ACCOUNTABLE RING SIGNATURES: A SMART CARD APPROACH

Shouhuai Xu
Department of Computer Science, University of Texas at San Antonio
shxu@cs.utsa.edu

Moti Yung
Department of Computer Science, Columbia University
moti@cs.columbia.edu

Abstract

Ring signatures are an important primitive for protecting signers' privacy while ensuring that a signature in question is indeed issued by some qualified user. This notion can be seen as a generalization of the well-known notion of group signatures. A group signature is a signature such that a verifier can establish its validity but not the identity of the actual signer, who can nevertheless be identified by a designated entity called group manager. A ring signature is also a signature such that a verifier can establish its validity but not the identity of the actual signer, who indeed can *never* be identified by any party. An important advantage of ring signatures over group signatures is that there is no need to pre-specify rings or groups of users.

In this paper, we argue that the lack of an *accountability* mechanism in ring signature schemes would result in severe consequences, and thus accountable ring signatures might be very useful. An accountable ring signature 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 – a system-wide participant independent of any possible ring of users. Further, we present a system framework for accountable ring signatures. The framework is based on a compiler that transforms a traditional ring signature scheme into an accountable one. We also conduct a case study by elaborating on how a traditional ring signature scheme is transformed into an accountable one while assuming a weak trust model.

1. Introduction

The notion of ring signatures was formally introduced to resolve the following problem [26]: 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. On one hand, it should be clear that some straightforward solutions would not really solve the problem. It seems that even group signatures [14] may not suffice because they assume that the cabinet has deployed a group signature scheme, which means that there has been a designated entity (called group manager) specific to the cabinet. Even if the cabinet has deployed a group signature scheme, Bob still cannot send the journalist a group signature on the message, because it is likely that the group manager is under the control of the Prime Minister. On the other hand, ring signatures sufficiently solve the above problem because they allow Bob to send a message to the journalist without assuming any manager (meaning that the Prime Minister has absolutely no way to figure out that the signature is produced by Bob), where the message is accompanied with a ring signature indicating that it is really from a cabinet member.

In this paper, we argue that the lack of an *accountability* mechanism in a ring signature scheme would result in severe consequences, because Bob could rumor anything without being held accountable. Therefore, it is desirable to have accountable ring signatures while assuming no manager (an advantage of ring signatures over group signatures) and being able to hold misbehaving parties accountable (an advantage of group signatures over ring signatures) simultaneously. Moreover, it would also be desirable that such a scheme can be seamlessly integrated with a standard public key infrastructure (PKI).

1.1 Our Contributions

We introduce the concept of accountable ring signatures. An accountable ring signature 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 – a system-wide participant independent of any possible set of users. Accountable ring signatures would be more appropriate than ring signatures for real-world utilization because: (1) a message accompanied with an accountable ring signature would be more reliable than a message accompanied with a tra-

ditional unaccountable ring signature, and (2) the information-theoretic anonymity provided by traditional ring signature schemes would hurdle their deployment in the real world – a similar scenario has happened in the context of anonymous e-cash schemes [13, 29].

Further, we present a system framework for accountable ring signatures. The framework is based on a compiler that transforms a traditional ring signature scheme into an accountable one while utilizing tamper-resistant smart cards, and can be seamlessly integrated with a standard PKI (including the traditional application of smart cards for protecting people's private keys, and the certificate revocation methods). We also conduct a case study to show how a traditional ring signature scheme is transformed into an accountable one, while assuming a *weak* trust model. This is particularly important in settings where smart cards may not be completely trusted (e.g., the manufacturers may be interested in getting information about the users' cryptographic keys via various subliminal channels, although they will not compromise the tamper-resistance of the smart cards – after all, that's how they make revenues).

1.2 Related Work

The notion of group signatures was introduced in [14]. A group signature can be seen as a normal digital signature with the following extra properties: anyone can verify that a signature is generated by a legitimate group member, while the actual signer can nevertheless be identified by a designated trusted entity called group manager. The basic idea underlying most group signature schemes is the following: In order for a group member (Alice) to sign a message, she needs to construct an *authorization-proof* to show that she has a legitimate membership certificate, and an *ownership-proof* to demonstrate knowledge of the secret corresponding to the membership certificate. Group signatures have been intensively investigated in the literature early on (cf. [15, 11, 10, 3, 9, 28] and the references therein). However, a group signature scheme requires an initialization procedure for specifying a group, which may not be possible or desirable under certain circumstances.

The notion of ring signatures was explicitly introduced in [26], although the basic idea had been mentioned several times [14, 8, 17, 18]. For example, [17] showed how one can produce witness-indistinguishable interactive proofs, which can be naturally transformed into a ring signature scheme via the Fiat-Shamir heuristics [20]. The scheme presented in [26] is based on the RSA public key cryptosystem [25], and its security is analyzed in the ideal cipher model [4]. This scheme was improved in

[7] where the authors presented a variant ring signature scheme whose security is analyzed in the random oracle model [5]. [1] proposed a general ring signature scheme that accommodates both RSA and discrete logarithm based cryptosystems. However, none of the above-mentioned ring signature schemes provides a revocation mechanism to hold a misbehaving party accountable.

In regard to utilizing smart cards, [12] showed how to implement a group signature scheme based on tamper-resistant smart cards. The basic idea underlying [12] is as follows: A group manager chooses two pairs of public and private keys, namely $\langle pk_1, sk_1 \rangle$ with respect to a public key cryptosystem and $\langle pk_2, sk_2 \rangle$ with respect to a digital signature scheme, where both pk_1 and pk_2 are publicly known. Each group member has a unique identity U and holds a tamper-resistant smart card that is equipped with pk_1 and sk_2. A group signature is indeed a digital signature obtained by applying sk_2 to the concatenation of a message m and a ciphertext c that is the encryption of U under public key pk_1 (i.e., c allows the group manager to revoke anonymity of the actual signer of a given group signature). Besides that this scheme is an implementation of traditional group signatures (i.e., the groups are pre-specified), it has the drawback that compromise of a single smart card (after all, tamper-resistance is still heuristic [2]) or revoking a user's membership (e.g., due to group dynamics) would force all the rest of smart cards to participate in a key-update process. As we will see, we achieve a higher assurance that a compromise of a smart card does not result in such a complex procedure, while making the revocation of a smart card or certificate completely transparent to the rest smart cards.

1.3 Organization

In Section 2, we briefly review some cryptographic preliminaries. In Section 3, we define accountable ring signatures. In Section 4 we present a system framework for accountable ring signatures, which is based on a generic compiler that transforms a traditional ring signature scheme into an accountable one in a *strong* trust model. In Section 5 we conduct a case study on transforming a traditional ring signature scheme into an accountable one while assuming a *weak* trust model. We conclude in Section 6.

2. Cryptographic Preliminaries

Besides using standard public key cryptosystems, $\mathcal{E} = (GEN, E, D)$, that are semantically secure [22], and digital signature schemes, $\mathcal{S} = (GEN, S, V)$, that are existentially unforgeable under adaptive chose-

message attacks [23], we use a threshold symmetric key cryptosystem specified below.

Threshold symmetric key cryptosystems. In order to revoke anonymity of accountable ring signatures while ensuring optimal resilience, we need to facilitate the following task: an entity (called dealer) encrypts a message so that the ciphertext can be decrypted by a set of a constant number n of servers $\{S_1, \cdots, S_n\}$, of which at most $t = \lfloor \frac{n-1}{2} \rfloor$ servers can be corrupted.[1] Below we give a concrete threshold symmetric key cryptosystem DSKC=(DSKC.Setup, DSKC.Enc, DSKC.Dec), which is adapted from a secure distributed pseudorandom function [24, 21] that can be based on a block cipher for performance reason.

DSKC.Setup. Define $d = \binom{n}{t}$, and define d subsets $\{\mathcal{S}_j\}_{j=1}^{d}$ as all the subsets of $n-t$ of the n servers. The dealer chooses a key $\Theta = \{\theta_1, \cdots, \theta_d\}$, and defines a function F_Θ as $F_\Theta(x) = \oplus_{j=1}^{d} f_{\theta_j}(x)$, where $f_\alpha(\cdot) : \{0,1\}^\kappa \times \{0,1\}^l \rightarrow \{0,1\}^l$ is a pseudorandom function keyed by α of length κ. Finally, let all the servers in subset $\mathcal{S}_j$ hold θ_j, which means that server S_i $(1 \leq i \leq n)$ holds $\Theta_i = \{\theta_j | S_i \in \mathcal{S}_j\}$ such that $\bigcup_{i=1}^{n} \Theta_i = \{\theta_1, \cdots, \theta_d\} = \Theta$. Note that the dealer always holds Θ.

DSKC.Enc. The dealer encrypts a message m of length l as follows: choose $R \in_R \{0,1\}^l$ and set the ciphertext $\phi = \langle R, c = m \oplus F_\Theta(R) \rangle$.

DSKC.Dec. Given a ciphertext $\phi = \langle R, c \rangle$, the servers jointly decrypt it as follows: let S_i $(1 \leq i \leq n)$ contribute $f_{\theta_j}(R)$ for every $\theta_j \in \Theta_i$, then a straightforward algorithm can decide the correct evaluations and recover the plaintext m.

3. Definition of Accountable Ring Signatures

In this section we review the definition of traditional (i.e., unaccountable) ring signatures and then give a definition of accountable ring signatures.

3.1 Definition of Ring Signatures

A ring signature scheme (RS) specifies a set of possible signers and a proof that is intended to convince a verifier that the actual signer of the

[1] Technically, a threshold message authentication scheme also suffices, but it would incur a linear communication complexity because of the size of the signatures. Note that a threshold message authentication scheme can be easily derived from a threshold symmetric key cryptosystem.

signature belongs to the set, while preserving her anonymity. A crucial property of ring signatures is that they are setup-free (i.e., there is no need for any special procedure beyond a standard PKI). Note that the size of a ring signature grows linearly with the size of the specified ring, given that the dynamic ring membership is not known in advance and has to be provided as part of a signature.

DEFINITION 1 *Suppose each user U_s is associated (via a PKI or certificate) with a pair of public and private keys $\langle pk_{U_s}, sk_{U_s}\rangle$. A ring signature scheme* RS *has the following two procedures.*

RS.Sign. *This is a probabilistic algorithm which, on input a message m, the public keys $pk_{U_1}, \cdots, pk_{U_z}$ of z ring members (i.e., possible signers) $U_1, \cdots, U_z$, and the secret key sk_{U_s} of U_s ($1 \leq s \leq z$), produces a ring signature σ (including $pk_{U_1}, \cdots, pk_{U_z}$) for the message m.*

RS.Ver. *This is a deterministic algorithm which takes as input (m, σ) and outputs either* `TRUE` *or* `FALSE`.

DEFINITION 2 *A ring signature scheme* RS *is said to be secure if it possesses the following properties:*

`Correctness`. *A ring signature produced by an honest user is always accepted as valid with respect to the specified ring.*

`Unforgeability`. *It must be infeasible for any user, except for a negligible probability, to generate a valid ring signature with respect to a ring he does not belong to.*

`Anonymity`. *No verifier is able to guess the actual signer's identity with probability greater than $1/z + \epsilon$, where z is the size of the ring and ϵ is a negligible function.*

3.2 Definition of Accountable Ring Signatures

DEFINITION 3 *Suppose each user U_s is associated with a pair of public and private keys $\langle pk_{U_s}, sk_{U_s}\rangle$. An accountable ring signature scheme (*ARS*) consists of the following algorithms.*

ARS.Sign. *This is a probabilistic algorithm which, on input a message m, the public keys $pk_{U_1}, \cdots, pk_{U_z}$ of z respective ring members $U_1, \cdots, U_z$, the secret key sk_{U_s} of U_s ($1 \leq s \leq z$), and some other information that will be used to produce a tag ϕ that allows some designated authority to recover the identity U_s, produces a ring signature σ (including $pk_{U_1}, \cdots, pk_{U_z}$ and ϕ) for the message m.*

ARS.Ver. *This is a deterministic algorithm which takes as input* (m, σ) *and outputs either* `TRUE` *or* `FALSE`.

ARS.Open. *This is a deterministic algorithm which takes as input* (m, σ) *as well as the authority's secret, and outputs the identity of the actual signer* U_s.

DEFINITION 4 *An accountable ring signature scheme* ARS *is said to be secure if it possesses the following properties:*

`Correctness.` *A ring signature produced by an honest user is always accepted as valid with respect to the specified ring.*

`Unforgeability.` *It must be infeasible for any user, except for a negligible probability, to generate a valid ring signature with respect to a ring he does not belong to.*

`Anonymity.` *No verifier should be able to guess the actual signer's identity with probability greater than* $1/z + \epsilon$, *where* z *is the size of the ring, and* ϵ *is a negligible function.*

`Unlinkability.` *No verifier is able to associate two ring signatures produced by the same user, even if the user's identity remains to be unknown to the adversary.*[2]

`Revocability.` *There is an anonymity revocation authority that is able to identify the actual signer of a given ring signature.*

`No-misattribution.` *It is infeasible for the anonymity revocation authority to convince an honest verifier that a given ring signature is produced by an honest user, who is actually not the signer though.*

`Coalition-resistance.` *A colluding subset of users (even all users) cannot generate a ring signature that the anonymity revocation authority cannot trace back to at least one of the colluding users.*

4. A Compiler

The basic idea underlying the compiler is to introduce into the system model an Anonymity Revocation Authority (ARA) that recovers the identity of the actual signer of a given accountable ring signature when the need arises, and tamper-resistant smart cards that are issued to the

[2]Previous ring signature schemes [26, 7] indeed achieve unconditional anonymity, which immediately imply unlinkability because $\epsilon = 0$. In accountable ring signature schemes, unconditional anonymity is still possible but would render them impractical.

users who want to produce accountable ring signatures. In order to make the compiler concise and generic, here we assume a *strong* trust model (We will show through a case study in Section 5 how one can weaken the trust model, particularly the trust on smart cards, to a more practical level.):

- The Certificate Authority (CA) is trusted in issuing public key certificates, which means that it will not issue a certificate to a user without conducting an appropriate authentication. This assumption is inherited from traditional ring signatures based on a standard PKI.
- The smart cards are tamper-resistant, while we must minimize the damage due to the compromise of a smart card. Such a cautious measure seems necessary given that tamper-resistance is currently only heuristic.
- The smart cards are trusted not to leak any information about its secrets (e.g., it will not utilize any channel to leak the holder's private key to the CA, the ARA, or the holder).
- The ARA is trusted in preserving the users' privacy.

Suppose each user U_s holds a smart card, while the unique identity U_s may be publicly known. The compiler takes as input a secure ring signature scheme RS=($\mathsf{RS.Sign}$, $\mathsf{RS.Ver}$), and outputs a secure accountable ring signature scheme ARS=($\mathsf{ARS.Sign}$, $\mathsf{ARS.Ver}$, $\mathsf{ARS.Open}$) as specified below:

1 There is a setup procedure that is extended from the setup procedure in a standard PKI. (If users would utilize smart cards in the PKI, then this setup procedure does not incur any significant extra complexity.)

   (a) It establish an Anonymity Revocation Authority (ARA), which initializes a crypto-context whereby the identities of the card holders can be "embedded" into signatures in a certain way so that the ARA can recover them.

   (b) Suppose U_s intends to produce accountable ring signatures, the CA gives her a smart card that will generate a pair of public and private keys $\langle pk_{U_s}, sk_{U_s} \rangle$ with respect to an appropriate digital signature scheme, where pk_{U_s} is certified by the CA but sk_{U_s} is known only to the smart card.

2 ARS.Sign works as follows: Suppose U_s $(1 \leq s \leq z)$ wants to generate an accountable ring signature on message m with respect to the ring members' public keys $pk_{U_1}, \cdots, pk_{U_z}$.

   (a) U_s's smart card generates a token ϕ that embeds U_s. We require that the token ϕ does not leak any information (in either a computational or information-theoretic sense) about U_s, but does somehow allow ARA to recover U_s.

   (b) U_s's smart card executes RS.Sign on the message $m||\phi$, where "||" means string concatenation. Denote the output of RS.Sign by σ, which is the resulting accountable ring signature.

3 ARS.Ver is the same as RS.Ver, except that the message is $m||\phi$.

4 ARS.Open works as follows: Given a valid accountable ring signature σ on message $m||\phi$, ARA extracts the identity U_s of the actual signer from ϕ (perhaps via an inversion to the procedure that ϕ is generated).

5. Case Study

In this section we conduct a case study to show how the compiler transforms a secure ring signature scheme RS into a secure accountable ring signature scheme ARS while assuming a *weak* trust model. (Recall that, in order to make the compiler concise and generic, we assumed a strong trust model in Section 4.)

This section is organized as follows. In Section 5.1 we elaborate on the system model. In Section 5.2 we present the input, namely a secure ring signature scheme, and in Section 5.3 we present the output, namely a secure accountable ring signature scheme, whose security is analyzed in the full version of this paper due to space limitation.

5.1 System Model

Participants. There are three explicit categories of participants that are modeled as probabilistic polynomial-time Turing machines.

- As in a standard PKI, we assume that there is a certificate authority (CA) that certifies the users' public keys. (The extension to accommodating multiple CAs is straightforward.) The CA gives smart cards to individual users after an appropriate procedure (e.g., initializing some cryptographic parameters and installing some software programs).

- A user U_s may utilize her public key certificate as in a standard PKI for generating normal signatures. However, if she needs to generate any accountable ring signature, she must apply for a tamper-resistant smart card from the CA. The same pair of public and private keys $\langle pk_{U_s}, sk_{U_s} \rangle$ may be used to produce normal signatures and/or ring signatures. Besides holding a smart card, the user also possesses a computer that typically interacts with her smart card.
- There is an Anonymity Revocation Authority (ARA) that is responsible for identifying the actual signers of accountable ring signatures.

Trust. In order to simplify the system, we claim the following trust relationship (which is weaker than the trust model adopted by the generic compiler in Section 4).

- The smart card hardware is tamper-resistant in the sense that it will erase the secrets stored in it if there is an attempt at breaking into it. However, since the notion of tamper-resistance is only heuristic, we must ensure that the damage due to the compromise of a smart card is minimized. In particular, it would be highly desirable that compromise of a smart card does not allow the adversary to frame any other honest user for generating any accountable ring signature.
- The smart card software (e.g., operating system) is secure in the following sense: (1) it will not tamper with any application software program installed in the smart card; (2) there is no back-door for leaking the secrets stored in the smart card; (3) it provides a secure pseudorandom generator, if necessary.
- The CA will not collude with any user to issue him or her a certificate on a public key that will be accepted as eligible for generating accountable ring signatures. This also implies that the software program installed in the smart card by the CA (e.g., for generating temporary certificates) will not leak any information about the private keys to the card holders.
- The users' computers (including the software programs) are secure. In particular, the randomness output by the software program is guaranteed to be chosen uniformly at random, and there are no Trojan Horses in their computers.
- The ARA is trusted to preserve the anonymity of the users. To mitigate the trust, it is natural to implement the functionality via

a distributed cryptosystem. Again, the ARA cannot be simply based on a tamper-resistant hardware because we must maintain certain access structure for restricting the access to it.

Communication channels. Once a user U_s obtained her smart card from the CA, the smart card cannot communicate with the outside world except U_s's computer. We assume that the communication channel between U_s's computer and smart card is physically secure, and that the channel through which U_s publishes an accountable ring signature is anonymous (the same as what is assumed in any ring signature scheme). All the other communication channels (including the one between U_s and the CA) are public and thus subject to eavesdropping by any party.

Adversary. Given the above trust model, we consider the following potential attacks.

- A dishonest user U_s manages to obtain a certificate for his public key pk_{U_s} corresponding to the private key sk_{U_s} that is known to U_s (e.g., she may try to forge a certificate on pk_{U_s}).
- A dishonest user U_s manages to obtain the private key sk_{U_s} corresponding to the public key pk_{U_s} that is generated by the software program installed by the CA in her smart card and thus certified by the CA.
- Although the CA is trusted to behave honestly in issuing certificates, it may not be trusted in any other sense. For example, the software program installed by the CA for generating "proofs" may not be trustworthy and may intend to leak information via a subliminal channel [27] or a kleptographic channel [30]. As a result, the private key may be completely compromised after further cryptanalysis. A similar channel may be established between a smart card and the ARA to compromise the secrets on the smart card. For simplicity, we assume that the software program on a smart card will not adopt the *halting* strategy [19] to construct a subliminal channel for leaking secrets. By *halting* strategy we mean that the software program installed by the CA decides whether or not to generate a valid message that is requested by the smart card holder. For example, the software program does generate the requested message only when it has embedded certain information into the subliminal channel known to the CA or ARA. We also assume that the software program will not adopt the *delaying* strategy to construct a subliminal channel. By *delaying* strategy we mean that the software program outputs a valid response to

a request from the smart card holder only at the time that coincides with a predefined subliminal time channel. For example, a message generated in the morning meaning the bit of 0, and a message generated in the afternoon meaning the bit of 1. In short, we assume no covert or subliminal channels.

5.2 Input: A Secure Ring Signature Scheme

This scheme is adapted from [7], where it is shown to be secure with respect to Definition 2. Let l, l_b be security parameters, and $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^l$ be a hash function that is modeled as a random oracle. Suppose that each user U_i has a regular signature scheme built on a trapdoor one-way permutation g_i (e.g., RSA) and that the modulus has length $l_b < l$. Since the involved RSA moduli N_i are different, an adaptation has to be made in order to combine them efficiently. This can be done by extending the trap-door permutation g_i over $\mathbb{Z}_{N_i}$ to an permutation g_i' over $\{0,1\}^l$ as follows. For any l-bit input x define non-negative integers q_i and r_i such that $x = q_i N_i + r_i$ and $0 \leq r_i < N_i$. Then, define

$$g_i'(x) = \begin{cases} q_i N_i + g_i(r_i) & \text{if } (q_i+1)N_i \leq 2^l \\ x & \text{otherwise} \end{cases}$$

Intuitively, g_i' is defined by using g_i to operate on the low-order digit of the N_i-ary representation of x, leaving the higher order bits unchanged. The exception is when this might cause a result larger than $2^l - 1$, in which case x is unchanged. If we choose $l - l_b \geq 160$, the chance that a randomly chosen x is unchanged by the extended g_i' becomes negligible. The function g_i' is clearly a one-way trapdoor permutation over $\{0,1\}^l$. The scheme is described below:

RS.Sign. Suppose user U_s $(1 \leq s \leq z)$ is to generate a ring signature with respect to the ring $U_1, \cdots, U_z$.

1 U_s chooses $\delta \in_R \{0,1\}^l$, and computes along the ring as follows (where $z+1$ is treated as 1):

$$\begin{aligned} v_{s+1} &= \mathcal{H}(m, \delta), \\ v_{s+2} &= \mathcal{H}(m, v_{s+1} \oplus g_{s+1}'(r_{s+1})), \\ &\cdots \\ v_s &= \mathcal{H}(m, v_{s-1} \oplus g_{s-1}'(r_{s-1})). \end{aligned}$$

Just before closing the ring, the signer uses her secret key to compute r_s such that $\delta = v_s \oplus g_s'(r_s)$.

2 In order to make the signature anonymous, U_s chooses at random an index i_0 ($1 \leq i_0 \leq z$), and outputs the signature $(i_0, v_{i_0}, r_1, \cdots, r_n)$.

RS.Ver. The verifier accepts if $v_{i_0} = \mathcal{H}(m, v_{i_0-1} \oplus g'_{i_0-1}(r_{i_0-1}))$, where $v_j = \mathcal{H}(m, v_{j-1} \oplus g'_{j-1}(r_{j-1}))$ for $j = i_0 + 1, \cdots, i_0 - 1$.

5.3 Output: A Secure Accountable Ring Signature Scheme

The basic idea underlying the setup procedure is to let a user U_s hold a pair of public and private keys $\langle pk_{U_s}, sk_{U_s} \rangle$ such that sk_{U_s} is only known to U_s's smart card, and let U_s's computer initialize an instance of a distributed symmetric key cryptosystem DSKC so that the card holds Θ and server ARA_i ($1 \leq i \leq n$) holds Θ_i such that $\bigcup_{i=1}^{n} \Theta_i = \Theta$. However, the effort to prevent various possible subliminal channels makes it a bit complicated. Below is the scheme.

1 A user U_s applies for a certificate on her public key pk_{U_s} that is eligible for generating accountable ring signatures, where the corresponding private key sk_{U_s} is known to U_s's smart card but not U_s. Note that a public key is always assumed to be publicly known (e.g., certified by the CA).

   (a) The ARA is implemented by n servers $ARA_1, \cdots, ARA_n$, among them at most $t = \lfloor \frac{n-1}{2} \rfloor$ servers can be corrupted. Suppose ARA_i ($1 \leq i \leq n$) has two pairs of public and private keys: $\langle pk_{ARA_i}, sk_{ARA_i} \rangle$ for a secure public key cryptosystem and $\langle pk'_{ARA_i}, sk'_{ARA_i} \rangle$ for a secure digital signature scheme. Suppose ARA_i maintains a database for storing the secrets that the users will provide.

   (b) U_s applies for a smart card and a certificate for her public key pk_{U_s}.

      i U_s applies for a smart card from the CA, just as she applies for a certificate in a standard PKI (meaning that there is an appropriate authentication procedure).

      ii The CA returns U_s a smart card equipped with a pair of public and private keys $\langle pk_{CARD}, sk_{CARD} \rangle$ with respect to a deterministic signature scheme (for the purpose of avoiding subliminal channels), and $(pk_{ARA_1}, \cdots, pk_{ARA_n})$.

      iii U_s's computer executes DSKC.Setup to obtain $\Theta = (\theta_1, \cdots, \theta_d)$ for U_s's smart card and computer, and Θ_i for ARA_i where $1 \leq i \leq n$. In order to securely send Θ_i to

ARA_i, U_s's computer encrypts Θ_i using pk_{ARA_i}; denote the resulting ciphertext by η_i. Then, U_s's computer sends $(\Theta; \Theta_1, \eta_1, \cdots, \Theta_n, \eta_n)$ to U_s's smart card via the physically secure communication channel, and keeps Θ for itself.

iv U_s's card checks if the Θ_i's and η_i's are correctly generated. If so, U_s's card generates a pair of public and private keys $\langle pk_{U_s}, sk_{U_s}\rangle$ and a signature ρ on $(U_s, pk_{U_s}, \eta_1, \cdots, \eta_n)$ using its private key sk_{CARD}, keeps Θ, and sends $\Upsilon = (pk_{CARD}, U_s, pk_{U_s}, \eta_1, \cdots, \eta_n, \rho)$ to U_s's computer.

v U_s's computer checks the validity of Υ, and forwards it to the CA and the ARA_i's via the public channels.

vi CA checks the validity of ρ and returns a certificate $cert_{U_s}$ on pk_{U_s}. On the other hand, ARA_i $(1 \leq i \leq n)$ verifies the validity of ρ, decrypts η_i to obtain Θ_i, keeps (U_s, Θ_i) in its secret database, and generates (using $sk_{ARA'_i}$) and returns a signature ζ_i on (pk_{CARD}, U_s) back to U_s's computer.

vii U_s's computer sends valid $cert_{U_s}$ and $\{\zeta_i\}_{i=1}^n$ to her smart card, which is now ready for generating accountable ring signatures (and may erase sk_{CARD}).

2 ARS.Sign works as follows. Suppose U_s $(1 \leq s \leq z)$ is to generate an accountable ring signature with respect to a ring consisting of $U_1, \cdots, U_z$, whose certificates have not be revoked.

(a) U_s's computer executes DSKC.Enc to encrypt U_s (for the purpose of avoiding a subliminal channel), and sends the resulting ciphertext $\phi = \langle R, F_\Theta(U_s)\rangle$ to her smart card.

(b) U_s's card checks the validity of ϕ, and then executes RS.Sign to generate a ring signature σ on $m' = m||\phi$. The signature σ is sent back to her computer, which then publishes it via an anonymous channel.

3 ARS.Ver is the same as RS.Ver, except that $m' = m||\phi$.

4 Suppose ϕ is embedded in a valid ring signature σ, ARS.Open works as follows: For $j = 1$ `to` z, the ARA's jointly execute DSKC.Dec to check whether ϕ is an encryption of U_j. If so, U_j is the actual signer.

THEOREM 5 *Suppose that the adopted signature schemes are existentially unforgeable under adaptive chosen-message attacks, that adopted*

public key cryptosystems are semantically secure, and that the security parameters are chosen such that the birthday attack is avoided. The above ARS *scheme is secure with respect to Definition 4.*

The proof is left to the full version of this paper.

Remark. It is clear that our accountable ring signature scheme is essentially as efficient as the underlying ring signature scheme, except that it additionally involves some symmetric key operations (i.e., evaluating d pseudorandom functions). Although the computational overhead on a card may not be an issue, the communication overhead may be a serious problem (e.g., $z = 100$). We observe that both computational and communication overhead on a smart card can be substantially released, because we can let U_s's computer execute ARS.Sign until the point that it is supposed to close the ring, and then send $(m||\phi, \delta, v_{s-1}, r_{s-1})$ to U_s's card. U_s's card checks if ϕ is correctly computed. If so, it returns r_s such that $\delta = v_s \oplus g'_s(r_s)$, where $v_s = H(m||\phi, v_{s-1} \oplus g'_{s-1}(r_{s-1}))$. We stress that this performance optimization does not jeopardize security, because $\delta \oplus v_s$ is uniformly distributed over $\{0,1\}^l$ and U_s is entitled to query the oracle for inverting g'_s at such points.

6. Conclusion

We argued that accountable ring signatures would be very useful. We presented a system framework for accountable ring signatures, and conducted a case study showing how a traditional ring signature scheme is transformed into an accountable ring signature scheme while assuming a weak trust model.

References

[1] M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n Signatures from a Variety of Keys. Asiacrypt'02.

[2] R. Anderson and M. Kuhn. Low Cost Attacks on Tamper Resistant Devices. Security Protocol'97.

[3] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. Crypto'00.

[4] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated Key Exchange Secure against Dictionary Attacks. Eurocrypt'00.

[5] M. Bellare and P. Rogaway. Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols. ACM CCS'93.

[6] D. Boneh and M. Franklin. Efficient Generation of Shared RSA Keys (Extended Abstract). Crypto'97.

[7] E. Bresson, J. Stern, and M. Szydlo. Threshold Ring Signatures and Applications to Ad-Hoc Groups. Crypto'02.

[8] J. Camenisch. Efficient and Generalized Group Signatures. Eurorypt'97.

[9] J. Camenisch and A. Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. Crypto'02.

[10] J. Camenisch and M. Michels. A Group Signature Scheme based on an RSA-variant. Tech. Report RS-98-27, BRICS. Preliminary version appeared at Asiacrypt'98.

[11] J. Camenisch and M. Stadler. Efficient Group Signature Schemes for Large Groups (Extended Abstract). Crypto'97.

[12] S. Canard and M. Girault. Implementing Group Signature Schemes with Smart Cards. Cardis'02.

[13] D. Chaum. Blind Signatures for Untraceable Payments. Crypto'82.

[14] S. Chaum and E. van Heyst. Group Signatures. Eurocrypt'91.

[15] L. Chen and T. Pedersen. New Group Signature Schemes. Eurocrypt'94.

[16] J. Coron, M. Joye, D. Naccache, and P. Paillier. Universal Padding Schemes for RSA. Crypto'02.

[17] R. Cramer, I. Damgard, and B. Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. Crypto'94.

[18] A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung. On Monotone Formula Closure of SZK. FOCS'94. pp 454–465.

[19] Y. Desmedt. Simmons' Protocol Is Not Free of Subliminal Channels. Computer Security Foundation Workshop'96.

[20] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Crypto'86.

[21] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. J. ACM, Vol. 33, No. 4, 1986, pp 210-217.

[22] S. Goldwasser and S. Micali. Probabilistic Encryption. JCSS, 1984.

[23] S. Goldwasser, S. Micali, R. Rivest. A Digital Signature Scheme Secure against Adaptive Chosen-message Attacks. SIAM J. Computing, 17(2), 1988.

[24] M. Naor, B. Pinkas, and O. Reingold. Distributed Pseudo-Random Functions and KDCs. Eurocrypt'99.

[25] R. L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystem. Communication of the ACM, Vol. 21, No. 2, 1978.

[26] R. Rivest, A. Shamir, and Y. Tauman. How to Leak a Secret. Asiacrypt'01.

[27] G. J. Simmons. The History of Subliminal Channels. IEEE Journal on Selected Areas in Communication, vol. 16, no. 4, May 1998.

[28] G. Tsudik and S. Xu. Accumulating Composites and Improved Group Signing. Asiacrypt'03.

[29] B. von Solms and D. Naccache, On Blind Signatures and Perfect Crimes, Computers & Security, 11(6), 1992, 581-583.

[30] A. Young and M. Yung. Kleptography: using Cryptography Against Cryptography. Crypto'97.