Ring Signatures: Stronger Definitions, and Constructions without Random Oracles

Adam Bender* Jonathan Katz*† Ruggero Morselli*‡

Abstract

Ring signatures, first introduced by Rivest, Shamir, and Tauman, enable a user to sign a message so that a *ring* of possible signers (of which the user is a member) is identified, without revealing exactly *which member* of that ring actually generated the signature. In contrast to group signatures, ring signatures are completely "ad-hoc" and do not require any central authority or coordination among the various users (indeed, users do not even need to be aware of each other); furthermore, ring signature schemes grant users fine-grained control over the level of anonymity associated with any particular signature.

This paper has two main areas of focus. First, we examine previous definitions of security for ring signature schemes and suggest that most of these prior definitions are too weak, in the sense that they do not take into account certain realistic attacks. We propose new definitions of anonymity and unforgeability which address these threats, and then give separation results proving that our new notions are strictly stronger than previous ones. Next, we show two constructions of ring signature schemes in the standard model: one based on generic assumptions which satisfies our strongest definitions of security, and a second, more efficient scheme achieving weaker security guarantees and more limited functionality. These are the first constructions of ring signature schemes that do not rely on random oracles or ideal ciphers.

*Dept. of Computer Science, University of Maryland. {bender,jkatz,ruggero}@cs.umd.edu

†This research was supported in part by NSF Trusted Computing Grants #0310499 and #0310751, NSF-ITR #0426683, and NSF CAREER award #0447075.

‡Supported by NSF Trusted Computing Grant #0310499 and NSF-ITR #0426683.

1 Introduction

Ring signatures enable a user to sign a message so that a "ring" of possible signers (of which the user is a member) is identified, without revealing exactly which member of that ring actually generated the signature. This notion was first formally introduced by Rivest, Shamir, and Tauman [16] (though the concept was also present in earlier work [5, 6]), and ring signatures — along with the related notion of ring/ad-hoc identification and authentication schemes — have been studied extensively since then [4, 15, 1, 19, 8, 18, 14, 2]. Ring signatures are related, but incomparable, to the notion of group signatures [5]. On the one hand, group signatures have the additional feature that the anonymity of a signer can be revoked (i.e., the signer can be traced) by a designated group manager. On the other hand, ring signatures allow greater flexibility: no centralized group manager or coordination among the various users is required (indeed, users may be unaware of each other at the time they generate their public keys); rings may be formed completely "on-the-fly" and in an ad-hoc manner; and users are given fine-grained control over the level of anonymity associated with any particular signature (via selection of an appropriate ring).

Ring signatures naturally lend themselves to a variety of applications which have been suggested already in previous work (see especially [16, 15, 8, 2]). The original motivation was to allow secrets to be leaked anonymously. Here, for example, a high-ranking government official can sign information with respect to the ring of *all* similarly high-ranking officials; the information can then be verified as coming from *someone* reputable without exposing the actual signer. Ring signatures can also be used to provide a member of a certain class of users access to a particular resource without explicitly identifying this member; note that there may be cases when third-party verifiability is required (e.g., to prove that the resource has been accessed some number of times, as in web metering) and so ring signatures, rather than ad-hoc identification schemes, are needed. Finally, we mention the application to designated-verifier signatures [13] especially in the context of e-mail. Here, ring signatures enable the sender of an e-mail to sign the message with respect to the ring containing the sender and the receiver; thus, the receiver is assured that the e-mail originated from the sender but cannot prove this to any third party. We remark that for this latter application it is sufficient to use a ring signature scheme which supports only rings of size two.

1.1 Our Contributions in Relation to Previous Work

This paper focuses on both definitions and constructions. We summarize our results in each of these areas, and relate them to prior work.

Definitions of security. Prior work on ring signature/identification schemes provides definitions of security that are either rather informal or seem (to us) unnaturally weak, in that they do not address what seem (to us) to be valid security concerns. One example is the failure to consider the possibility of *adversarially-chosen* public keys. Specifically, both the anonymity and unforgeability definitions in most prior work assume that honest users always sign with respect to rings consisting entirely of *honestly-generated* public keys; no security is provided if users sign with respect to a ring containing even one adversarially-generated public key. Clearly, however, a scheme which is not secure in the latter case is not very useful; this is especially true since rings are constructed in an ad-hoc fashion using keys of (possibly unknown) users which are not validated as being correctly constructed by any central authority! We formalize security against such attacks (as well as others), and show separation results proving that our definitions are strictly stronger than those considered in previous work. In addition to the strong definitions we present, the *hierarchy* of formal definitions we give is useful for precisely characterizing the security of prior and subsequent constructions.

Constructions. We show two constructions of ring signature schemes which are proven secure in the standard model. We stress that these are the *first* such constructions, as all previous constructions of which we are aware rely on the random oracle/ideal cipher models.[1] It is worth remarking that ring identification schemes are somewhat easier to construct (using, e.g., techniques from [6]); ring signatures can then easily be derived from such schemes using the Fiat-Shamir methodology in the random oracle model [11]. This approach, however, is no longer viable (at least, based on our current understanding) when working in the standard model.

Our first construction is based on generic assumptions, and satisfies the strongest definitions of anonymity and unforgeability considered here. This construction is inspired by the generic construction of group signatures due to Bellare, et al. [3] and, indeed, the constructions share some similarities at a high level. We stress, however, that a number of subtleties arise in our context that do not arise in the context of group signatures, and the construction given in [3] does not immediately lend itself to a ring signature scheme. Two issues in particular that we need to deal with are the fact that we have no central group manager to issue "certificates" as in [3], and that we additionally need to take into account the possibility of adversarially-generated public keys as discussed earlier (this is not a concern in [3] where there is only a single group key published by a (semi-)trusted group manager). We remark that although our functional definition of a ring signature scheme (cf. Def. 1) requires users to generate keys specifically for that purpose (in contrast to the requirements of [1, 2]), our first construction can be easily modified to work with any ring of users as long as they each have a public key for both encryption and signing (see Section 5 for further discussion).

Our second construction is more efficient than the first, but relies on specific number-theoretic assumptions. Furthermore, it provides more limited functionality and security guarantees than our first construction; most limiting is that it only supports rings of size two. We stress, however, that such a scheme is still useful for the application to designated-verifier signatures as discussed earlier; furthermore, even constructing an efficient 2-user ring signature scheme without random oracles seems difficult, as we still do not have the Fiat-Shamir methodology available in our toolbox. In fact, we explored various known signature schemes to see whether they could be adapted to give ring signatures, and found only one that was amenable: the signature scheme recently proposed by Waters [17] in the context of ID-based encryption. Interestingly, the ring signature we construct based on the Waters scheme does not increase the signature length (a concern addressed in [8]).

2 Preliminaries

We use the standard definitions of public-key encryption schemes and semantic security; signature schemes and existential unforgeability under adaptive chosen-message attacks; and computational indistinguishability. In this paper we will assume public-key encryption schemes for which, with all but negligible probability over (pk, sk) generated at random using the specified key generation algorithm, $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(M)) = M$ holds with probability 1.

We will also use the notion of a *ZAP*, which is a 2-round, public-coin, witness-indistinguishable proof system for any language in $\mathcal{NP}$ (the formal definition is given in Appendix A). ZAPs were introduced by Dwork and Naor [9], who show that ZAPs can be constructed based on any non-interactive zero-knowledge proof system; the latter, in turn, can be constructed based on trapdoor permutations [10]. For notational purposes, we represent a ZAP by a triple $(\ell, \mathcal{P}, \mathcal{V})$ such that

[1] Although Xu, Zhang, and Feng [18] claim a ring signature scheme in the standard model based on specific assumptions, their proof was later found to be flawed (personal communication from J. Xu, March 2005).

(1) the initial message from the verifier is chosen to be of length $\ell(k)$ (where k is the security parameter); (2) the prover $\mathcal{P}$, on input the prover-message r, statement x, and witness w, outputs $\pi \leftarrow \mathcal{P}_r(x, w)$; finally, (3) $\mathcal{V}_r(x, \pi)$ outputs 1 or 0, indicating acceptance or rejection of the proof.

3 Definitions

We begin by presenting the functional definition of a ring signature scheme [16]. We will refer to an ordered list $R = (PK_1, \ldots, PK_n)$ of public keys as a *ring*, and let $R[i] = PK_i$. We will also freely use set notation, and say, e.g., that $PK \in R$ if there exists an index i such that $R[i] = PK$. We will always assume, without loss of generality, that the keys in a ring are ordered lexicographically.

Definition 1 [**Ring signature**] A *ring signature scheme* is a triple of PPT algorithms (Gen, Sign, Vrfy) that, respectively, generate a key pair for a user, sign a message, and verify the signature of a message. Formally:

- $\mathsf{Gen}(1^k)$, where k is a security parameter, outputs a public key PK and secret key SK.
- $\mathsf{Sign}_{s,SK_s}(M, R)$ outputs a signature σ on the message M with respect to the ring $R = \{PK_1, \ldots, PK_n\}$. We assume the following conventions for simplicity: (1) $n \geq 2$ (since a ring signature scheme is not intended[2] to serve as a standard signature scheme); (2) each public key in the ring is distinct; and (3) (SK_s, PK_s) is a valid key-pair output by Gen, where $PK_2 = R[s]$.
- $\mathsf{Vrfy}_R(M, \sigma)$ verifies a purported signature σ on a message M with respect to the ring of public keys R.

We require the following completeness condition: for any k, any $\{(SK_i, PK_i)\}_{i=1}^n$ output by $\mathsf{Gen}(1^k)$, any $s \in [n]$, and any M, we have $\mathsf{Vrfy}_R(M, \mathsf{Sign}_{s,SK_s}(M, R)) = 1$, where $R = \{PK_1, \ldots, PK_n\}$.

A *c-user ring signature scheme* is a variant of the above that only supports rings of fixed size c (i.e., the Sign and Vrfy algorithms only take as input rings R for which $|R| = c$, and completeness is only required to hold for such rings).

A ring signature scheme is used as follows: At various times, some collection of users run the key generation algorithm Gen to generate public and private keys. We stress that no coordination among these users is assumed or required. When a user wishes to generate an anonymous signature on a message M, he chooses a ring R of public keys which includes his own; let s denote the index of this user's key within R. This user then computes $\sigma \leftarrow \mathsf{Sign}_{s,SK_s}(M, R)$ and outputs (σ, R). (In such a case, we will refer to the holder of PK_s as the *signer* of the message and to the holders of the other public keys in R as the *non-signers*.) Anyone can now verify that this signature was generated by *someone* holding a key in R by running $\mathsf{Vrfy}_R(M, \sigma)$.

As discussed in the Introduction, ring signatures must satisfy two independent notions of security: anonymity and unforgeability. There are various ways each of these notions can be defined (and various ways these notions have been defined in the literature); we present these different definitions in Sections 3.1 and 3.2, and compare them in Section 4.

[2]Furthermore, it is easy to modify any ring signature scheme to allow signatures with $n = 1$ by including a special key for just that purpose.

3.1 Definitions of Anonymity

The anonymity condition requires, informally, that an adversary not be able to tell which member of a ring generated a particular signature.[3] A number of subtleties, however, arise in developing a formal definition. We begin with a basic definition of anonymity which is (essentially) either explicit or (seems to be) implicit in most previous work (e.g., [16, 1, 8]).

Definition 2 [**Basic anonymity**] Given a ring signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$, a polynomial $n(\cdot)$, and a PPT adversary $\mathcal{A}$, consider the following game:

1. Key pairs $\{(PK_i, SK_i)\}_{i=1}^{n(k)}$ are generated using $\mathsf{Gen}(1^k)$, and the set of public keys $S \stackrel{\text{def}}{=} \{PK_i\}_{i=1}^{n(k)}$ is given to $\mathcal{A}$.
2. $\mathcal{A}$ outputs a message M, distinct indices i_0, i_1, and a ring $R \subseteq S$ for which $PK_{i_0}, PK_{i_1} \in R$. A random bit b is chosen, and $\mathcal{A}$ is given $\sigma \leftarrow \mathsf{Sign}_{i_b, SK_{i_b}}(M, R)$.[4]
3. The adversary outputs a bit b', and succeeds if $b' = b$.

We say $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ achieves *basic anonymity* if the success probability of any $\mathcal{A}$ in the above game is negligibly close to 1/2.

(Some previous papers consider a variant of the above in which the adversary is given a signature computed by a randomly-chosen member of S, and should be unable to guess the actual signer with probability better than $1/|S| + \mathsf{negl}(k)$. A simple simulation argument shows that such a variant is equivalent to the above.)

Unfortunately, the above definition of basic anonymity does *not* seem to suffice for many applications since it leaves open the possibility of the following attack: (1) an adversary generates public keys in some arbitrary manner (which may possibly depend on the public keys of the honest users), and then (2) a legitimate signer generates a signature with respect to a ring that contains some of these adversarially-generated public keys. Note that the definition above offers no protection in this case! Furthermore, the attack is quite realistic since, by their very nature, ring signatures are intended to be used in settings where there is not necessarily any central authority checking validity of public keys. This motivates the following, stronger definition (used in [15] in a slightly different context):

Definition 3 [**Anonymity w.r.t. adversarially-chosen keys**] Given a ring signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$, a polynomial $n(\cdot)$, and a PPT adversary $\mathcal{A}$, consider the following game:

1. As in Definition 2.
2. $\mathcal{A}$ outputs a message M, distinct indices i_0, i_1, and a ring R for which $PK_{i_0}, PK_{i_1} \in R$ and all keys in R are distinct (we stress that it is not required that $R \subseteq S$). A random bit b is chosen, and $\mathcal{A}$ is given $\sigma \leftarrow \mathsf{Sign}_{i_b, SK_{i_b}}(M, R)$.
3. The adversary outputs a bit b', and succeeds if $b' = b$.

[3] All the anonymity definitions that follow can be phrased in either a *computational* or an *unconditional* sense (where, informally, in the former case anonymity holds for polynomial-time adversaries while in the latter case anonymity holds even for all-powerful adversaries). For simplicity, we only present the computational versions.

[4] Technically speaking, we should say that $\mathcal{A}$ is given $\sigma \leftarrow \mathsf{Sign}_{j_b, SK_{i_b}}(M, R)$, where j_b is such that $R[j_b] = PK_{i_b}$. In an attempt to improve readability here and in the rest of the paper, however, we will avoid such precise but hard-to-parse notation and will instead use indices liberally whenever doing so will not cause confusion.

We say $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ achieves *anonymity w.r.t. adversarially-chosen keys* if the success probability of any $\mathcal{A}$ in the above game is negligibly close to 1/2.

Definition 3 also may not be sufficient in certain scenarios. In particular, it may also be desirable for anonymity to be preserved even in case the secret keys of (some) members of the ring are exposed or, more generally, even if the randomness used to generate the secret keys of (some) members of the ring are exposed (which provides security in case erasure cannot be guaranteed). Beyond the obvious advantages of such a guarantee in case of long-term key exposure, etc., the main advantage is that it ensures anonymity in case the non-signers decide to reveal their random coins in an attempt to attribute the signature to the actual signer by proving that they did not sign the message. Additional motivation is given below. (For simplicity, we also protect against adversarially-chosen keys, although one could consider the weaker definition which does not.)

Definition 4 [**Anonymity against attribution attacks/against key exposure**] Given (Gen, Sign, Vrfy), $n(\cdot)$, and $\mathcal{A}$ as in Definition 3, consider the following game:

1. For $i = 1$ to $n(k)$, generate $(PK_i, SK_i) \leftarrow \mathsf{Gen}(1^k; \omega_i)$ for randomly-chosen ω_i. Give to $\mathcal{A}$ the set of public keys $\{PK_i\}_{i=1}^{n(k)}$.
2. $\mathcal{A}$ outputs a message M, distinct indices i_0, i_1, and a ring R for which $PK_{i_0}, PK_{i_1} \in R$ and all keys in R are distinct. $\mathcal{A}$ is given $\{\omega_i\}_{i \neq i_0}$. Furthermore, a random bit b is chosen and $\mathcal{A}$ is given $\sigma \leftarrow \mathsf{Sign}_{i_b, SK_{i_b}}(M, R)$.
3. The adversary outputs a bit b', and succeeds if $b' = b$.

We say $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ achieves *anonymity against attribution attacks* if the success probability of any $\mathcal{A}$ in the above game is negligibly close to 1/2. If, in the second step, $\mathcal{A}$ is instead given $\{\omega_i\}_{i=1}^{n(k)}$ then we say $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ achieves *anonymity against key exposure*.

The definition of anonymity against key exposure parallels (in fact, is stronger than) the anonymity definition given by Bellare, et al. in the context of group signatures [3]. We also believe that anonymity against attribution attacks is a useful relaxation since a signer can then be assured of anonymity as long as he *himself* refuses to divulge his secret key; although this might lead to suspicion, it still cannot be proved (in court, say) that this user indeed issued the signature. We also note that considering the case in which the *randomness* used to generate keys is revealed (rather than "only" the secret keys themselves) makes sense when erasure cannot be ensured, or when it cannot be guaranteed that all users will comply with the directive to erase their random coins.

Linkability. Another desideratum of a ring signature scheme is that it be *unlinkable*; that is, that it be infeasible to determine whether two signatures (possibly generated with respect to different rings) were generated by the same person. We remark that, similar to the case in [3], computational anonymity against attribution attacks is sufficient to ensure unlinkability; this serves as additional motivation for considering that definition. Also, any scheme which is *unconditionally* anonymous (with respect to either Definition 2 or 3), and in which there is a unique secret key corresponding to any public key, is also unlinkable (with respect to an appropriate extension of the corresponding definition). In particular, then, both constructions given in this paper are unlinkable (in the appropriate sense).

3.2 Definitions of Unforgeability

The notion of unforgeability considered in many previous works [16, 8, 2] is the following:

Definition 5 [**Unforgeability against fixed-ring attacks**] A ring signature scheme (Gen, Sign, Vrfy) is *unforgeable against fixed-ring attacks* if for any PPT adversary $\mathcal{A}$ and for any polynomial $n(\cdot)$, the probability that $\mathcal{A}$ succeeds in the following game is negligible:

1. Key pairs $\{(PK_i, SK_i)\}_{i=1}^{n(k)}$ are generated using $\mathsf{Gen}(1^k)$, and the set of public keys $R \stackrel{\text{def}}{=} \{PK_i\}_{i=1}^{n(k)}$ is given to $\mathcal{A}$.
2. $\mathcal{A}$ is given access to a *signing oracle* $\mathsf{OSign}(\cdot,\cdot)$, where $\mathsf{OSign}(s, M)$ outputs $\mathsf{Sign}_{s,SK_s}(M, R)$.
3. $\mathcal{A}$ outputs (M^*, σ^*), and succeeds if $\mathsf{Vrfy}_R(M^*, \sigma^*) = 1$ and it never queried $(\star, M^*)$ to its signing oracle.

Note that not only is the adversary restricted to making signing queries with respect to the entire ring, but its forgery is required to verify with respect to the entire ring. The following stronger, and more natural, definition was used in [1]:

Definition 6 [**Unforgeability against chosen-subring attacks**] A ring signature scheme (Gen, Sign, Vrfy) is *unforgeable against chosen-subring attacks* if for any PPT adversary $\mathcal{A}$ and for any polynomial $n(\cdot)$, the probability that $\mathcal{A}$ succeeds in the following game is negligible:

1. Key pairs $\{(PK_i, SK_i)\}_{i=1}^{n(k)}$ are generated using $\mathsf{Gen}(1^k)$, and the set of public keys $S \stackrel{\text{def}}{=} \{PK_i\}_{i=1}^{n(k)}$ is given to $\mathcal{A}$.
2. $\mathcal{A}$ is given access to a *signing oracle* $\mathsf{OSign}(\cdot,\cdot,\cdot)$, where $\mathsf{OSign}(s, M, R)$ outputs $\mathsf{Sign}_{s,SK_s}(M, R)$ and we require that $R \subseteq S$ and $PK_s \in R$ (and also $|R| \geq 2$, as detailed in Definition 1).
3. $\mathcal{A}$ outputs (M^*, σ^*, R^*), and succeeds if $R^* \subseteq S$, $\mathsf{Vrfy}_{R^*}(M^*, \sigma^*) = 1$, and $\mathcal{A}$ never queried $(\star, M^*, R)$ to its signing oracle.

While the above definition is an improvement, it still leaves open the possibility of an attack whereby users might generate signatures with respect to rings containing adversarially-generated public keys. Such an attack was also previously suggested by [15, 14]. The following definition takes this into account as well as (for completeness) an adversary who adaptively corrupts[5] honest participants and obtains their secret keys, as was considered by Bellare, et al. for the case of group signatures [3]. Since either of these attacks may be viewed as the outcome of corrupting an "insider", we use this terminology.[6]

Definition 7 [**Unforgeability w.r.t. insider corruption**] A ring signature scheme (Gen, Sign, Vrfy) is *unforgeable w.r.t. insider corruption* if for any PPT adversary $\mathcal{A}$ and for any polynomial $n(\cdot)$, the probability that $\mathcal{A}$ succeeds in the following game is negligible:

1. Key pairs $\{(PK_i, SK_i)\}_{i=1}^{n(k)}$ are generated using $\mathsf{Gen}(1^k)$, and the set of public keys $S \stackrel{\text{def}}{=} \{PK_i\}_{i=1}^{n(k)}$ is given to $\mathcal{A}$.
2. $\mathcal{A}$ is given access to a *signing oracle* $\mathsf{OSign}(\cdot,\cdot,\cdot)$, where $\mathsf{OSign}(s, M, R)$ outputs $\mathsf{Sign}_{s,SK_s}(M, R)$ and we require that $PK_s \in R$.
3. $\mathcal{A}$ is also given access to a *corrupt oracle* $\mathsf{Corrupt}(\cdot)$, where $\mathsf{Corrupt}(i)$ outputs SK_i.

[5] We remark that Definitions 3 and 4 already guarantee anonymity in case of the corruption of honest users, and so we do not explicitly consider corruptions when defining anonymity.

[6] Although we are aware that, technically speaking, there are not really any "insiders" in the context of ring signatures since there is no coordination among the various users.

4. $\mathcal{A}$ outputs (M^*, σ^*, R^*), and succeeds if $\mathsf{Vrfy}_{R^*}(M^*, \sigma^*) = 1$, $\mathcal{A}$ never queried $(\star, M^*, R^*)$, and $R^* \subseteq S \setminus C$, where C is the set of corrupted users.

4 Separations Between the Security Definitions

In the previous section, we presented various definitions of anonymity and unforgeability in order of increasing strength. Here, we show that these definitions are in fact distinct, in the sense that there exist (under certain assumptions) schemes satisfying a weaker definition but not a stronger one. First, we show separations for the definitions of anonymity. (So as not to interrupt the main flow of the paper, we defer all proofs to Appendix B.)

Claim 1 *If there exists a scheme which achieves* **basic** *anonymity and is unforgeable w.r.t. insider corruption, then there exists a scheme which achieves these same properties but which is* **not** *anonymous w.r.t.* **adversarially-chosen keys**.

Claim 2 *If there exists a scheme which is anonymous w.r.t.* **adversarially-chosen keys** *and is unforgeable w.r.t. insider corruption, then there exists a scheme which achieves these same properties but which is* **not** *anonymous against* **attribution attacks**.

We also show separations for the definitions of unforgeability:

Claim 3 *If there exists a scheme which is anonymous against key exposure and unforgeable w.r.t. insider corruption, then there exists a scheme which is anonymous against key exposure and unforgeable against* **fixed-ring attacks**, *but* **not** *unforgeable against* **chosen-subring attacks**.

(In contrast to the rest of the claims, the assumption in the above claim is not minimal. Nevertheless, the claim is meaningful especially since we show in Section 5 that the assumption of the claim is satisfied under certain cryptographic assumptions.) We remark also that the scheme of [12] serves as a *natural* example of a scheme that is unforgeable against fixed-ring attacks, but which is **not** unforgeable against chosen-subring attacks. See Appendix B.1.

Claim 4 *If there exists a scheme which is anonymous against key exposure and is unforgeable against* **chosen-subring attacks**, *then there exists a scheme which achieves these same properties but which is* **not** *unforgeable w.r.t.* **insider corruption**.

5 A Ring Signature Scheme Based on General Assumptions

We now describe our construction of a ring signature scheme that satisfies the strongest of our proposed definitions, and is based on general assumptions. In what follows, we let $(\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme, let $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ be a (standard) signature scheme, and let $(\ell, \mathcal{P}, \mathcal{V})$ be a ZAP (for the witness relation $\mathcal{R}$ defined below). We denote by $C^* \leftarrow \mathsf{Enc}^*_{R_E}(m)$ the probabilistic algorithm that takes as input a set of public encryption keys $R_E = \{pk_{E,1}, \ldots, pk_{E,n}\}$ and a message m, and does the following: it first chooses random $s_1, \ldots, s_{n-1} \in \{0,1\}^{|m|}$ and then outputs:

$$C^* = \mathsf{Enc}_{pk_{E,1}}(s_1), \mathsf{Enc}_{pk_{E,2}}(s_2), \cdots, \mathsf{Enc}_{pk_{E,n-1}}(s_{n-1}), \mathsf{Enc}_{pk_{E,n}}(m \oplus \bigoplus_{j=1}^{n-1} s_j)\,.$$

Note that, informally, m can be recovered from the above ciphertext only if *all* the corresponding secret keys are known. We also let L denote the $\mathcal{NP}$ language:

$$\left\{(pk_S, M^*, R_E, C^*) : \exists \sigma', \omega \text{ s.t. } C^* = \mathsf{Enc}^*_{R_E}(\sigma'; \omega) \bigwedge \mathsf{Vrfy}'_{pk_S}(M^*, \sigma') = 1\right\},$$

with $\mathcal{R}_L$ the obvious associated witness relation. The relation for the ZAP is $\mathcal{R} = \{((x_1, \ldots, x_n), w) : \exists i \text{ s.t. } (x_i, w) \in \mathcal{R}_L\}$. With this in mind, we give the details of our construction, which is specified by the key-generation algorithm Gen, the ring signing algorithm Sign, and the ring verification algorithm Vrfy:

$\underline{\mathsf{Gen}(1^k)}$:

1. Generate signing key pair $(pk_S, sk_S) \leftarrow \mathsf{Gen}'(1^k)$.
2. Generate encryption key pair $(pk_E, sk_E) \leftarrow \mathsf{Gen}(1^k)$ and erase sk_E.
3. Choose an initial ZAP message $r \leftarrow \{0,1\}^{\ell(k)}$.
4. Output the public key $PK = (pk_S, pk_E, r)$, and the secret key $SK = sk_S$.

$\underline{\mathsf{Sign}_{i^*, SK_{i^*}}(M, (PK_1, \ldots, PK_n))}$:

(We assume SK_{i^*} corresponds to PK_{i^*}, and that the $\{PK_i\}$ are in lexicographic order.)

1. Parse each PK_i as $(pk_{S,i}, pk_{E,i}, r_i)$, and parse SK_{i^*} as sk_{S,i^*}. Set $R_E := \{pk_{E,1}, \ldots, pk_{E,n}\}$.
2. Set $M^* := M \,|\, PK_1 \,|\, \cdots \,|\, PK_n$, where "|" denotes concatenation. Compute the signature $\sigma'_{i^*} \leftarrow \mathsf{Sign}'_{sk_{S,i^*}}(M^*)$.
3. Choose random coins $\omega_1, \ldots, \omega_n$ for Enc^* and: (1) compute $C^*_{i^*} = \mathsf{Enc}^*_{R_E}(\sigma'_{i^*}; \omega_{i^*})$ and (2) for $i \in \{1, \ldots, n\} \setminus \{i^*\}$, compute $C^*_i = \mathsf{Enc}^*_{R_E}(0^{|\sigma'_{i^*}|}; \omega_i)$.
4. For $i \in [n]$, let x_i denote the statement: "$(pk_{S,i}, M^*, R_E, C^*_i) \in L$", and let $x = \bigvee x_i$. Compute the proof $\pi \leftarrow \mathcal{P}_{r_1}(x, (\sigma'_{i^*}, \omega_{i^*}))$.
5. The signature is $\sigma = (C^*_1, \ldots, C^*_n, \pi)$.

$\underline{\mathsf{Vrfy}_{PK_1,\ldots,PK_n}(M,\sigma)}$

(We assume the $\{PK_i\}$ are in lexicographic order.)

1. Parse each PK_i as $(pk_{S,i}, pk_{E,i}, r_i)$. Set $M^* \stackrel{\text{def}}{=} M \,|\, PK_1 \,|\, \cdots \,|\, PK_n$ and define $R_E \stackrel{\text{def}}{=} \{pk_{E,1}, \ldots, pk_{E,n}\}$. Parse σ as $(C_1^*, \ldots, C_n^*, \pi)$.
2. For $i \in [n]$, let x_i denote the statement "$(pk_{S,i}, M^*, R_E, C_i^*) \in L$" and set $x = \bigvee x_i$.
3. Output $\mathcal{V}_{r_1}(x, \pi)$.

It is easy to see that the scheme above satisfies the functional definition of a ring signature scheme. We now prove that the scheme satisfies strong notions of anonymity and unforgeability:

Theorem 1 *If encryption scheme* $(\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$ *is semantically secure, signature scheme* $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ *is existentially unforgeable under adaptive chosen-message attacks, and* $(\ell, \mathcal{P}, \mathcal{V})$ *is a ZAP for* $\mathcal{R}$ *as described above, then the above ring signature scheme is (computationally) anonymous against attribution attacks, and unforgeable w.r.t. insider corruption.*

Proof (Sketch) We prove each of the desired security properties in turn.

Anonymity. For simplicity of exposition, we consider Definition 4 with $n = 2$; i.e., we assume there are only two honest users. By a straightforward hybrid argument, this implies the general case. Given any PPT adversary $\mathcal{A}$, we consider a sequence of experiments E_0, Hybrid_0, Hybrid_1, E_1 such that E_0 (resp., E_1) corresponds to the experiment of Definition 4 with $b = 0$ (resp., $b = 1$), and such that each experiment is computationally indistinguishable from the one before it. This implies that $\mathcal{A}$ has negligible advantage in distinguishing E_0 from E_1, as desired.

For convenience, we review experiment E_0. Here, two key pairs $(PK_0 = (pk_{S,0}, pk_{E,0}, r_0), SK_0)$ and $(PK_1 = (pk_{S,1}, pk_{E,1}, r_1), SK_1)$ are generated and $\mathcal{A}$ is given PK_0 and the randomness used to generate (PK_1, SK_1) (recall we use $n = 2$ and so we can assume without loss of generality that $i_0 = 0$ and $i_1 = 1$). The adversary $\mathcal{A}$ then outputs a message M along with a ring of public keys R containing both PK_0 and PK_1. Finally, $\mathcal{A}$ is given $\sigma \leftarrow \mathsf{Sign}_{0,SK_0}(M, R)$.

Experiment Hybrid_0 is the same as experiment E_0 except that we change how the signature σ is generated. In particular, step 3 of the ring signing algorithm is modified as follows: let R_E and M^* be as in the description of the ring signing algorithm given earlier. In step 3, instead of setting C_1^* to be an encryption of all zeros, we now compute $\sigma_1' \leftarrow \mathsf{Sign}_{sk_{S,1}}(M^*)$ and then set $C_1^* = \mathsf{Enc}^*_{R_E}(\sigma_1'; \omega_1)$. We stress that, as in E_0, the ciphertext C_0^* is still set to be an encryption of the signature σ_0', and the remaining ciphertexts are still encryptions of all zeros.

It is not hard to see that experiment Hybrid_0 is computationally indistinguishable from experiment E_0, assuming semantic security of the encryption scheme $(\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$. This follows from the observations that (1) adversary $\mathcal{A}$ is *not* given the random coins used in generating PK_0 and so, in particular, it is not given the coins used to generate $pk_{E,0}$; (2) (informally) semantic security of encryption under $\mathsf{Enc}_{pk_{E_0}}$ implies semantic security of encryption using $\mathsf{Enc}^*_{R_E}$ as long as $pk_{E,0} \in R_E$ (a formal proof is straightforward); and, finally, (3) the coins ω_1 used in generating C_1^* are not used in the remainder of the ring signing algorithm.

Experiment Hybrid_1 is the same as Hybrid_0 except that we use a different witness when computing the proof π for the ZAP. In particular, instead of using witness (σ_0', ω_0) we use the witness (σ_1', ω_1). The remainder of the signing algorithm is unchanged.

It is relatively immediate that experiment Hybrid_1 is computationally indistinguishable from Hybrid_0, assuming witness indistinguishability of the ZAP. (We remark that the use of a ZAP, rather than non-interactive zero-knowledge, is essential here since the adversary may choose the

"random string" component of all the adversarially-chosen public keys any way it likes.) In more detail, we can construct the following malicious verifier algorithm $\mathcal{V}^*$ using $\mathcal{A}$: verifier $\mathcal{V}^*$ generates (PK_0, SK_0) and (PK_1, SK_1) exactly as in experiments Hybrid_0 and Hybrid_1, and gives these keys and the appropriate associated random coins to $\mathcal{A}$. When $\mathcal{A}$ makes its signing query, $\mathcal{V}^*$ computes the C_i^* exactly as in Hybrid_1 and then gives to the prover $\mathcal{P}$ the keys $\{pk_{S,i}\}_{i \in R}$, the message M^*, the set of keys R_E, and the ciphertexts $\{C_i^*\}_{i \in R}$; this defines the $\mathcal{NP}$-statement x exactly as in step 4 of the ring signing algorithm. In addition, $\mathcal{V}^*$ gives the two witnesses (σ'_0, ω_0) and (σ'_1, ω_1) to $\mathcal{P}$. Finally, $\mathcal{V}^*$ sends as its first message the "random string" component r of the lexicographically-first public key in R (note that this r is the random string that would be used to generate the proof π in step 4 of the ring signing algorithm). The prover responds with a proof $\pi \leftarrow \mathcal{P}_r(x, (\sigma'_b, \omega_b))$ (for some $b \in \{0,1\}$), and then $\mathcal{V}^*$ outputs $(C_1^*, \ldots, C_n^*, \pi)$.

Note that if the prover uses the first witness provided to it by $\mathcal{V}^*$ then the output of $\mathcal{V}^*$ is distributed exactly according to Hybrid_0, while if the prover uses the second witness provided to it by $\mathcal{V}^*$ then the output of $\mathcal{V}^*$ is distributed exactly according to Hybrid_1. Witness indistinguishability of the ZAP thus implies computational indistinguishability of Hybrid_0 and Hybrid_1.

We may now notice that Hybrid_1 is computationally indistinguishable from E_1 by exactly the same argument used to show the indistinguishability of Hybrid_0 and E_0. This completes the proof.

Unforgeability. Assume there exists a PPT adversary $\mathcal{A}$ that breaks the above ring signature scheme (in the sense of Definition 7) with non-negligible probability. We construct an adversary $\mathcal{A}'$ that breaks the underlying signature scheme $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ (in the standard sense of existential unforgeability) with non-negligible probability.

$\mathcal{A}'$ receives as input a public key pk_S. Let $n = n(k)$ be a bound on the number of (honest user) public keys that $\mathcal{A}$ expects to be generated. $\mathcal{A}'$ runs $\mathcal{A}$ with input public keys $PK_1, \ldots, PK_n$, that $\mathcal{A}'$ generates as follows. $\mathcal{A}'$ chooses $i^* \leftarrow \{1, \ldots, n\}$ and sets $pk_{S,i^*} = pk_S$. The remainder of public key PK_{i^*} is generated exactly as prescribed by the Gen algorithm, with the exception that the decryption key sk_{E,i^*} that is generated is *not* erased. Public keys PK_i for $i \neq i^*$ are also generated exactly as prescribed by the Gen algorithm, again with the exception that the decryption keys $\{sk_{E,i}\}$ are not erased.

$\mathcal{A}'$ then proceeds to simulate the oracle queries of $\mathcal{A}$ in the natural way. In particular:

1. When $\mathcal{A}$ requests a signature on message M, with respect to ring R (which may possibly contain some public keys generated in an arbitrary manner by $\mathcal{A}$), to be signed by user $i \neq i^*$, then $\mathcal{A}'$ can easily generate the response to this query by running the Sign algorithm completely honestly;
2. When $\mathcal{A}$ requests a signature on message M, with respect to ring $R = \{PK_1, \ldots, PK_n\}$ (which may possibly contain some public keys generated in an arbitrary manner by $\mathcal{A}$) to be signed by user i^*, then $\mathcal{A}'$ cannot directly respond to this query since it does not have sk_{S,i^*}. Instead, $\mathcal{A}'$ sets $M^* := M \,|\, PK_1 \,|\, \cdots \, PK_n$, submits M^* to its signing oracle, and obtains in return a signature σ'_{i^*}. It then computes the remainder of the ring signature by following the rest of the Sign algorithm; note, in particular, that sk_{S,i^*} is not needed for this;
3. Any corruption query made by $\mathcal{A}$ for a user $i \neq i^*$ can be faithfully answered by $\mathcal{A}'$. On the other hand, if $\mathcal{A}$ ever makes a corruption query for i^*, then $\mathcal{A}'$ simply aborts.

At some point, $\mathcal{A}$ outputs a forgery $\bar{\sigma} = (\bar{C}_1^*, \ldots, \bar{C}_n^*, \bar{\pi})$ on a message $\bar{M}$ with respect to some ring of honest-user public keys $\bar{R} = \{\overline{PK}_1, \ldots, \overline{PK}_{n'}\}$. If PK_{i^*} is not contained in the ring $\bar{R}$, then $\mathcal{A}'$ aborts. Otherwise, since $\mathcal{A}'$ knows all relevant decryption keys (recall that the ring $\bar{R}$ contains public keys of honest users only, and these keys were generated by $\mathcal{A}'$) it can decrypt $\bar{C}_{i^*}^*$ and obtain

the candidate signature $\bar{\sigma}_{i^*}$. Finally, $\mathcal{A}'$ sets $\bar{M}^* = \bar{M} \,|\, \overline{PK}_1 \,|\, \cdots \,|\, \overline{PK}_{n'}$ and outputs $(\bar{M}^*, \bar{\sigma}_{i^*})$. Note that (by requirement) $\mathcal{A}$ never requested a signature on message $\bar{M}$ with respect to the ring $\bar{R}$, and so $\mathcal{A}'$ never requested a signature on message $\bar{M}^*$ from its own oracle.

We claim that if $\mathcal{A}$ forges a signature with non-negligible probability $\varepsilon = \varepsilon(k)$, then $\mathcal{A}'$ forges a signature with probability at least $\varepsilon' = \varepsilon/n - \mathsf{negl}$. To see this, note first that if $\mathcal{A}$ outputs a valid forgery then with all but negligible probability (by soundness of the ZAP) it holds that $(\overline{pk}_{S,i}, \bar{M}^*, \bar{R}_E, \bar{C}_i^*) \in L$ for some i (where $\overline{pk}_{S,i}$ and $\bar{R}_E$ are defined in the natural way based on the ring $\bar{R}$ and the public keys it contains). Conditioned on this, with probability $1/n$ it is the case that (1) $\mathcal{A}'$ did not abort and furthermore (2) $(\overline{pk}_{S,i^*}, \bar{M}^*, \bar{R}_E, \bar{C}_{i^*}^*) \in L$. When this occurs, then with all but negligible probability $\mathcal{A}'$ will recover (by decrypting as described above) a valid signature $\bar{\sigma}_{i^*}$ on the message $\bar{M}^*$ with respect to the given public key $\overline{pk}_{S,i^*} = pk_S$ (we remark that we rely here on the fact that with all but negligible probability over choice of public encryption keys the encryption scheme Enc^* has zero decryption error). Security of $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ thus implies that ε must be negligible. ■

Extension. The scheme above can also be used (with a few easy modifications) in a situation where some users in the ring have not generated a key pair according to Gen, as long as every ring member has a public key both for encryption and for signing, and at least one of the members has included a sufficiently-long random string in his public key. Furthermore, the encryption (signature) public keys of different members of the ring may be associated with different encryption (signature) schemes. Thus, a *single* user who establishes a public key for a ring signature scheme suffices to provide anonymity for everyone. Alternately, this provides a way to include "oblivious" users in the signing ring, as in [1, 2].

5.1 Achieving Stronger Anonymity Guarantees

In our proof that the previous scheme satisfies the anonymity requirement, it was essential that the adversary not be given the random coins used to generate *both* ring signature keys.[7] (If the adversary gets both sets of random coins, it can potentially decrypt ciphertexts encrypted using $\mathsf{Enc}^*_{R_E}$ and thereby determine the true signer of a message.) In this section, we note how it is possible to achieve the stronger notion of anonymity against key exposure, whereby the adversary is given *all* the random coins used by the honest players in generating their keys.

To achieve this, we will use an enhanced form of encryption for which, informally, there exists an "oblivious" way to generate a public key without generating a corresponding secret key. This notion, introduced by Damgård and Nielsen [7], can be viewed as a generalization of *dense cryptosystems* in which the public key is required to be a uniformly distributed string (in particular, dense cryptosystems satisfy the definition below). We review the formal definition here.

Definition 8 An *oblivious public-key generator* for public-key encryption scheme $(\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$ is a pair of PPT algorithms $(\mathsf{OblEGen}, \mathsf{OblRand})$ such that:

- $\mathsf{OblEGen}$, on input 1^k and random coins $\omega \in \{0,1\}^{n(k)}$, outputs a key pk;
- $\mathsf{OblRand}$, on input a key pk, outputs a string ω;

and the following distribution ensembles are computationally indistinguishable:

$$\left\{\omega \leftarrow \{0,1\}^{n(k)} : (\omega, \mathsf{OblEGen}(1^k; \omega))\right\}$$

[7] However, anonymity still holds even if the adversary is given both secret keys (but not the randomness used to generate both secret keys). This is because the decryption key sk_E is erased, and not included in SK.

and

$$\left\{(pk, sk) \leftarrow \mathsf{EGen}(1^k); \omega \leftarrow \mathsf{OblRand}(pk) : (\omega, pk)\right\}.$$

Note that if $(\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$ is semantically secure, then (informally) it is also semantically secure to encrypt messages using a public key pk generated by $\mathsf{OblEGen}$, even if the adversary has the random coins used by $\mathsf{OblEGen}$ in generating pk. We remark for completeness that the El Gamal encryption scheme (over the group of quadratic residues modulo a prime) is an example of a scheme having an oblivious public-key generator.

Given the above, we adapt the scheme of the previous section in the natural way. Specifically, the Gen algorithm is changed so that instead of generating the key pk_E using EGen (and then erasing the secret key sk_E and the random coins used), we now generate pk_E using $\mathsf{OblEGen}$. Adapting Theorem 1, we can easily show:

Theorem 2 *Under the assumptions of Theorem 1 and assuming* $(\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$ *has an oblivious public-key generator, the modified ring signature scheme described above is (computationally) anonymous against key exposure, and unforgeable w.r.t. insider corruption.*

The proof is given in Appendix C

6 An Efficient 2-User Ring Signature Scheme

In this section, we present a more efficient construction of a 2-user ring signature scheme based on specific assumptions. The scheme is based on the (standard) signature scheme[8] constructed by Waters [17] which we briefly review now.

6.1 The Waters Scheme

Let $\mathbb{G}, \mathbb{G}_1$ be groups of prime order q such that there exists an efficiently computable bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$. We assume that $q, \mathbb{G}, \mathbb{G}_1, \hat{e}$, and a generator $g \in \mathbb{G}$ are publicly known. The Waters signature scheme for messages of length n is defined as follows:

Key Generation. Choose $\alpha \leftarrow \mathbb{Z}_q$ and set $g_1 = g^\alpha$. Additionally choose random elements $h, u', u_1, \ldots, u_n \leftarrow \mathbb{G}$. The public key is $(g_1, h, u', u_1, \ldots, u_n)$ and the secret key is h^α.

Signing. To sign the n-bit message M, first compute $w = u' \cdot \prod_{i:M_i=1} u_i$. Then choose random $r \leftarrow \mathbb{Z}_q$ and output the signature $\sigma = (h^\alpha \cdot w^r,\ g^r)$.

Verification. To verify the signature (A, B) on message M with respect to public key (g_1, h, u', u_1, …, u_n), compute $w = u' \cdot \prod_{i:M_i=1} u_i$ and then check whether $\hat{e}(g_1, h) \cdot \hat{e}(B, w) \stackrel{?}{=} \hat{e}(A, g)$.

6.2 A 2-User Ring Signature Scheme

The main observation we make with regard to the above scheme is the following: element h is arbitrary, and only knowledge of h^α is needed to sign. So, we can dispense with including h in the public key altogether; instead, a user U with secret α and the value $g_1 = g^\alpha$ in his public key will use as his "h-value" the value $\bar{g}_1$ contained in the public key of a *second* user $\bar{U}$. This

[8] More accurately, the signature scheme shown here is derived from the identity-based encryption (IBE) scheme of Waters using the idea first suggested by Naor for converting IBE schemes to signature schemes.

provides anonymity since $\bar{U}$ could *also* have computed the very same value $(\bar{g}_1)^\alpha$ using the secret value $\bar{\alpha} = \log_g \bar{g}_1$ known to him (because $(\bar{g}_1)^\alpha = g_1^{\bar{\alpha}}$). We now proceed with the details.

Key Generation. Choose $\alpha \leftarrow \mathbb{Z}_q$ and set $g_1 = g^\alpha$. Additionally choose random elements $u', u_1, \ldots, u_n \leftarrow \mathbb{G}$. The public key is $(g_1, u', u_1, \ldots, u_n)$ and the secret key is α. (We again assume that $q, \mathbb{G}, \mathbb{G}_1, \hat{e}$, and g are system-wide parameters.)

Ring Signing. To sign message $M \in \{0,1\}^n$ with respect to the ring $R = \{PK, \overline{PK}\}$ using secret key α (where we assume without loss of generality that α is the secret corresponding to PK), proceed as follows: parse PK as $(g_1, u', u_1, \ldots, u_n)$ and $\overline{PK}$ as $(\bar{g}_1, \bar{u}', \bar{u}_1, \ldots, \bar{u}_n)$, and compute $w = u' \cdot \prod_{i:M_i=1} u_i$ and $\bar{w} = \bar{u}' \cdot \prod_{i:M_i=1} \bar{u}_i$. Then choose random $r \leftarrow \mathbb{Z}_q$ and output the signature

$$\sigma = (\bar{g}_1^\alpha \cdot (w\bar{w})^r,\ g^r)\,.$$

Ring Verification. To verify the signature (A, B) on message M with respect to the ring $R = \{PK, \overline{PK}\}$ (parsed as above), compute $w = u' \cdot \prod_{i:M_i=1} u_i$ and $\bar{w} = \bar{u}' \cdot \prod_{i:M_i=1} \bar{u}_i$ and then check whether $\hat{e}(g_1, \bar{g}_1) \cdot \hat{e}(B, (w\bar{w})) \stackrel{?}{=} \hat{e}(A, g)$.

It is not hard to see that correctness holds. We prove the following regarding the above scheme:

Theorem 3 *Assume the Waters signature scheme is existentially unforgeable under adaptive chosen message attack. Then the 2-user ring signature scheme described above is unconditionally anonymous against key exposure attacks, and unforgeable against chosen-subring attacks.*

Proof Unconditional anonymity against key exposure attacks follows easily from the observations made earlier: namely, that only the value $\bar{g}_1^\alpha = g_1^{\bar{\alpha}}$ (where $\bar{\alpha} \stackrel{\text{def}}{=} \log_g \bar{g}_1$) is needed to sign, and either of the two (honest) parties can compute this value.

We now prove that the scheme satisfies Definition 6. We do this by showing how an adversary $\mathcal{A}$ that forges a signature with respect to the ring signature scheme with non-negligible probability, can be used to construct an adversary $\hat{\mathcal{A}}$ that forges a signature with respect to the Waters signature scheme (in the standard sense) with the same probability. For simplicity in the proof, we assume that $\mathcal{A}$ only ever sees the public keys of two users, requests all signatures to be signed with respect to the ring R containing these two users, and forges a signature with respect to that same ring R. By a hybrid argument, it can be shown easily that this is equivalent to the more general case when $\mathcal{A}$ may see multiple public keys, request signatures with respect to various (different) 2-user subsets, and then output a forgery with respect to any 2-user subset of its choice.[9]

Construct $\hat{\mathcal{A}}$ as follows: $\hat{\mathcal{A}}$ is given the public key $(\hat{g}_1, \hat{h}, \hat{u}', \hat{u}_1, \ldots, \hat{u}_n)$ of an instance of the Waters scheme. $\hat{\mathcal{A}}$ constructs two user public keys as follows: first, it sets $g_1 = \hat{g}_1$ and $\bar{g}_1 = \hat{h}$. Then, it chooses random $u', u_1, \ldots, u_n \leftarrow \mathbb{G}$ and sets $\bar{u}' = \hat{u}'/u'$ and $\bar{u}_i = \hat{u}_i/u_i$ for all i. It gives to $\mathcal{A}$ the public keys $(g_1, u', u_1, \ldots, u_n)$ and $(\bar{g}_1, \bar{u}', \bar{u}_1, \ldots, \bar{u}_n)$. Note that both public keys have the appropriate distribution. When $\mathcal{A}$ requests a ring signature on a message M with respect to the ring R containing these two public keys, $\hat{\mathcal{A}}$ requests a signature on M from its signing oracle, obtains in return a signature (A, B), and gives this signature to $\mathcal{A}$. Note that this is indeed a perfect simulation, since

$$\left(\hat{h}^{\log_g \hat{g}_1} \cdot \left(\hat{u}' \prod_{i:M_i=1} \hat{u}_i\right)^r,\ g^r\right) = \left(\bar{g}_1^{\log_g g_1} \cdot \left(u'\bar{u}' \prod_{i:M_i=1} u_i\bar{u}_i\right)^r,\ g^r\right),$$

[9] Equivalence holds for the case of any c-user ring signature scheme for any constant c (as is the case here), since the number of possible subrings in such a case in polynomial. In the general case when rings can be any polynomial size, the number of possible rings is exponential and hence the hybrid argument no longer applies.

which is an appropriately-distributed ring signature with respect to the public keys given to $\mathcal{A}$.

When $\mathcal{A}$ outputs a forgery (A^*, B^*) on a message M^*, this same forgery is output by $\hat{\mathcal{A}}$. Note that $\hat{\mathcal{A}}$ outputs a valid forgery whenever $\mathcal{A}$ does, since

$$\hat{e}(g_1, \bar{g}_1) \cdot \hat{e}(B^*, (u'\bar{u}' \textstyle\prod_{i:M_i^*=1} u_i \bar{u}_i)) = \hat{e}(A^*, g) \implies \hat{e}(\hat{g}_1, \hat{h}) \cdot \hat{e}(B^*, (\hat{u}' \textstyle\prod_{i:M_i^*=1} \hat{u}_i)) = \hat{e}(A^*, g).$$

We conclude that $\hat{\mathcal{A}}$ outputs a forgery with the same probability as $\mathcal{A}$. Since, by assumption, the Waters scheme is secure, this completes the proof. ■

We do not know how to prove unforgeability of the above scheme with respect to the stronger Definition 7.

References

[1] M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n signatures from a variety of keys. In *Advances in Cryptology — Asiacrypt 2002*.

[2] B. Adida, S. Hohenberger, and R. Rivest. Ad-hoc-group signatures from hijacked keypairs. `http://theory.lcs.mit.edu/~srhohen/papers/AHR.pdf`, June 2005.

[3] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *Advances in Cryptology — Eurocrypt 2003*.

[4] E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *Advances in Cryptology — Crypto 2002*.

[5] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology — Eurocrypt '91*.

[6] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology — Crypto '94*.

[7] I. Damgård and J. Nielsen. Improved non-committing encryption schemes based on a general complexity assumption. In *Advances in Cryptology — Crypto 2000*.

[8] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous identification in ad hoc groups. In *Advances in Cryptology — Eurocrypt 2004*.

[9] C. Dwork and M. Naor. Zaps and their applications. In *Proc. 41st Annual Symposium on Foundations of Computer Science (FOCS '00)*. IEEE Computer Society, 2000.

[10] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs under general assumptions. *SIAM J. Computing*, 29(1):1–28, 1999.

[11] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology — Crypto '86*.

[12] J. Herranz and G. Sáez. Forking lemmas for ring signature schemes. In *Indocrypt*, 2003.

[13] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In *Advances in Cryptology — Eurocrypt '96*.

[14] J. Liu, V. Wei, and D. Wong. Linkable spontaneous anonymous group signatures for ad hoc groups. In *ACISP 2004*.

[15] M. Naor. Deniable ring authentication. In *Advances in Cryptology — Crypto 2002*.

[16] R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Asiacrypt 2001*. Full version available at `http://www.mit.edu/~tauman` and to appear in *Essays in Theoretical Computer Science: in Memory of Shimon Even*.

[17] B. Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology — Eurocrypt 2005*.

[18] J. Xu, Z. Zhang, and D. Feng. A ring signature scheme using bilinear pairings. In *Workshop on Information Security Applications (WISA) 2004*.

[19] F. Zhang and K. Kim. ID-based blind signature and ring signature from pairings. In *Advances in Cryptology — Asiacrypt 2002*.

A ZAPs

Let L be an $\mathcal{NP}$ language with associated polynomial-time *witness relation* $\mathcal{R}_L$ (i.e., such that $L \stackrel{\text{def}}{=} \{x \mid \exists w : (x,w) \in \mathcal{R}_L\}$). If $(x,w) \in \mathcal{R}_L$ we refer to x as the *statement* and w as the associated *witness* for x. We now recall the definition of a ZAP from [9]:

Definition 9 [**ZAP**] A ZAP for an $\mathcal{NP}$ language L with associated witness relation $\mathcal{R}_L$ is a triple $(\ell, \mathcal{P}, \mathcal{V})$, where $\ell(\cdot)$ is a polynomial, $\mathcal{P}$ is a PPT algorithm, and $\mathcal{V}$ is polynomial-time deterministic algorithm, and such that.

Completeness For[10] any $(x,w) \in \mathcal{R}_L$ and any $r \in \{0,1\}^{\ell(k)}$:

$$\mathbf{Pr}[\pi \leftarrow \mathcal{P}_r(x,w) : \mathcal{V}_r(x,\pi) = 1] = 1.$$

Adaptive soundness There exists a negligible function ε such that

$$\mathbf{Pr}[r \leftarrow \{0,1\}^{\ell(k)} : \exists (x,\pi) : \quad x \notin L \text{ and } \mathcal{V}_r(x,\pi) = 1] \leq \varepsilon(k).$$

Witness indistinguishability (Informal) For any $x \in L$, any pair of witnesses w_0, w_1 for x, and any $r \in \{0,1\}^{\ell(k)}$, the distributions $\{\mathcal{P}_r(x,w_0)\}$ and $\{\mathcal{P}_r(x,w_1)\}$ are computationally indistinguishable. (Note: more formally, we need to speak in terms of sequences $\{r_k \in \{0,1\}^{\ell(k)}\}$, $\{x_k\}$, and $\{(w_{k,0}, w_{k,1})\}$ but we avoid doing so for ease of exposition.)

A ZAP is used in the following way: The verifier generates a random first message $r \leftarrow \{0,1\}^{\ell(k)}$ and sends it to the prover $\mathcal{P}$. The prover, given r, a statement x, and associated witness w, sends $\pi \leftarrow \mathcal{P}_r(x,w)$ to the verifier. The verifier then runs $\mathcal{V}_r(x,\pi)$ and accepts iff the output is 1.

[10] We remark that the definition in [9] allows for a negligible completeness error. However, their construction achieves perfect completeness when instantiated using the NIZK of [10].

B Proofs of Claims 1–4

Proof of Claim 1: Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a ring signature scheme satisfying the conditions stated in the claim. Construct the following scheme Π': the key generation algorithm $\mathsf{Gen}'(1^k)$ computes $(PK, SK) \leftarrow \mathsf{Gen}(1^k)$ and outputs $PK' = 0|PK$ and $SK' = SK$. The signing algorithm $\mathsf{Sign}'_{s,SK_s}(M, R)$ checks whether all public keys in R begin with a "0": if so, it outputs $\sigma \leftarrow \mathsf{Sign}_{s,SK_s}(M, \bar{R})$ (where $\bar{R}$ contains the same keys as R, but with the leading bit of each key removed); otherwise, it outputs s. The verification algorithm is modified in the obvious way (recall that completeness is only required to hold for rings containing honestly-generated public keys).

Clearly, the above scheme does not achieve anonymity w.r.t. adversarially-chosen keys. On the other hand, it clearly does remain (unconditionally) anonymous in the basic sense. It is also not difficult to see that it remains unforgeable w.r.t. insider corruption.

Proof of Claim 2: Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a ring signature scheme satisfying the conditions stated in the claim, and assume a symmetric-key encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ (which exists given the assumption of the claim). Construct scheme Π' as follows: the key generation algorithm $\mathsf{Gen}'(1^k)$ computes $(PK, SK) \leftarrow \mathsf{Gen}(1^k)$ but additionally chooses $\kappa \leftarrow \{0,1\}^k$; it outputs $PK' = PK$ and $SK' = SK|\kappa$. The signing algorithm $\mathsf{Sign}'_{s,SK_s|\kappa}(M, R)$ computes $\sigma \leftarrow \mathsf{Sign}_{s,SK_s}(M, R)$ and $C \leftarrow \mathsf{Enc}_\kappa(s)$; it outputs the signature (σ, C). Verification is changed in the obvious way.

The scheme does not achieve anonymity under attribution attacks since, given a signature computed by (say) a user with secret key $SK|\kappa$ with respect to any ring, as long as the adversary has all-but-one of the secret keys of the members of the ring (and, in particular, has the $\{\kappa_i\}$ values for all-but-one of the members), it can determine who the correct signer is with all but negligible probability. On the other hand, it is not hard to show that the scheme remains anonymous w.r.t. adversarially-generated keys and also remains unforgeable w.r.t. insider corruption.

Proof of Claim 3: Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a ring signature scheme satisfying the conditions of the claim. Construct $\Pi' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ as follows. The key generation algorithm Gen' is the same as Gen. The signing algorithm $\mathsf{Sign}'_{s,SK_s}(M, R)$ sets $R' = R \cup \{M\}$ (where M is treated as a public key) and computes $\sigma_1 \leftarrow \mathsf{Sign}_{s,SK_s}(M, R)$ and $\sigma_2 \leftarrow \mathsf{Sign}_{s,SK_s}(0^k, R')$. The output is the signature (σ_1, σ_2). To verify a signature (σ_1, σ_2) (using Vrfy'), simply verify that signature σ_1 is correct (using Vrfy).

It is not hard to see that the scheme is insecure against chosen-subring attacks. Specifically, consider the adversary $\mathcal{A}$ who receives the set of public keys (PK_1, PK_2, PK_3) and then requests a signature on the message $M = PK_3$ with respect to the ring $R = (PK_1, PK_2)$. Let (σ_1, σ_2) be the response of the signing oracle. $\mathcal{A}$ outputs $(0^k, (\sigma_2, \sigma_2), (PK_1, PK_2, PK_3))$ and terminates. Note that (σ_2, σ_2) is a valid ring signature (with respect to the scheme Π') for the message 0^k with respect to the ring (PK_1, PK_2, PK_3). Also note that $\mathcal{A}$ never requested a signature for such a message-ring pair. We therefore conclude that $\mathcal{A}$ succeeds in producing a valid forgery with probability 1.

On the other hand, it is not difficult to show that Π' remains anonymous against key exposures (we omit the details).

Π' is also unforgeable against fixed-ring attacks. We prove this by contradiction. Let $\mathcal{A}'$ be an adversary that breaks the unforgeability of Π' against fixed-ring attacks. We construct an adversary $\mathcal{A}$ that breaks the unforgeability of Π w.r.t. insider attacks. $\mathcal{A}$ takes as input a ring $S = (PK_1, \ldots, PK_n)$ and feeds it to $\mathcal{A}'$. When $\mathcal{A}'$ requests a signature (under Sign') on the message M with respect to the ring R, $\mathcal{A}$ uses its signing oracle to obtain the two components σ_1 and σ_2. Note that $\mathcal{A}$ can obtain σ_2, because he can request a signature on a ring that contains public keys of its choice (M in this case); here is where we use the fact that Π is unforgeable

w.r.t. insider attacks. When $\mathcal{A}'$ outputs a candidate forgery $(M, (\sigma_1, \sigma_2))$, then $\mathcal{A}$ outputs σ_1 as a candidate forgery for message M with respect to the ring S. Note that if the output of $\mathcal{A}'$ is valid signature under Π', then the output of $\mathcal{A}$ is a valid signature under Π. Also, if $\mathcal{A}'$ never requested a signature on M, then $\mathcal{A}$ never requested a signature on M with respect to the ring S. We conclude that $\mathcal{A}$ outputs a valid forgery whenever $\mathcal{A}'$ does.

Proof of Claim 4: Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a scheme satisfying the conditions of the claim. We construct the scheme Π' as follows. The key generation algorithm Gen' runs Gen to obtain (PK, SK), then outputs $PK' = 0|PK$ and $SK' = SK$. We will say that a public key is "good" if it begins with a zero and that it is "bad" if it begins with a one. We note that all public keys generated by Gen' are "good".

The signing algorithm $\mathsf{Sign}'_{s,SK_s}(M, R)$ proceeds as follows. If $R[s] = PK'_s$ is "bad", then the algorithm returns $\perp$. Otherwise, Sign' sets R' to be the ring consisting of only the "good" public keys from the ring R, with the initial bit stripped. It then computes $\sigma \leftarrow \mathsf{Sign}_{s,SK_s}(M, R')$ and outputs this as the signature. The verification algorithm Vrfy' is modified in the appropriate way.

Π' is not unforgeable w.r.t. insider corruption. To see this, consider the adversary $\mathcal{A}$ who receives public keys (PK'_1, PK'_2). Next, $\mathcal{A}$ generates an arbitrary "bad" public key $PK' = 1 \mid PK''$. The adversary then requests a signature on an arbitrary message M with respect to the ring (PK'_1, PK'_2, PK') on behalf of the signer holding PK'_1. The signing oracle returns a signature σ that is a valid signature for message M respect to the ring (PK'_1, PK'_2) (recall that PK' is ignored, since it is "bad"). But now $\mathcal{A}$ can output the forgery $(M, \sigma, (PK'_1, PK'_2))$ and succeed with probability 1.

It is not hard to see that Π' remains unforgeable against chosen-subring attacks (since, in such attacks, the adversary can only request signatures with respect to rings that consist only of "good" public keys). One can also easily show that Π' remains anonymous w.r.t. key exposures.

B.1 The Herranz-Sáez Ring Signature Scheme

In this section we show that the Herranz-Sáez ring signature scheme [12] gives a natural separation between unforgeability against fixed-ring and chosen-subring attacks. Recall that, in the proof of Claim 3 (above), we presented a ring signature scheme that is anonymous against key exposure and unforgeable against fixed-ring attacks, but not unforgeable against chosen-subring attacks. Such a scheme was artificial and purposely designed to achieve the separation under a very relaxed assumption. In contrast, we now show a "natural" scheme, introduced in previous work, that illustrates the same separation albeit under specific and stronger assumptions.

We first review the Herranz-Sáez ring signature scheme. Let $\mathbb{G}, \mathbb{G}_1$ be groups of prime order q. Let $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ be a hash function modeled as a random oracle. We assume that H, q, $\mathbb{G}$, and a generator $g \in \mathbb{G}$ are publicly known. The scheme is defined as follows:

Key Generation. Choose $x \leftarrow \mathbb{Z}_q$ and set $y = g^x$. The public key is y and the secret key is x.

Ring Signing. To sign message M with respect to the ring $R = \{y_1, \ldots, y_n\}$ using secret key x_s, proceed as follows:

1. for $i = 1, \ldots, n$, $i \neq s$, choose random $a_i \leftarrow \mathbb{Z}_q$ and set $C_i = g^{a_i}$;
2. choose random $a_s \leftarrow \mathbb{Z}_q$;

3. compute C_s and b as follows:

$$\begin{aligned} C_s &= g^{a_s} \prod_{i \neq s} y_i^{-H(M,C_i)} \\ b &= a_s + \sum_{i \neq s} a_i + x_s H(M, C_s); \end{aligned}$$

4. in the unlikely event that the C_i are not all distinct, restart from the beginning;

5. output the signature $\sigma = (b, C_1, \ldots, C_n)$.

Ring Verification. To verify the signature $(b, C_1, \ldots, C_n)$ on message M with respect to the ring $R = \{y_1, \ldots, y_n\}$, check that the C_i are all distinct and that:

$$g^b \stackrel{?}{=} \prod_{i=1}^{n} C_i \cdot y_i^{H(M,R_i)}.$$

It is not hard to see that the scheme above is unconditionally anonymous against key exposure, even in the standard model. This is because a ring signature on message M with respect to a ring R is a uniformly random sample from the set of the tuples $(b, C_1, \ldots, C_n)$ that satisfy the ring verification condition, and this distribution is independent of the index s of the signing key used. Additionally, Herranz and Sáez [12] prove that this scheme is unforgeable against fixed-ring attacks[11] under the discrete logarithm assumption in the random oracle model.

However, we point out that the Herranz-Sáez scheme is *not* unforgeable against chosen-subring attacks. Consider an adversary that requests two signatures on the same arbitrary message M with respect to the *disjoint* rings $R = (y_1, \ldots, y_n)$ and $R' = (y'_1, \ldots, y'_m)$, obtaining signature $\sigma = (b, C_1, \ldots, C_n)$ in the first case and $\sigma' = (b', C'_1, \ldots, C'_n)$ in the second. The adversary can then output the forged signature

$$\sigma^* = (b + b', C_1, \ldots, C_n, C'_1, \ldots, C'_m)$$

on message M with respect to the ring $R \cup R' = (y_1, \ldots, y_n, y'_1, \ldots, y'_m)$. Applying the ring verification algorithm shows that this is indeed a valid forgery (except in the unlikely case that $C_i = C'_j$ for some i, j).

C Proof of Theorem 2

We restate Theorem 2 for convenience:

Under the assumptions of Theorem 1 and assuming $(\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$ *has an oblivious public-key generator, the modified ring signature scheme described above is (computationally) anonymous against key exposure, and unforgeable w.r.t. insider corruption.*

Proof (Sketch) The proof of unforgeability follows immediately from Theorem 1 since, by Definition 8, the adversary cannot distinguish between the original scheme (in which the encryption key is generated using EGen) and the modified scheme (in which the encryption key is generated using $\mathsf{OblEGen}$).

[11] The authors do not formally define unforgeability, but an inspection of their proof of security reveals that their notion of unforgeability matches our Definition 5.

We now argue that the modified scheme achieves anonymity against key exposure. First we note that the anonymity against attribution attacks claimed in Theorem 1 holds even when the adversary is given all random coins used to generate (PK_0, SK_0) *except* for those coins used to generate $pk_{E,0}$ (using EGen). Now, if there exists a PPT adversary $\mathcal{A}$ that breaks anonymity of the modified scheme in the sense of key exposure, we can use it to construct a PPT adversary $\mathcal{A}'$ that breaks anonymity of the original scheme against attribution attacks. $\mathcal{A}'$ receives PK_0, the random coins $\omega_{S,1}, \omega_{E,1}$ used to generate (PK_1, SK_1), and the random coins $\omega_{S,0}$ used to generate $pk_{S,0}$ (i.e., $\mathcal{A}$ is *not* given the coins used to generate $pk_{E,0}$). Next, $\mathcal{A}'$ runs $\omega'_{E,0} \leftarrow \mathsf{OblRand}(pk_{E_0})$ and $\omega'_{E,1} \leftarrow \mathsf{OblRand}(pk_{E,1})$ and gives to $\mathcal{A}$ the public key PK_0 it received as well as the random coins $\omega_{S,0}, \omega'_{E,0}, \omega_{S,1}, \omega'_{E,1}$. The remainder of $\mathcal{A}$'s execution is simulated in the natural way by $\mathcal{A}'$.

Now, Definition 8 implies that the advantage of $\mathcal{A}$ in the above is negligibly close to the advantage of $\mathcal{A}$ in attacking the modified scheme in the sense of key exposure attacks. But the advantage of $\mathcal{A}$ in the above is exactly the advantage of $\mathcal{A}'$ in attacking the original scheme via key attribution attacks. Since we have already proved that the original scheme is anonymous against attribution attacks (cf. Theorem 1), we see that the modified scheme is anonymous against key exposure. ■