XOR of a set of r values, with exactly one value chosen from each pair (y'_i, y''_i). By choosing the corresponding value of either x'_i or x''_i, we can solve the verification equation without inverting any of the trapdoor one-way permutations g_i. (One approach to countering this attack, which we don't explore further here, is to let b grow with r.)

Even worse problems can be shown to exist in other natural combining functions such as addition mod 2^b. Assume that we use the RSA trapdoor functions $g_i(x_i) = x_i^3 (\text{mod } n_i)$ where all the moduli n_i have the same size b. It is known [HW79] that any nonnegative integer z can be efficiently represented as the sum of exactly nine nonnegative integer cubes $x_1^3 + x_2^3 + \ldots + x_9^3$. If z is a b-bit target value, we can expect each one of the x_i^3 to be slightly shorter than z, and thus their values are not likely to be affected by reducing each x_i^3 modulo the corresponding b-bit n_i. Consequently, we can solve the verification equation $(x_1^3 \text{ mod } n_1) + (x_2^3 \text{ mod } n_2) \ldots + (x_9^3 \text{ mod } n_9) = z (\text{mod } 2^b)$ with nine RSA permutations without inverting any one of them.

Our proposed combining function utilizes the symmetric encryption function E_k as follows:

$$C_{k,v}(y_1, y_2, \ldots, y_r) = E_k(y_r \oplus E_k(y_{r-1} \oplus E_k(y_{r-2} \oplus E_k(\ldots \oplus E_k(y_1 \oplus v) \ldots)))) \ .$$

This function is applied to the sequence $(y_1, y_2, \ldots, y_r)$, where $y_i = g_i(x_i)$, as shown in Figure 1.

This function is clearly a permutation on each input, since the XOR and E_k functions are permutations. In addition, it is efficiently solvable for any single input since knowledge of k makes it possible to run the evaluation forwards from the initial v and backwards from the final z in order to uniquely compute any missing value y_i.

This function can be used to construct a signature scheme as follows: In order to sign a message m, set $k = h(m)$, where h is some predetermined hash function, and output $x_1, \ldots, x_r$ such that $C_{k,v}(g_1(x_1), g_2(x_2), \ldots, g_r(x_r)) = v$. Notice that forcing the output z to be equal to the input v, bends the line into the ring shape shown in Fig. 2.

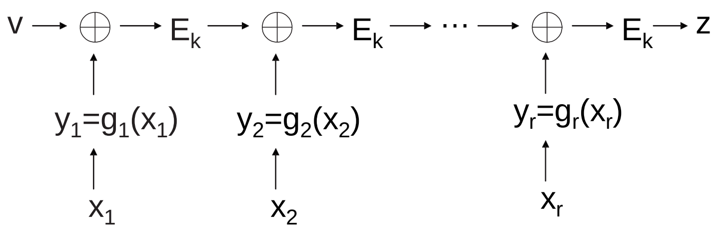

Fig. 1. An illustration of the proposed combining function

A slightly more compact ring signature variant can be obtained by always selecting 0 as the "glue value" v. This variant is also secure, but we prefer the total ring symmetry of our main proposal.

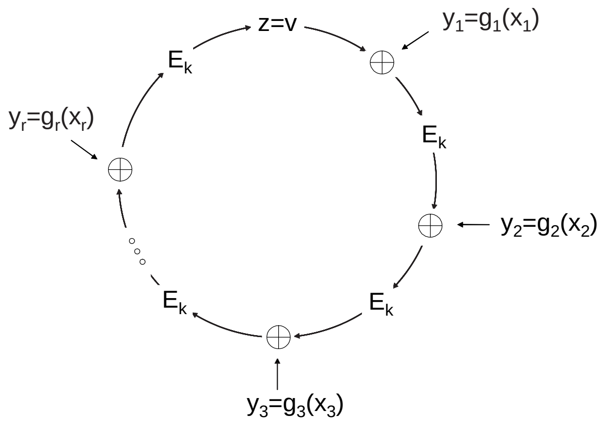

Fig. 2. Ring signatures

4.5 The Ring Signature Scheme

We now formally describe the signature generation and verification procedures:

<u>**Generating a ring signature:**</u>

Given the message m to be signed, a sequence of public keys $P_1, P_2, \ldots, P_r$ of all the ring members (each public key P_i specifies a trapdoor permutation g_i), and a secret key S_s (which specifies the trapdoor information needed to compute g_s^{-1}), the signer computes a ring signature as follows.

1. **Determine the symmetric key:** The signer first computes the symmetric key k as the hash of the message m to be signed:
$$k = h(m)$$
(a more complicated variant computes k as $h(m, P_1, \ldots, P_r)$; however, the simpler construction is also secure.)
2. **Pick a random glue value:** Second, the signer picks an initialization (or "glue") value v uniformly at random from $\{0,1\}^b$.
3. **Pick random x_i's:** Third, the signer picks random x_i for all the other ring members $1 \leq i \leq r$, where $i \neq s$, uniformly and independently from $\{0,1\}^b$, and computes
$$y_i = g_i(x_i) \ .$$
4. **Solve for y_s:** Fourth, the signer solves the following ring equation for y_s:
$$C_{k,v}(y_1, y_2, \ldots, y_r) = v \ .$$
By assumption, given arbitrary values for the other inputs, there is a unique value for y_s satisfying the equation, which can be computed efficiently.
5. **Invert the signer's trapdoor permutation:** Fifth, the signer uses his knowledge of his trapdoor in order to invert g_s on y_s, to obtain x_s:
$$x_s = g_s^{-1}(y_s) \ .$$
6. **Output the ring signature:** The signature on the message m is defined to be the $(2r+1)$-tuple:
$$(P_1, P_2, \ldots, P_r; v; x_1, x_2, \ldots, x_r) \ .$$

Verifying a ring signature:

A verifier can verify an alleged signature
$$(P_1, P_2, \ldots, P_r; v; x_1, x_2, \ldots, x_r) \ .$$
on the message m as follows.

1. **Apply the trapdoor permutations:** First, for $i = 1, 2, \ldots, r$ the verifier computes
$$y_i = g_i(x_i) \ .$$
2. **Obtain k:** Second, the verifier hashes the message to compute the symmetric encryption key k:
$$k = h(m) \ .$$
3. **Verify the ring equation:** Finally, the verifier checks that the y_i's satisfy the fundamental equation:
$$C_{k,v}(y_1, y_2, \ldots, y_r) = v \ . \tag{2}$$
If the ring equation (2) is satisfied, the verifier accepts the signature as valid. Otherwise the verifier rejects.

4.6 Security

The identity of the signer is unconditionally protected with our ring signature scheme. To see this, note that for each k and v the ring equation has exactly $(2^b)^{(r-1)}$ solutions, and all of them can be chosen by the signature generation procedure with equal probability, regardless of the signer's identity. This argument does not depend on any complexity-theoretic assumptions or on the randomness of the oracle (which determines E_k).

The soundness of the ring signature scheme must be computational, since ring signatures cannot be stronger than the individual signature scheme used by the possible signers.

Theorem 1. *The above ring signature scheme is secure against adaptive chosen message attacks in the ideal cipher model (assuming each public key specifies a trapdoor one-way permutation).*

We need to prove that in the ideal cipher model, any forging algorithm A which on input $(P_1, \ldots, P_r)$ can generate with non-negligible probability a new ring signature for m^* by analyzing polynomially many ring signatures for other chosen messages $m \neq m^*$, can be turned into an algorithm B which inverts one of the trapdoor one-way permutations corresponding to $(P_1, \ldots, P_r)$ on a random input, with non-negligible probability.

The basic idea behind the proof is the following: We first show that the ring signing oracle "does not help" A in generating a new signature. This is done by showing that the ring signing oracle can be simulated by an efficient algorithm that has control over the oracles h, E and E^{-1}. We then show that any forgery algorithm (with no ring signing oracle) can be used to invert one of the trapdoor permutations $g_1, \ldots, g_r$ corresponding to the public keys $(P_1, \ldots, P_r)$, on a random input y. This is done by showing how to control the oracles h, E, and E^{-1}, so as to force the "gap" between the output and input values of two cyclically consecutive E_k's along the ring equation of the forgery to be equal to the value y. This forces the forger to close the gap by providing the corresponding $g_i^{-1}(y)$ in the generated signature (for some $i \in \{1, \ldots, r\}$). Since y is a random value which is not known to the forger, the forger cannot "recognize the trap" and refuse to sign the corresponding messages.

In what follows, we prove Theorem 1 by formalizing the above basic idea.

Proof of Theorem 1: Assume that there exists a forging algorithm A, that succeeds in creating a forgery with non-negligible probability. More specifically, algorithm A gets as input a set of random public keys $(P_1, P_2, \ldots, P_r)$ (but not any of the corresponding secret keys), where each P_i specifies a trapdoor one-way permutation g_i. Algorithm A is also given oracle access to h, E, E^{-1}, and to a ring signing oracle. It can work adaptively, querying the oracles at arguments that may depend on previous answers. Eventually, it produces a valid ring signature on a new message that was not presented to the signing oracle, with a non-negligible probability (over the random answers of the oracles and its own random coin tosses). We show that A can be turned into an algorithm

B which inverts one of the trapdoor one-way functions g_i on random inputs y with non-negligible probability.

Algorithm B, on input $g_1, \ldots, g_r$ and a random value $y \in \{0,1\}^b$, uses A on input $(g_1, \ldots, g_r)$ as a black-box (while simulating its oracles), in order to find a value $g_i^{-1}(y)$, for some $i \in \{1, \ldots, r\}$.

We first note that A must query the oracle h with the message that it is actually going to forge (otherwise the probability of satisfying the ring equation becomes negligible). Assume that, with non-negligible probability, A forges the j'th message that it sends to the oracle h. We denote this message by m^*. Algorithm B begins by guessing randomly this index j. Note that B guesses the correct value with non-negligible probability (since A makes in total at most polynomially many queries to the oracle h).

Algorithm B simulates A's oracles in the following way. The oracle h is simulated in the straightforward manner: Whenever A makes a query to h, the query is answered by a uniformly chosen value (unless this query has previously appeared, in which case it is answered the same way as it was before, to ensure consistency).

Algorithm B simulates the ring signing oracle by providing a random vector $(v, x_1, x_2, \ldots, x_r)$ as a ring signature to any query m. It then adjusts the random answers to queries of the form $E_{h(m)}$ and $E^{-1}_{h(m)}$, to support the correctness of the ring equation for these messages. Namely, B chooses randomly $r-1$ values $z_1, \ldots, z_{r-1}$, and sets $E_{h(m)}(v \oplus g_1(x_1)) = z_1$ and $E_{h(m)}(z_i \oplus g_{i+1}(x_{i+1})) = z_{i+1}$, such that $z_r = v$. Similarly, B sets $E^{-1}_{h(m)}(z_1) = v \oplus g_1(x_1)$ and $E^{-1}_{h(m)}(z_{i+1}) = z_i \oplus g_{i+1}(x_{i+1})$. Note that A cannot ask oracle queries that will limit B's freedom of choice, before providing m to the signing oracle, since all the values $v, z_1, \ldots, z_{r-1}$ are chosen randomly by B, and cannot be guessed in advance by A.

In order to simulate the oracles E_k and E_k^{-1}, algorithm B first checks whether $k = h(m^*)$ (where m^* is the j'th query that A sends the oracle h). If $k \neq h(m^*)$ (or if A has not yet queried its j'th query to the oracle h), then B simulates these oracles in the straightforward manner. Namely, each query to E_k or E_k^{-1} is answered randomly, unless the value of this query has already been determined by B, in which case it is answered with the predetermined value. Note that so far, the simulated oracles are statistically close to the real oracles, and thus in particular A cannot distinguish between the real oracles and the simulated oracles.

It remains to simulate the oracles E_k and E_k^{-1}, for $k = h(m^*)$. Recall that the goal of algorithm B is to compute $x_i = g_i^{-1}(y)$, for some i. The basic idea is to slip this value y as the "gap" between the output and input values of two cyclically consecutive E_k's along the ring equation of the final forgery, which forces A to close the gap by providing the corresponding x_i in the generated signature. This basic idea is carried out in the following way.

We note that with overwhelming probability E_k and E_k^{-1} are not constrained up to the point where A queries the oracle h with query m^*. Thus, B will do the following immediately after A queries the oracle h with query m^*. Notice that A must query the oracles E_k or E_k^{-1} about each one of the r symmetric encryptions along the forged ring signature of m^*. Without loss of generality, we may assume that each of these r symmetric encryptions is queried *once* either in the "clockwise" E_k direction or in the "counterclockwise" E_k^{-1} direction, but not in both directions since this is redundant. We distinguish between the following three cases:

Case 1: Each of these r symmetric encryptions is queried in the "clockwise" E_k direction.
Case 2: Each of these r symmetric encryptions is queried in the "counterclockwise" E_k^{-1} direction.
Case 3: Some of these queries are in the "clockwise" E_k direction and some are in the "counterclockwise" E_k^{-1} direction.

We next show how in each of these cases, B can simulate answers to these queries in such a way that A's ring signature of m^* would yield the value $g_i^{-1}(y)$ for some $i \in \{1, \ldots, r\}$.

Case 1: The structure of the ring implies that for every E_k on the ring there exists an E_k that precedes it (we note that the r'th E_k precedes the 1'st E_k). This implies that there must exist an E_k that was queried *before* the E_k that precedes it. Assume that the i'th E_k was queried before the $i-1$'st E_k. B will guess which query corresponds to the i'th E_k and which query corresponds to the $i-1$'st E_k (there are only polynomially many possibilities and thus he will succeed with non-negligible probability). B will provide an answer to the $i-1$'st E_k based on its knowledge of the input to the i'th E_k. More precisely, if the input to the i'th E_k was z, then B will set the output of the $i-1$'st E_k to be $z \oplus y$ (so that the XOR of the values across the gap is the desired y). All other queries are answered randomly (unless the value of this query has already been determined by B, in which case it is answered with the predetermined value).
Case 2: This case is completely analogous to the previous case, and so B behaves accordingly.
Case 3: The structure of the ring is such that for every E_k on the ring there exists an E_k that proceeds it (we note that the 1'st E_k proceeds the r'th E_k). This implies that there exists an E_k that was queried in the "clockwise" direction whereas the proceeding E_k was queried in the "counterclockwise" direction. Assume that the i'th E_k was queried in the "clockwise" direction whereas the $i+1$'st E_k was queried in the "counterclockwise" direction. As in the previous two cases, B will guess which query corresponds to the i'th E_k and which query corresponds to the $i+1$'st E_k (there are only polynomially many possibilities and thus he will succeed with non-negligible probability). B will answer the query to the i'th E_k with a random value z and will answer the query to the $i+1$'st E_k^{-1} with $z \oplus y$ (so that the XOR of the values across the gap is the desired y). All other queries are answered randomly (unless the value of this query has already been determined by B, in which case it is answered with the predetermined value).

Note that since y is a random value, the simulated oracles E_k and E_k^{-1} cannot be distinguished from the real oracles, and therefore, with non-negligible probability, A will output a signature $(v; x_1, \ldots, x_r)$ to a message m^*. Moreover, with non-negligible probability there exists $i \in \{1, \ldots, r\}$ such that $g_i(x_i) = y$, as desired. $\square$

Remark. When the trapdoor one-way functions g_i are RSA functions, we can slightly strengthen the result. Since RSA is homomorphic, we can randomize y by computing $y' = y \cdot t^{e_i} (\bmod\ n_i)$ for a randomly chosen t. By using y' instead of y, we can show that successful forgeries of ring signatures can be used to extract modular roots from particular numbers such as $y = 2$, and not just from random inputs y. This is not necessarily true for other trapdoor one-way functions, since the forger A can intentionally decide not to produce any forgeries in which one of the gaps between cyclically consecutive E functions happens to be 2.

5 Our Ring Signature Scheme (Rabin version)

Rabin's public-key cryptosystem [Rab79] has more efficient signature verification than RSA, since verification involves squaring rather than cubing, which reduces the number of modular multiplications from 2 to 1. However, we need to deal with the fact that the Rabin mapping $f_i(x_i) = x_i^2 \ (\bmod\ n_i)$ is not a permutation over $\mathbf{Z}_{n_i}^*$, and thus only one quarter of the messages can be signed, and those which can be signed have multiple signatures.

We note that Rabin's function, $f_N(x) = x^2 (\bmod\ N)$, is actually a permutation over $\{x : x < \frac{N}{2} \ \wedge \ (\frac{x}{N}) = 1\}$, assuming N is a Blum integer. Moreover, it can be easily extended to be a permutation over $\mathbb{Z}_N^*$ ([G04, Section C.1]). However this permutation is no longer as efficient, since in order to compute it on a value x, one first needs to compute $(\frac{x}{N})$, which is a relatively expensive computation. Moreover, both in the signing and verifying procedures, the number of times that a Jacobi symbol needs to be computed grows linearly with the size of the ring.

Rather than trying to convert Rabin's function to a permutation, we suggest the following natural operational fix: when signing, change your last random choice of x_{s-1} if $g_s^{-1}(y_s)$ is undefined. Since only one trapdoor one-way function has to be inverted, the signer should expect on average to try four times before succeeding in producing a ring signature. The complexity of this search is essentially the same as in the case of regular Rabin signatures, regardless of the size of the ring.

A more important difference is in the proof of unconditional anonymity, which relied on the fact that all the mappings were permutations. When the g_i are not permutations, there can be noticeable differences between the distribution of randomly chosen and computed x_i values in given ring signatures. This could lead to the identification of the real signer among all the possible signers, and can be demonstrated to be a real problem in many concrete types of trapdoor one-way functions.

We overcome this difficulty in the case of Rabin signatures with the following simple observation:

Lemma 1. *Let S be a given finite set of "marbles" and let B_1, B_2, ..., B_n be disjoint subsets of S (called "buckets") such that all non-empty buckets have the same number of marbles, and every marble in S is in exactly one bucket. Consider the following sampling procedure: pick a bucket at random until you find a non-empty bucket, and then pick a marble at random from that bucket. Then this procedure picks marbles from S with uniform probability distribution.*

Proof. Trivial. □

Rabin's functions $f_i(x_i) = x_i^2 \pmod{n_i}$ are extended to functions $g_i(x_i)$ over $\{0,1\}^b$ in the usual way. Both the marbles and the buckets are all the b-bit numbers $u = q_i n_i + r_i$ in which $r_i \in \mathbf{Z}_{n_i}^*$ and $(q_i+1)n_i \leq 2^b$. Each marble is placed in the bucket to which it is mapped by the extended Rabin mapping g_i. We know that each bucket contains either zero or four marbles, and the lemma implies that the sampled distribution of the marbles x_i is exactly the same regardless of whether they were chosen at random or picked at random among the computed inverses in a randomly chosen bucket. Consequently, even an infinitely powerful adversary cannot distinguish between signers and non-signers by analyzing actual ring signatures produced by one of the possible signers.

6 Generalizations and Special Cases

The notion of ring signatures has many interesting extensions and special cases. In particular, ring signatures with $r = 1$ can be viewed as a randomized version of Rabin's signature scheme (or RSA's signature scheme): As shown in Fig. 3, the verification condition can be written as $(x^2 \bmod n) = v \oplus E_{h(m)}^{-1}(v)$. The right hand side is essentially a hash of the message m, randomized by the choice of v.

Ring signatures with $r = 2$ have the ring equation:

$$E_{h(m)}(x_2^2 \oplus E_{h(m)}(x_1^2 \oplus v)) = v$$

(see Fig. 3). A simpler ring equation (which is not equivalent but has the same security properties) is:

$$(x_1^2 \bmod n_1) = E_{h(m)}(x_2^2 \bmod n_2)$$

where the modular squares are extended to $\{0,1\}^b$ in the usual way. This is our recommended method for implementing designated verifier signatures in email systems, where n_1 is the public key of the sender and n_2 is the public key of the recipient.

In regular ring signatures it is impossible for an adversary to expose the signer's identity. However, there may be cases in which the signer himself wants to have the option of later proving his authorship of the anonymized email

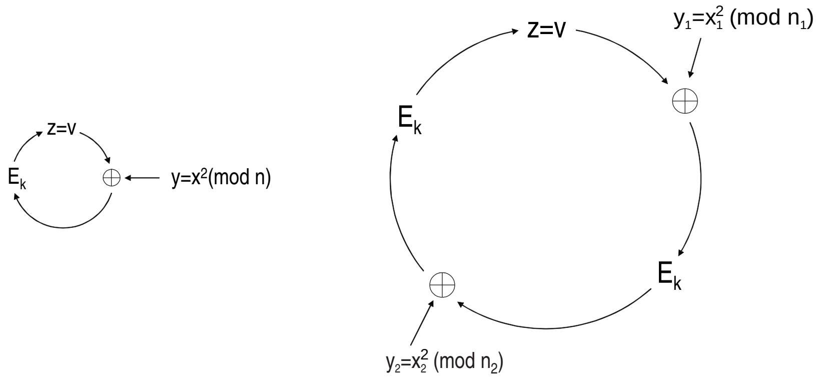

Fig. 3. Rabin-based Ring Signatures with $r = 1, 2$

(e.g., if he is successful in toppling the disgraced Prime Minister). Yet another possibility is that the signer A wants to initially use {A,B,C} as the list of possible signers, but later prove that C is *not* the real signer. There is a simple way to implement these options, by choosing the x_i values for the non-signers in a pseudorandom rather than truly random way. To show that C is *not* the author, A publishes the seed which pseudorandomly generated the part of the signature associated with C. To prove that A *is* the signer, A can reveal a single seed which was used to generate all the non-signers' parts of the signature. The signer A cannot misuse this technique to prove that he is not the signer since his x_i is computed by applying g^{-1} to a random value given to him by the oracle (where g is the trapdoor one-way permutation corresponding to his public key). Thus, his x_i is extremely unlikely to have a corresponding seed. Note that these modified versions can guarantee only computational anonymity, since a powerful adversary can search for such proofs of non-authorship and use them to expose the signer.

A different approach that guarantees unbounded anonymity is to choose the x_i value for each non-signer by choosing a random w_i and letting $x_i = f(w_i)$, where f is a one-way function with the additional property that each element in the range has a pre-image under f. By demonstrating w_i, the signer proves that the i'th ring member is not the signer. Notice that the fact that the signer (which corresponds to the s'th ring member) is computationally bounded, implies that he cannot produce $f^{-1}(x_s)$, and therefore he cannot prove that he himself is not the signer. Moreover, an adversary with unlimited computational power cannot figure out who the signer is since any x_i (including x_s) has a pre-image under f.

7 Followup Papers

In this section we summarize the followup papers on the theory and applications of ring signatures.

Deniable Ring Signature Schemes. In [Na02] Naor defined the notion of *Deniable Ring Authentication*. This notion allows a member of an ad hoc subset of participants (a ring) to convince a verifier that a message m is authenticated by one of the members of the subset without revealing by which one, and the verifier cannot convince a third party that message m was indeed authenticated. Naor also provided an efficient protocol for deniable ring authentication based on any secure encryption scheme. The scheme is interactive. Susilo and Mu [SM03,SM04] constructed *non interactive* deniable ring authentication protocols. They first showed in [SM03] how to use any ring signature scheme and a chameleon hash family to construct a deniable ring signature scheme. In this construction the verifier is assumed to be associated with a pair of secret and public keys (corresponding to the chameleon hash family). They then showed in [SM04] how to use any ring signature scheme and an ID based chameleon hash family [AM04] to construct a deniable ring signature scheme. In this construction the verifier is only assumed to have his ID published.

Threshold and General Access Ring Signature Schemes. A t-threshold ring signature scheme is a ring signature scheme where each ring signature is a proof that at least t members of the ring are confirming the message. In a general access ring signature scheme, members of a set can freely choose any family of sets including their own set, and prove that all members of some set in the access structure have cooperated to compute the signature, without revealing any information about which set it is.

There have been many papers which considered these scenarios. The early work of [CDS94] has already considered this scenario, and showed (using different terminology) that a witness indistinguishable proof (with witnesses that correspond to some monotone access structure), can be combined with the Fiat-Shamir paradigm, to obtain a monotone access ring signature scheme. The work of Naor [Na02] also contains a construction of a general access (and in particular threshold) ring signature scheme. His scheme is interactive and its security is based only the existence of secure encryption schemes. There have been subsequent works which consider the general access scenario, such as [HS04a].

The work of Bresson et. al. [BSS02] contains a construction of a threshold ring signature scheme (proven secure in the Random Oracle Model under the RSA Assumption). Subsequent works which consider the threshold setting are [Wei04,KT03,WFLW03] (where security is proved in the Random Oracle Model).

Identity-based Ring Signature Schemes. Shamir introduced in 1984 the concept of Identity-based (ID-based) cryptography [Sha84]. The idea is that the public-key of a user can be publicly computed from his identity (for example, from a complete name, an email or an IP address). ID-based schemes avoid the necessity of certificates to authenticate public keys in a digital communication system. This is especially desirable in applications which involve a large number of public keys in each execution, such as ring signatures.

The first to construct an ID-based ring signature scheme were Zhang and Kim [ZK02]. Its security was analyzed in [Her03], based on bilinear pairings in the Random Oracle Model. Subsequent constructions of ID-based ring signatures appear in [HS04b,LW03a,AL03,TLW03,CYH04].

Identity-based Threshold Ring Signature Schemes. ID-based threshold ring signature schemes proven secure in the Random Oracle Model, under the bilinear pairings were constructed in [CHY04,HS04c]. This was extended in [HS04c], to a general access setting, where any subset of users S can cooperate to compute an anonymous signature on a message, on behalf of any family of users that includes S.

Separable Ring Signature Schemes. A ring signature scheme is said to be separable if all participants can choose their keys independently with different parameter domains and for different types of signature schemes. Abe et. al. [AOS02] were the first to address the problem of constructing a separable ring signature scheme. They show how to construct a ring signature scheme from a mixture of both trapdoor-type signature schemes (such as RSA based) and three-move-type signature schemes (such as Discrete Log based). This was extended in [LWW03] to the threshold setting.

Linkable Ring Signature Schemes. The notion of linkable ring signatures, introduced by Liu et al. [LWW04], allows anyone to determine if two ring signatures are signed by the same group member. In [LWW04] they also presented a linkable ring signature scheme that can be extended to the threshold setting. Their construction was improved in [TWC+04], who presented a separable linkable threshold ring signature scheme.

Verifiable Ring Signature Schemes. Lv and Wang [LW03b] formalized the notion of verifiable ring signatures, which has the following additional property: if the actual signer is willing to prove to a recipient that he signed the signature, then the recipient can correctly determine whether this is the fact. We note that this additional property was considered in our (original) work, and as was mentioned in Section 6, we showed that this property can be obtained by choosing the x_i values for the non-signers in a pseudorandom rather than a truly random way.

Accountable Ring Signaure Schemes. An accountable ring signature scheme, a notion introduces by Xu and Yung [XY04], ensures the following: anyone can verify that the signature is generated by a user belonging to a set of possible signers (that may be chosen on-the-fly), whereas the actual signer can nevertheless be identified by a designated trusted entity. Xu and Yung [XY04] also presented a framework for constructing accountable ring signatures. The framework is based on a compiler that transforms a traditional ring signature scheme into an accountable one.

Short Ring Signature Schemes. Dodis et. al. [DKNS04] were the first to construct a ring signature scheme in which the length of an "actual signature" is independent of the size of the ad hoc group (where an "actual signature" does not include the group description). We note that in all other constructions that we are aware of, the size of an "actual signature" is at least linear in the size of the group. Their scheme was proven secure in the Random Oracle Model assuming the existence of accumulators with one-way domain (which in turn can be based on the Strong RSA Assumption).

Ring Authenticated Encryption. An authenticated encryption scheme [LRCK04] allows the verifier to recover and verify the message simultaneously. Lv et al. [LRCK04] introduced a new type of authenticated encryption, called ring authenticated encryption, which loosely speaking, is an authenticated encryption scheme where the verifiability property holds with respect to a ring signature scheme.

References

[AL03] Amit K. Awasthi and Sunder Lal. ID-based Ring Signature and Proxy Ring Signature Schemes from Bilinear Pairings. In Cryptology ePrint Archive: Report 2004/184.

[AM04] G. Ateniese and B. de Medeiros. Identity-Based Chameleon Hash and Applications. In *Financial Cryptography 2004*, pages 164-180.

[AOS02] Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n Signatures from a Variety of Keys. In *ASIACRYPT 2002, LNCS 2501*, pages 415-432.

[BGLS03] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and Verfiably Encrypted Signatures from Bilinear Maps. In *E. Biham, ed., Proceedings of Eurocrypt 2003, vol. 2656 of LNCS*, pages 416-432.

[BR93] M. Bellare and P. Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security 1993*, pages 62-73.

[BSS02] Emmanuel Bresson, Jacque Stern, and Michael Szydlo. Threshold Ring Signatures and Applications to Ad-Hoc Groups (Extended abstract). In *Advances in Cryptology - Proceedings of CRYPTO '02, LNCS 2442*, pages 465-480.

[Cam97] Jan Camenisch. Efficient and Generalzied Group Sigmatures. In *Advances in Cryptology - Eurocrypt 97*, pages 465-479.

[Ch81] David Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. In *Communications of the ACM 24(2)*, pages 84-88, 1981.

[Ch88] David Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. In *Journal of Cryptology 1(1),* pages 65-75, 1988.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO '94*, pages 174–187, Berlin, 1994. Springer-Verlag. Lecture Notes in Computer Science Volume 839.

[CHY04] Sherman S.M. Chow, Lucas C.K.Hui, and S.M.Yiu. Identity Based Threshold Ring Signature. In *Cryptology ePrint Archive: Report 2004/179.*

[CV91] David Chaum and Eugène Van Heyst. Group signatures. In D.W. Davies, editor, *Advances in Cryptology — Eurocrypt '91*, pages 257–265, Berlin, 1991. Springer-Verlag. Lecture Notes in Computer Science No. 547.

[CYH04] Sherman S.M. Chow, S.M. Yiu, and Lucas C.K. Hui. Efficient Identity Based Ring Signature. In *Cryptology ePrint Archive: Report 2004/327.*

[DH76] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22:644–654, November 1976.

[DKNS04] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup Anonymous Identification in Ad-Hoc Groups. In em EUROCRYPT 2004, LNCS 3027, pages 609-626. Springer-Verlag, 2004.

[G04] O. Goldreich. *Foundations of Cryptography: Volume 2 – Basic Applications*. Cambridge University Press, 2004.

[GRS99] David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Onion Routing for Anonymous and Private Internet Connections. In *Communications of the ACM 42(2)*, pages 39-41, 1999.

[Her03] Javier Herranz. A Formal Proof of Security of Zhang and Kim's ID-Based Ring Signature Scheme. In *WOSIS 2004*, pages 63-72.

[HS03] J. Herranz and G. Saez. Forking Lemmas in the Ring Signatures' Scenario. In *Cryptology ePrint Archive: Report 2003/067.*

[HS04a] J. Herranz and G. Saez. Ring Signature Schemes for General Ad-Hoc Access Structures. In *Proceedings of ESAS 2004*, pages 54-65.

[HS04b] J. Herranz and G. Saez. New Identity-Based Ring Signature Schemes. In *Information and Communications Security (ICICS 2004)*, pages 27-39. Springer-Verlag, 2004.

[HS04c] J. Herranz and G. Saez. Distributed Ring Signatures for Identity-Based Scenarios. In *Cryptology ePrint Archive: Report 2004/190.*

[HW79] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford, fifth edition, 1979.

[JSI96] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In Ueli Maurer, editor, *Advances in Cryptology - EuroCrypt '96*, pages 143–154, Berlin, 1996. Springer-Verlag. Lecture Notes in Computer Science Volume 1070.

[KT03] Hidenori Kuwakado, Hatsukazu Tanaka. Threshold Ring Signature Scheme Based on the Curve. In *IPSJ JOURNAL Abstract Vol.44*, pages 8-32.

[LR88] M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Computing*, 17(2):373–386, April 1988.

[LRCK04] J. Lv, K. Ren, X. Chen and K. Kim. Ring Authenticated Encryption: A New Type of Authenticated Encryption. In *The 2004 Symposium on Cryptography and Information Security, vol. 1/2*, pages 1179-1184.

[LW03a] C.Y. Lin and T. C. Wu. An Identity Based Ring Signature Scheme from Bilinear Pairings. In *Cryptology ePrint Archive, Report 2003/117, 2003.*

[LW03b] J. Lv, X. Wang. Verifiable Ring Signature. In *Proc. of DMS 2003 - The 9th International Conference on Distribted Multimedia Systems*, pages 663-667, 2003.

[LWW03] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. A Separable Threshold Ring Signature Scheme. In *ICISC 2003, LNCS 2971*, pages 12-26.

[LWW04] J. K. Liu, V. K. Wei, and D. S. Wong. Linkable Spontaneous Anonymous Group Signatures for Ad Hoc Groups (Extended Abstract). In *ACISP 2004 , volume 3108 of LNCS*, pages 325-335, Springer-Verlag, 2004.

[Na02] Moni Naor. Deniable Ring Authentication. In *CRYPTO 2002, LNCS 2442*, pages 481-498. Springer-Verlag, 2002.

[Rab79] M. Rabin. Digitalized signatures as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, January 1979.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[Sha84] Adi Shamir Identity Based Cryptosystems and Signature Schemes. In *Proceedings of CRYPTO 84, LNCS 196*, pages 47-53. Springer-Verlag, 1984.

[SCPY94] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. On monotone formula closure of SZK. In *Proc. 35th FOCS*, pages 454–465. IEEE, 1994.

[SM03] Willy Susilo and Yi Mu. Non-Interactive Deniable Ring Authentication. In *the 6th International Conference on Information Security and Cryptology (ICISC 2003),* pages 397-412, 2003.

[SM04] Willy Susilo and Yi Mu. Deniable Ring Authentication Revisited. In *Applied Cryptography and Network Security (ACNS 2004), LNCS 3089*, pages 149-163. Springer-Verlag, 2004.

[TLW03] Chunming Tang, Zhupjun Liu, and Mingsheng Wang. An Improved Identity-Based Ring Signature Scheme from Bilinear Pairings. In *NM Research Preprints*, pages 231-234. MMRC, AMSS, Academia, Sinica, Beijing, No. 22, December, 2003.

[TWC+04] P. P. Tsang, V. K. Wei, T. K. Chan, M. H. Au, J. K. Liu, and D. S. Wong. Separable Linkable Threshold Ring Signatures. In *INDOCRYPT 2004*, 384-398.

[Wei04] Victor K. Wei. A Bilinear Spontaneous Anonymous Threshold Signature for Ad Hoc Groups. In *Cryptology ePrint Archive: Report 2004/039.*

[WFLW03] Ducan S. Wong, Karyin Fung, Joseph K. Liu, and Victor K. Wei. On the RS-Code Construction of Ring Signature Schemes and a Threshold Setting of RST. In *Information and Communications Security (ICICS'03), LNCS 2836*, pages 34-46. Springer-Verlag, 2003.

[XY04] Shouhuai Xu and Moti Yung. Accountable Ring Signatures: A Smart Card Approach. In *Sixth Smart Card Research and Advanced Application IFIP Conference*, pages 271-286.

[ZK02] Fangguo Zhang and Kwangjo Kim. ID-Based Blind Signature and Ring Signature from Pairings. In *ASIACRYPT 2002, LNCS 2501*, pages 533-547. Springer-Verlag, 2002