Distributed Ring Signatures for Identity-Based Scenarios

Javier Herranz and Germán Sáez

Dept. Matemàtica Aplicada IV, Universitat Politècnica de Catalunya
C. Jordi Girona, 1-3, Mòdul C3, Campus Nord, 08034 Barcelona, Spain
e-mail: {jherranz,german}@mat.upc.es

Abstract

In a ring signature scheme, a signer in a subset (or *ring*) of potential signers produces a signature of a message in such a way that the receiver can verify that the signature comes from a member of the ring, but cannot know which member has actually signed.

In this work, we extend this concept to that of distributed ring signatures, where a subset of users cooperate to compute a distributed anonymous signature on a message, on behalf of a family of subsets. We propose two schemes, one for general families of subsets, and a more efficient one for threshold families of subsets. The security of both proposals is formally proved, assuming the hardness of the Computational Diffie-Hellman problem.

Our two schemes run in an identity-based scenario, where public keys of the users can be derived from their identities. This fact avoids the necessity of digital certificates, and therefore allows more efficient implementations of such systems.

1 Introduction

In a standard public key scenario: each user U has a secret key SK_U, and usually the matching public key PK_U is computed from SK_U. In these scenarios, a serious problem appears: how can one be sure that PK_U is actually the public key of user U, or in other words, that the only person who knows SK_U is user U? For example, a different user U' can generate $SK_{U'}$, compute the matching public key and broadcast it as if it was the public key of user U.

To solve this problem, the public keys of the users are authenticated via a Public Key Infrastructure (PKI) based on digital certificates: a user who wants to use a public key cryptosystem turns to a certification authority, who signs a message linking the public key PK_U with the identity of user U. Later, a user who must use public key PK_U (to encrypt a message or to verify a signature, for example) must first verify that the certificate which links U and PK_U is still valid. Other problems appear when revocation of some certificate is necessary, because a secret key SK_U corresponding to a certified PK_U has been compromised, for instance.

All these facts make the use of cryptographic protocols less efficient in the real life. Thus, any possible alternative which avoids the necessity of digital certificates is welcome in order to design more efficient public key cryptosystems.

Shamir introduced in 1984 the concept of *identity-based* (from now on, ID-based) cryptography [17]. The idea is that the public key of a user can be publicly computed from his identity (for example, from a complete name, an e-mail or an IP address). Then, the secret key is derived from the public key. In this way, digital certificates are not necessary, because anyone can easily verify that some public key PK_U corresponds in fact to user U.

The process that generates secret keys from public keys must be executed by an external entity, known as the *master*. Thus, the master knows the secret keys of all the users of the system. A way to relax this negative point could be to consider a set of master entities which share the secret information.

A clear example of cryptographic schemes where the use of digital certificates dramatically decreases the efficiency of the implementation are ring signature schemes, because of the number of public keys that can be involved in any basic operation (signature and verification).

In a ring signature scheme, an entity signs a message on behalf of a set (or ring) of members that includes himself. The verifier of the signature is convinced that it was produced by some member of the ring, but he does not obtain any information about which member of the ring actually signed.

Ring signatures are a useful tool to provide anonymity in some scenarios. For example, if a member of a group wants to leak to the media a secret information about the group, he can sign this information using a ring scheme. Everybody will be convinced that the information comes from the group itself, but anybody could accuse him of leaking the secret.

The concept of ring signatures was formally introduced in [15]. After that, many proposals of ring signature schemes have been published [4, 1, 20, 10, 6, 12]. Two of these proposals [20, 12] are ring signature schemes which work in ID-based scenarios.

We consider in this work the following extension of the concept of ring signature. Suppose that a set of users $\mathcal{U}_s$ want to anonymously sign some message, in such a way that the verifier of the signature will be convinced that at least the members of some set have all agreed in signing this message, but he could not know which set has actually computed the signature, among the sets of a certain family of possibly signing sets (we will denote this family as the access structure of the signature).

Members of $\mathcal{U}_s$ can freely choose the rest of users and the family of sets that will form the access structure (in an *ad-hoc* way). We denote as $\mathcal{U} = \{\mathcal{U}_1, \dots, \mathcal{U}_d\}$ the access structure, where the set $\mathcal{U}_s$ must be one of the sets in $\mathcal{U}$.

The resulting signature will be a ring signature, taking as ring the set $\mathcal{U}$. In this way, the verifier will be convinced that at least *all* the members of some set in $\mathcal{U}$ have cooperated to compute the signature, but he will not have any information about which set in $\mathcal{U}$ is the actual author of the signature.

An example of such a situation can be thought inside a company: workers of the company are divided in different branches according to their functionality. Suppose all the workers in some branch of the company want to sign a message where they complain about some point of the politics of the company. They want the head of the company to know that many different workers disagree with him, but not to know who is complaining. Members of the complaining branch can form an access structure with all the branches of the company, and compute a ring signature for this structure. The head of the company will be convinced that the complaint comes from all the members of some of the branches, but he will never know which branch dared to complain.

This extension of ring signature schemes, that we denote distributed ring signature schemes, was first considered in [4]. Their specific RSA-based scheme, however, runs only when the ad-hoc access structures are necessarily threshold (that is, they contain all the sets with a minimum number of users). Recently a more general proposal, which allows the use of different types of keys, has appeared in [19]; but again this scheme is valid only for threshold access structures. In [11], a scheme for general access structures is proposed, for scenarios based on Discrete Logarithm keys.

In Section 3, we provide definitions for the protocols that take part in a distributed ring signature scheme, and the security properties that such schemes must satisfy. Then we propose the first distributed ring signature schemes for ID-based scenarios. We first propose, in Section 4, a scheme which works for any general access structure. Then, in Section 5, we propose a more efficient scheme for the particular case of threshold access structures. We provide formal and exact proofs of the security of the two proposed protocols. Roughly speaking, we prove that they achieve unconditional anonymity and computational unforgeability, assuming that the Computational Diffie-Hellman problem is hard to solve. This well-known problem is explained in Section 2, as long as other tools that we use in the design and analysis of our schemes, for example bilinear pairings. Finally, we conclude the work in Section 6.

2 Preliminaries

2.1 Bilinear Pairings

Let $\mathbb{G}_1$ be an additive group of prime order q, generated by some element P. Let $\mathbb{G}_2$ be a multiplicative group with the same order q.

A *bilinear pairing* is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ with the following three properties:

1. It is bilinear, which means that given elements $A_1, A_2, A_3 \in \mathbb{G}_1$, we have that $e(A_1 + A_2, A_3) = e(A_1, A_3) \cdot e(A_2, A_3)$ and $e(A_1, A_2 + A_3) = e(A_1, A_2) \cdot e(A_1, A_3)$. In particular, for all $a, b \in \mathbb{Z}_q$, we have $e(aP, bP) = e(P, P)^{ab} = e(P, abP) = e(abP, P)$.

2. The map e can be efficiently computed for any possible input pair.

3. The map e is non-degenerate: there exist elements $A_1, A_2 \in \mathbb{G}_1$ such that $e(A_1, A_2) \neq 1_{\mathbb{G}_2}$.

Combining properties 1 and 3, it is easy to see that $e(P, P) \neq 1_{\mathbb{G}_2}$ and that the equality $e(A_1, P) = e(A_2, P)$ implies that $A_1 = A_2$.

The typical way of obtaining such pairings is by deriving them from the Weil or the Tate pairing on an elliptic curve over a finite field. The interested reader is referred to [21] for a complete bibliography of cryptographic works based on pairings.

Let $H_1 : \{0,1\}^* \to \mathbb{G}_1 - \{0\}$ be a hash function. The most usual way to design an ID-based cryptosystem is the following. The master has a secret key $x \in \mathbb{Z}_q^*$, and he publishes the value $Y = xP \in \mathbb{G}_1$.

Every user U of the ID-based system has an identifier $ID_U \in \{0,1\}^*$, that can be an IP address, a telephone number, an e-mail address, etc. The public key of U is then defined to be $PK_U = H_1(ID_U) \in \mathbb{G}_1 - \{0\}$. In this way, everybody can verify the authenticity of a public key without the necessity of certificates.

The user U needs to contact the master to obtain his secret key $SK_U = xPK_U \in \mathbb{G}_1$. The drawback of this approach, as mentioned in the Introduction, is that the master must be completely trusted, because he knows the secret keys of all the users.

2.2 The Computational Diffie-Hellman Problem

We consider the following well-known problem in the additive group $\mathbb{G}_1$ of prime order q, generated by P:

Definition 1. *Given the elements P, aP and bP, for some random values $a, b \in \mathbb{Z}_q^*$, the Computational Diffie-Hellman problem consists of computing the element abP.*

The Computational Diffie-Hellman Assumption asserts that, if the order of $\mathbb{G}_1$ is $q \geq 2^k$, then any polynomial time algorithm that solves the Computational Diffie-Hellman problem has a success probability p_k which is negligible in the security parameter k. In other words, for all polynomial $f()$, there exists an integer k_0 such that $p_k < \frac{1}{f(k)}$, for all $k \geq k_0$.

The security of the ID-based ring signature schemes that we propose in this work is based on the Computational Diffie-Hellman Assumption.

2.3 The Splitting Lemma

We first state a well-known lemma that we will use in some of the security proofs of this paper. A proof of this lemma can be found, for example, in [14].

Lemma 1. *(The Splitting Lemma) Let $A \subset X \times Y$ be a set verifying that $\Pr[(x,y) \in A] \geq \epsilon$. For any $\alpha < \epsilon$, let us define*

$$B = \{(x,y) \in X \times Y | \Pr_{y' \in Y}\left[(x,y') \in A\right] \geq \epsilon - \alpha\} \text{ and } \bar{B} = (X \times Y) \setminus B.$$

Then the following statements hold:

1. $\Pr[B] \geq \alpha$.
2. *for any $(x,y) \in B$, $\Pr_{y' \in Y}[(x,y') \in A] \geq \epsilon - \alpha$.*
3. $\Pr[B|A] \geq \alpha/\epsilon$.

2.4 The Random Oracle Model

Bellare and Rogaway introduced in [3] a paradigm that makes easier the task of proving the security of some cryptographic schemes. This paradigm is the *random oracle model*. In this model, hash functions are seen as oracles that produce a truly random value for each new input. Obviously, if the same input is asked twice, then the outputs must be identical.

The random oracle model is unreal, because any instantiation of a hash function is in fact a deterministic function: once the instantiation is made public, everybody can know which will be the output corresponding to any input. Furthermore, any realization of a random function can be seen as a list with exponential size. But hash functions are part of the public key of the considered cryptographic scheme, and the size of public keys must be polynomial in the security parameter.

Although there are some theoretical works which criticize the paradigm of the random oracle model [5, 13, 2], it is widely believed that proofs in this model guarantee the security of the overall cryptographic scheme, provided the employed hash function has no weakness.

All the security results that we prove in this work are valid in the random oracle model.

2.5 Generic Ring Signature Schemes

Herranz and Sáez define in [10] a family of ring signature schemes that they call *generic* (influenced by the work of Pointcheval and Stern [14], where they give this name to a family of signature schemes which includes Schnorr's one). Consider a security parameter k, a hash function which outputs k-bit long elements, and a ring $\mathcal{U} = \{U_1, \ldots, U_d\}$ of d members. Given the input message m, a generic ring signature scheme produces a tuple $(\mathcal{U}, m, R_1, \ldots, R_d,\ h_1, \ldots, h_d, \sigma)$, where $R_1, \ldots, R_d$ take their values randomly in a large set G in such a way that $R_i \neq R_j$ for all $i \neq j$, h_i is the hash value of $(\mathcal{U}, m, R_i)$, for $1 \leq i \leq d$, and the value σ is fully determined by $R_1, \ldots, R_d, h_1, \ldots, h_d$ and the message m.

Another required condition is that no R_i can appear in a signature with probability greater than $2/2^k$, where k is the security parameter. This condition can be achieved by choosing the set G as large as necessary.

In [10], the authors prove a result, the Ring Forking Lemma, which is useful to prove the security of generic ring signature schemes. We state here a variation of their result, that we will use throughout Section 4. For integers Q and d such that $Q \geq d \geq 1$, we denote as $V_{Q,d}$ the number of d-permutations of Q elements; that is, $V_{Q,d} = Q(Q-1) \cdot \ldots \cdot (Q-d+1)$.

Theorem 1. *(The Ring Forking Lemma) Consider a generic ring signature scheme with security parameter k. Let $\mathcal{A}$ be a probabilistic polynomial time Turing machine which receives as input the digital identifiers of users in a set $\mathcal{U}^*$ and other public data; the machine $\mathcal{A}$ can ask Q queries to the random oracle.*

We assume that $\mathcal{A}$ produces, within time bound T and with non-negligible probability of success ε, a valid ring signature $(\mathcal{U}, m, R_1, \ldots, R_d, h_1, \ldots, h_d, \sigma)$ for a ring

$\mathcal{U} \subset \mathcal{U}^*$ of d users, such that $\mathcal{A}$ does not know any of the secret keys of the users in $\mathcal{U}$.

Then, within time $T' \leq 2T$, and with probability $\varepsilon' \geq \frac{\varepsilon^2}{66 V_{Q,d}}$, we obtain two valid ring signatures $(\mathcal{U}, m, R_1, \ldots, R_d, h_1, \ldots, h_d, \sigma)$ and $(\mathcal{U}, m, R_1, \ldots, R_d, h'_1, \ldots, h'_d, \sigma')$ such that $h_j \neq h'_j$, for some $j \in \{1, \ldots, d\}$ and $h_i = h'_i$ for all $i = 1, \ldots, d$ such that $i \neq j$.

In a PKI scenario, the digital identifier of a user is his public key, which can be verified by means of the corresponding digital certificate. In ID-based scenarios, however, the digital identifier of a user is simply an e-mail or IP address; the public key could be computed directly from this identifier.

3 Distributed Ring Signatures

A distributed ring signature scheme consists of three protocols:

1. **Key generation.** This protocol is executed individually by each user U_i of the system. The input is a security parameter and (possibly) some public parameters, common to all the users of the system. The output consists of a public key PK_i, that the user U_i makes public, and a secret key SK_i, that U_i keeps secret. In ID-based scenarios, this protocol is executed with the help of a master entity.

2. **Distributed ring signature generation.** Suppose users in a subset $\mathcal{U}_s = \{U_1, U_2, \ldots, U_{n_s}\}$ want to compute a ring signature on a message m on behalf of a family of subsets (or access structure) $\mathcal{U} = \{\mathcal{U}_1, \ldots, \mathcal{U}_s, \ldots, \mathcal{U}_d\}$. Then members of $\mathcal{U}_s$ jointly execute this protocol, taking as input the message m, the public keys of all users included in the access structure $\mathcal{U}$ and their own secret keys $SK_1, \ldots, SK_{n_s}$. The output is a signature θ.

3. **Verification of a distributed ring signature.** The recipient of a distributed ring signature checks its validity by running this protocol. It takes as input the message m, the signature θ and all the public keys involved in the access structure $\mathcal{U}$. The output is 1 if the signature is valid, and 0 if it is invalid.

Note that distributed ring signature schemes are related to standard distributed (or threshold) signature schemes [8, 18]. In both cases, the recipient of the signature is convinced that all the users in some subset of the access structure have jointly signed the message, but he does not know which is the signing subset. There are two main differences between these two types of signatures.

- In distributed signature schemes, the same access structure is fixed from the initialization of the system on; in distributed ring signature schemes, however, the signing users choose *ad-hoc* the access structure, just before signing.

- In distributed signature schemes, there is a unique public key for the whole set of users, and the matching secret key is shared among them. On the other hand, in distributed ring signature schemes, each user has his own public and secret keys, and therefore he can use them for other purposes (like individual signatures or encryption).

As we have said in the Introduction, all the distributed ring signature schemes proposed until now work in a traditional PKI scenario, where the validity of the public keys of the users must be checked before using them, by means of digital certificates. Some of them work only for threshold access structures [4, 19], whereas the only proposal which works for general access structures is [11]. In the rest of the work, we use the tools introduced in Section 2 to design and analyze two distributed ring signature schemes which work in ID-based scenarios.

3.1 Security Requirements

A distributed ring signature scheme must satisfy three properties, that we informally describe below.

1. **Correctness:** if a distributed ring signature is generated by properly following the protocol, then the result of the verification is always 1.

2. **Anonymity:** any verifier should not have probability greater than $1/d$ to guess the identity of the subset which has actually computed a distributed ring signature on behalf of an access structure which contains d subsets.

3. **Unforgeability:** among all the proposed definitions of unforgeability [9], we consider the strongest one, *existential unforgeability against chosen message attacks*, adapted to the scenario of distributed ring signatures. We will consider the exact unforgeability of a scheme, that measures all the resources and performances of the adversary. Remember that we analyze the security of our schemes in the random oracle model.

   Such an adversary is given as input a set $\mathcal{U}^*$ of users, and is allowed to corrupt up to Q_e users, obtaining their secret keys. The adversary can also make Q queries to the random oracle which models the behavior of a hash function. Finally, the adversary can require the execution of the signing algorithm for Q_s pairs of messages and rings that it adaptively chooses, obtaining a valid ring signature.

   We say that this adversary is $(T, \varepsilon, Q, Q_e, Q_s)$-successful if it obtains in polynomial time T and with non-negligible probability ε a valid ring signature for some message m and some ring of users $\mathcal{U}$, such that:

   (i) the pair formed by the message m and the ring $\mathcal{U}$ has not been asked to the signing oracle during the attack; and

   (ii) none of the users in the ring $\mathcal{U}$ has been corrupted by the adversary.

Finally, we say that a distributed ring signature scheme is $(T, \varepsilon, Q, Q_e, Q_s)$-unforgeable if there does not exist any $(T, \varepsilon, Q, Q_e, Q_s)$-successful adversary against it.

4 An ID-Based Distributed Ring Signature Scheme for General Access Structures

We will assume that any specific set of users can always have access to an authenticated broadcast channel, while the information in this channel remains secret to the rest of users. This can be achieved using different cryptographic techniques (for example, broadcast encryption schemes [7]).

The protocols of our distributed ring signature scheme work as follows:

Key generation: let $\mathbb{G}_1$ be an additive group of prime order q, generated by some element P. Let $\mathbb{G}_2$ be a multiplicative group with the same order q. We need $q \geq 2^k + \hat{d}$, where k is the security parameter of the scheme and $\hat{d}$ is the maximum possible number of subsets in an access structure. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear pairing as defined in Section 2.1. Let $H_1 : \{0,1\}^* \to \mathbb{G}_1^*$ and $H_2 : \{0,1\}^* \to \mathbb{Z}_q$ be two hash functions.

The master entity chooses at random his secret key $x \in \mathbb{Z}_q^*$ and publishes the value $Y = xP$.

Secret key extraction: any user U_i of the system, with identity ID_i, has public key $PK_i = H_1(ID_i)$. When he requests the master for his matching secret key, he obtains the value $SK_i = xPK_i$.

Distributed ring signature generation: assume that a set $\mathcal{U}_s$ of users (for simplicity, we denote them as $\mathcal{U}_s = \{U_1, U_2, \ldots, U_{n_s}\}$) want to compute an anonymous signature. They choose the access structure $\mathcal{U} = \{\mathcal{U}_1, \ldots, \mathcal{U}_d\}$, such that $\mathcal{U}_s \in \mathcal{U}$.

For each of the sets $\mathcal{U}_i \in \mathcal{U}$, we consider the public value

$$Y_i = \sum_{U_j \in \mathcal{U}_i} PK_j \ .$$

The algorithm for computing the ring signature is the following:

1. Each user $U_j \in \mathcal{U}_s$ chooses at random $a_{sj} \in \mathbb{Z}_q^*$ and computes $R_{sj} = e(a_{sj}P, P)$. He broadcasts the value R_{sj}.

2. One of the users in $\mathcal{U}_s$, for example U_1, chooses, for all $i = 1, \ldots, d$, $i \neq s$, random values $a_i \in \mathbb{Z}_q^*$, pairwise different, and computes $R_i = e(a_iP, P)$. He broadcasts these values R_i, and therefore all the members of $\mathcal{U}_s$ can compute $h_i = H_2(\mathcal{U}, m, R_i)$, for all $i = 1, \ldots, d$, $i \neq s$.

3. Members of $\mathcal{U}_s$ compute the value

$$R_s = e(-Y, \sum_{i \neq s} h_i Y_i) \prod_{U_j \in \mathcal{U}_s} R_{sj} \ .$$

If $R_s = 1_{\mathbb{G}_2}$ or $R_s = R_i$ for some $i = 1, \dots, d$, $i \neq s$, they return to step 1. Members of $\mathcal{U}_s$ can then compute $h_s = H_2(\mathcal{U}, m, R_s)$.

4. User U_1 computes and broadcasts the value $\sigma_1 = a_{s1}P + h_s SK_1 + \sum_{1 \le i \le d, i \neq s} a_i P \in \mathbb{G}_1$.

5. For $j = 2, \dots, n_s$, player U_j computes and broadcasts the value $\sigma_j = a_{sj}P + h_s SK_j + \sigma_{j-1} \in \mathbb{G}_1$.

6. Define $\sigma = \sigma_{n_s}$. The resulting valid signature is $(\mathcal{U}, m, R_1, \dots, R_d, h_1, \dots, h_d, \sigma)$.

Verification of a distributed ring signature: the validity of the signature is verified by the recipient of the message by checking that $h_i = H_2(\mathcal{U}, m, R_i)$, for $i = 1, \dots, d$ and that

$$e(\sigma, P) = e(Y, \sum_{i=1}^{d} h_i Y_i) \prod_{1 \le i \le d} R_i \ ,$$

where $Y_i = \sum_{U_j \in \mathcal{U}_i} PK_j$, for all the sets $\mathcal{U}_i$ in the access structure $\mathcal{U}$.

4.1 Some Remarks

- The ID-based distributed ring signature scheme proposed above allows to detect whether some of the signers in the subset $\mathcal{U}_s$ tries to boycott the process of signing. In effect, the correctness of the values σ_j can be verified by the rest of the signers, by using public information. Namely, for $j = 1$ the equation

$$e(\sigma_1, P) = R_{s1} \cdot e(h_s PK_1, Y) \cdot \prod_{1 \le i \le d, i \neq s} R_i$$

  must be satisfied. For the rest of users $U_j \in \mathcal{U}_s$, with $j \neq 1$, the equation that must be checked is

$$e(\sigma_j, P) = R_{sj} \cdot e(h_s PK_j, Y) \cdot e(\sigma_{j-1}, P).$$

- We consider the case where the signing users form an ad-hoc access structure. But the scheme runs as well if the access structure is fixed. In this case the resulting scheme would be in fact a distributed signature scheme (or threshold signature scheme, if the access structure is a threshold one).

- Note that this distributed ring signature scheme can be seen as a generic ring signature scheme, as defined in Section 2.5. In effect, we can see the subsets $\mathcal{U}_i$ in the access structure $\mathcal{U}$ as individual users of a standard ring signature scheme, with public keys $PK_i = Y_i = \sum_{U_j \in \mathcal{U}_i} PK_j$. There is a random value R_i for each subset $\mathcal{U}_i$, and a particular R_i appears with probability at most $1/(q - \hat{d}) \le 1/2^k$, as desired. Therefore, in the security analysis, we could use the Ring Forking Lemma stated in Section 2.5.

- The efficiency of the scheme depends on the total number of users and the number of sets in the access structure. Therefore, it is a good solution for situations where the number of sets is small. If the access structure is a threshold one, then the number of sets is very large (it is exactly $\binom{\ell}{t}$, if ℓ is the total number of users and t is the threshold). We design in Section 5 a more efficient proposal, specific for the threshold case.

4.2 Correctness and Anonymity of the Scheme

A ring signature $(\mathcal{U}, m, R_1, \ldots, R_d, h_1, \ldots, h_d, \sigma)$ computed by following the method explained above satisfies the verification equation. In effect:

$$e(\sigma, P) = e(\sigma_{n_s}, P) = e\left((\sum_{U_j \in \mathcal{U}_s} a_{sj}P + h_s SK_j) \ + \ (\sum_{1 \le i \le d, i \ne s} a_i P) \ , \ P \right) =$$

$$\left(\prod_{U_j \in \mathcal{U}_s} e(a_{sj}P, P) \cdot e(h_s x PK_j, P) \right) \prod_{1 \le i \le d, i \ne s} e(a_i P, P) =$$

$$= \left(\prod_{U_j \in \mathcal{U}_s} R_{sj} \cdot e(h_s PK_j, xP) \right) \prod_{1 \le i \le d, i \ne s} R_i =$$

$$= R_s \cdot \ e(\sum_{1 \le i \le d, i \ne s} h_i Y_i \ , \ Y) \ \cdot \ e(h_s \sum_{U_j \in \mathcal{U}_s} PK_j \ , \ Y) \prod_{1 \le i \le d, i \ne s} R_i = e(\sum_{i=1}^{d} h_i Y_i, Y) \prod_{i=1}^{d} R_i$$

With respect to the anonymity of the scheme, we can argue as follows: let $Sig = (\mathcal{U}, m, R_1, \ldots, R_d, h_1, \ldots, h_d, \sigma)$ be a valid ring signature of a message m on behalf of the access structure $\mathcal{U} = \{\mathcal{U}_1, \ldots, \mathcal{U}_d\}$. Let $\mathcal{U}_s$ be a subset of the access structure. We now find the probability that members of $\mathcal{U}_s$ compute exactly the ring signature Sig, when they produce a ring signature of message m on behalf of the access $\mathcal{U}$, by following the proposed scheme.

The probability that members of $\mathcal{U}_s$ compute all the values $R_i \neq 1_{\mathbb{G}_2}$ of Sig, pairwise different for $1 \le i \le d$, $i \ne s$, is $\frac{1}{q-1} \cdot \frac{1}{q-2} \cdot \ldots \cdot \frac{1}{q-d+1}$. Then, the probability that members of $\mathcal{U}_s$ choose values $a_{sj} \in \mathbb{Z}_q$ that lead to the value R_s of Sig, among all possible values for R_s different to $1_{\mathbb{G}_2}$ and different to all R_i with $i \ne s$, is $\frac{1}{q-d}$.

Summing up, the probability that users in $\mathcal{U}_s$ generate exactly the ring signature Sig is

$$\frac{1}{q-1} \cdot \frac{1}{q-2} \cdot \ldots \cdot \frac{1}{q-d+1} \cdot \frac{1}{q-d} \ = \ \frac{1}{V_{q-1,d}}$$

and this probability does not depend on the subset $\mathcal{U}_s$, so it is the same for all the subsets of the access structure. This fact proves the unconditional anonymity of the scheme.

4.3 Unforgeability of the Scheme

We first remember the definition of an adversary against distributed ring signature schemes, introduced in Section 3.1: a $(T, \varepsilon, Q_1, Q_2, Q_e, Q_s)$-successful attacker against a ring signature scheme is an algorithm which is given a list of identities ID_i, runs in time T, makes Q_1 queries to the random oracle H_1, Q_2 queries to the random oracle H_2, asks for Q_e secret keys of different users and asks for Q_s valid ring signatures. With probability ε, this algorithm obtains a valid new signature for a pair $(\mathcal{U}, m)$, such that all the sets of the access structure $\mathcal{U}$ contain at least one user whose secret key has not been queried by the adversary.

In the following theorem, we relate the difficulty of forging our ID-based distributed ring signature scheme with the difficulty of solving the Computational Diffie-Hellman problem.

Theorem 2. *Let $\mathcal{A}$ be a $(T, \varepsilon, Q_1, Q_2, Q_e, Q_s)$-successful adversary against the ID-based distributed ring signature scheme proposed above, such that the success probability ε of $\mathcal{A}$ is non-negligible in the security parameter k.*

We denote by $\hat{n}$ the maximum possible cardinality of the subsets and by $\hat{d}$ the maximum possible number of subsets in the access structures considered in the system.

Let μ be any value such that $\left(1 - \frac{\varepsilon}{12}\right)^{1/Q_e} \leq \mu < 1$.

Then the Computational Diffie-Hellman problem in $\mathbb{G}_1$ can be solved in time $T' \leq 2T + 2Q_1 + 2Q_2 + 2(\hat{d} + \hat{n})Q_s$ and with probability $\varepsilon' \geq \frac{(1-\mu)^{2\hat{d}+1}}{200 V_{Q_2,\hat{d}}}\, \varepsilon^2$.

Proof. Since $\mathcal{A}$'s success probability ε is non-negligible in k, we can assume that $\varepsilon \geq \frac{12\ V_{Q,\hat{d}} + 6(Q+Q_s)^2}{(1-\mu)^{\hat{d}} 2^k}$.

Let (P, aP, bP) be the input of an instance of the Computational Diffie-Hellman problem in $\mathbb{G}_1$. Here P is a generator of $\mathbb{G}_1$, with prime order q, and the elements a, b are taken uniformly at random in $\mathbb{Z}_q^*$.

We construct a probabilistic polynomial time Turing machine $\mathcal{F}$ which will solve the given instance of the Computational Diffie-Hellman problem; that is, it will compute the value abP. This machine $\mathcal{B}$ is given as input the digital identifiers ID_i of users U_i in a set $\mathcal{U}^*$. It will use the attacker $\mathcal{A}$ as a sub-routine, so it must perfectly simulate the environment of the attacker $\mathcal{A}$. The machine $\mathcal{F}$ is also allowed to make Q_2 queries to the random oracle for the hash function H_2.

The public data (P, aP, bP) is given to the machine $\mathcal{F}$, and the public key of the master entity is defined to be $Y = aP$. Then $\mathcal{F}$ runs the attacker $\mathcal{A}$ against our ID-based distributed ring signature scheme, answering to all the queries that $\mathcal{A}$ makes. First of all, $\mathcal{F}$ gives the public key $Y = aP$ to the attacker $\mathcal{A}$.

Without loss of generality, we can assume that $\mathcal{A}$ asks the random oracle H_1 for the value $H_1(ID)$ before asking for the secret key of ID.

The machine $\mathcal{F}$ constructs a table TAB_{H_1} to simulate the random oracle H_1. Every time an identity ID_j is asked by $\mathcal{A}$ to the oracle H_1, the machine $\mathcal{F}$ first checks if this input is already in the table; if this is the case, then $\mathcal{F}$ returns to $\mathcal{A}$ the corresponding relation $H_1(ID_j) = PK_j$. Otherwise, $\mathcal{F}$ acts as follows: with probability μ, it chooses the random bit $c_j = 0$; in this case, $\mathcal{F}$ chooses a different $x_j \in \mathbb{Z}_q^*$ at random and defines $PK_j = x_j P$ and $SK_j = x_j Y$. On the other hand, with probability $1 - \mu$, the machine $\mathcal{F}$ chooses $c_j = 1$; in this case, it chooses a different $\alpha_j \in \mathbb{Z}_q^*$ at random and defines $PK_j = (\alpha_j)bP$ and $SK_j = \perp$. The values $(ID_j, PK_j, x_j$ or $\alpha_j, SK_j, c_j)$ are stored in a new entry of TAB_{H_1}, and the relation $H_1(ID_j) = PK_j$ is sent to $\mathcal{A}$. The condition $PK_j \neq PK_\ell$ must be satisfied for all the different entries $j \neq \ell$ of the table; if this is not the case, the process is repeated for one of these users.

Since we are assuming that H_1 behaves as a random function, and the values PK_j are all randomly chosen, this step is consistent.

For any possible set of users $\mathcal{U}_i$, we define the value $Y_i = \sum_{U_j \in \mathcal{U}_i} PK_j$. Because of the way in which we have computed the values PK_j, we have that

$$Y_i = \gamma_i P + \delta_i(bP)$$

for some values $\gamma_i, \delta_i \in \mathbb{Z}_q$ that the machine $\mathcal{F}$ knows.

When $\mathcal{A}$ asks for the secret key corresponding to an identity ID_i, the machine $\mathcal{F}$ looks for ID_i in the table TAB_{H_1}. If $c_i = 0$, then $\mathcal{F}$ sends $SK_i = x_i Y$ to $\mathcal{A}$. If $c_i = 1$, the machine $\mathcal{F}$ cannot answer and halts. Note that the probability that $\mathcal{F}$ halts in this process is less than $1 - \mu^{Q_e} \leq \frac{\varepsilon}{12}$.

Every time $\mathcal{A}$ makes a query to the random oracle H_2, the machine $\mathcal{F}$ queries the same input to this random oracle H_2 (because it is allowed to do this), and sends the obtained answer to $\mathcal{A}$.

The adversary $\mathcal{A}$ is allowed to query for Q_s valid ring signatures for messages and access structures of its choice. The machine $\mathcal{F}$ must simulate the information that $\mathcal{A}$ would obtain from these execution of the signing algorithm. Let B be the set of the users for whom $\mathcal{A}$ has asked for their secret keys (we call them corrupted users). When $\mathcal{A}$ asks for a valid signature for a message m' and an access structure $\mathcal{U}' = \{\mathcal{U}'_1, \ldots, \mathcal{U}'_d\}$, the machine $\mathcal{F}$ chooses at random one of the sets of $\mathcal{U}'$ to be the "real" author of the ring signature; for simplicity, we denote this set as $\mathcal{U}'_s = \{U'_1, U'_2, \ldots, U'_{n_s}\}$. The information that $\mathcal{A}$ would obtain from such a real computation consists of all the information broadcast in the private broadcast channel of $\mathcal{U}'_s$ (because we can consider the worst case where some of the users in $\mathcal{U}'_s$ is corrupted, and so $\mathcal{A}$ has access to this channel), as well as the secret information generated by the corrupted players, in $B \cap \mathcal{U}'_s$. The machine $\mathcal{F}$ must execute the following algorithm in order to simulate this information:

1. For each user $U'_\ell \in \mathcal{U}'_s \cap B$, choose at random $a_{s\ell} \in \mathbb{Z}_q^*$, compute and broadcast $R'_{s\ell} = e(a_{s\ell}P, P)$.

2. Choose, for all $i = 1, \ldots, d$, $i \neq s$, random values $a_i \in \mathbb{Z}_q^*$, pairwise different, and compute $R'_i = e(a_i P, P)$ and $h_i = H_2(\mathcal{U}', m', R_i)$ (by querying the random oracle H_2); we can assume that $\mathcal{A}$ will later ask the random oracle H_2 with these inputs, to verify the correctness of the signature.

3. Choose at random $h'_s \in \mathbb{Z}_q$.

4. For user U'_1:

   - if $U'_1 \in B$ (since $\mathcal{F}$ has not halted, this means that the machine $\mathcal{F}$ knows the secret key SK_1 of this corrupted user, as well as the value a_{s1}), compute $\sigma_1 = a_{s1}P + h'_s SK_1 + \sum_{1 \le i \le d, i \ne s} a_i P$;
   - if $U'_1 \notin B$, choose at random $\sigma_1 \in \mathbb{G}_1$ and compute

$$R'_{s1} = e(\sigma_1, P) \cdot e(h'_s PK_1, -Y) \cdot \prod_{1 \le i \le d, i \ne s} (R'_i)^{-1}.$$

5. For user U'_j, for $j = 2, \ldots, n_s$:

   - if $U'_j \in B$ (since $\mathcal{F}$ has not halted, this means that the machine $\mathcal{F}$ knows the secret key SK_j of this corrupted user, as well as the value a_{sj}), compute $\sigma_j = a_{sj}P + h'_s SK_j + \sigma_{j-1}$;
   - if $U'_j \notin B$, choose at random $\sigma_j \in \mathbb{G}_1$ and compute

$$R'_{sj} = e(\sigma_j - \sigma_{j-1}, P) \cdot e(h'_s PK_j, -Y).$$

6. Compute the value

$$R'_s = e(-Y, \sum_{1 \le i \le d, i \ne s} h'_i Y_i) \prod_{U_j \in \mathcal{U}_s} R'_{sj}.$$

   If $R'_s = 1$ or $R'_s = R'_i$ for some $i = 1, \ldots, d$, $i \ne s$, then return to step 1.

7. Impose the relation $H_2(\mathcal{U}', m', R'_s) = h'_s$. Later, if $\mathcal{A}$ asks the random oracle H_2 for this input, then $\mathcal{F}$ will answer with h'_s. Since h'_s is a random value and we are in the random oracle model for H_2, this relation is consistent for $\mathcal{A}$.

The resulting signature $(\mathcal{U}', m', R'_1, \ldots, R'_d, h'_1, \ldots, h'_d, \sigma')$ is valid. However, the assignment $H_2(\mathcal{U}', m', R'_s) = h'_s$, in step 7 of the simulating algorithm, can cause some collision if the query $(\mathcal{U}', m', R'_s)$ has been previously made to the random oracle H_2, or if the same tuple $(\mathcal{U}', m', R'_s)$ is produced two times in two different runs of the signature simulation algorithm.

Since no R'_i appears with probability greater than $2/2^k$ in a simulated ring signature, we can bound the probability that such collisions occur:

- The probability that a tuple $(\mathcal{U}', m', R'_s)$ that $\mathcal{F}$ outputs, as part of a simulated ring signature, has been asked before to the random oracle by $\mathcal{A}$ is less than $Q_2 \cdot Q_s \cdot \frac{2}{2^k} \le \frac{\varepsilon}{6}$.
- The probability that the same tuple $(\mathcal{U}', m', R'_s)$ is output two times by $\mathcal{F}$ in two different signature simulations is less than $\frac{Q_s^2}{2} \cdot \frac{2}{2^k} \le \frac{\varepsilon}{6}$.

Altogether, the probability of collisions is less than $\varepsilon/3$. The probability that the machine $\mathcal{F}$ succeeds in obtaining a valid ring signature is the following:

$$\tilde{\varepsilon}_{\mathcal{F}} = \Pr[\mathcal{F} \text{ obtains a valid distributed ring signature}] =$$

$$\Pr[\mathcal{F} \text{ does not halt AND no-collisions in the simulations AND } \mathcal{A} \text{ succeeds}] \geq$$

$$\geq \Pr[\mathcal{A} \text{ succeeds} \mid \mathcal{F} \text{ does not halt AND no-collisions in the simulations }] -$$

$$- \Pr[\mathcal{F} \text{ halts OR collisions in the simulations}] \geq \varepsilon - \left(\frac{\varepsilon}{12} + \frac{\varepsilon}{3}\right) = \frac{7\varepsilon}{12} .$$

However, assuming that $\mathcal{A}$ provides $\mathcal{F}$ with a valid distributed ring signature for a pair $(m,\mathcal{U})$, where $\mathcal{U} = \{\mathcal{U}_1, \ldots, \mathcal{U}_d\}$ has $d \leq \hat{d}$ subsets, we need to be sure that $\mathcal{F}$ does not know any of the d "secret keys" in $\mathcal{U}$. In this case, the "secret key" of a subset $\mathcal{U}_i$, matching with the "public key" $PK_i = Y_i = \sum_{U_j \in \mathcal{U}_i} PK_j$, is $SK_i = \sum_{U_j \in \mathcal{U}_i} SK_j$. Otherwise, if $\mathcal{F}$ knows some of this secret keys, it could have generated this forged signature by itself, and then it would not be a real forgery.

$\mathcal{F}$ will know SK_i if and only if he knows the secret keys of all the members of $\mathcal{U}_i$, or in other words, if $c_j = 0$, for all $U_j \in \mathcal{U}_i$. Therefore, the probability that $\mathcal{F}$ does not know any of the d "secret keys" in $\mathcal{U}$ is

$$\Pr[\forall i = 1, \ldots, d, \ \exists U_j \in \mathcal{U}_i \text{ s.t. } c_j = 1] \geq (1-\mu)^d.$$

Summing up, with probability $\varepsilon_{\mathcal{F}} = (1-\mu)^d \ \tilde{\varepsilon}_{\mathcal{F}} \geq (1-\mu)^{\hat{d}} \ \frac{7\varepsilon}{12} \geq \frac{7V_{Q_2,\hat{d}}}{2^k}$, the machine $\mathcal{F}$ obtains a valid forged ring signature for an access structure where he does not know any "secret key". The execution time of the machine $\mathcal{F}$ is $T_{\mathcal{F}} \leq T + Q_1 + Q_2 + (\hat{d} + \hat{n})Q_s$.

Applying the Ring Forking Lemma (Theorem 1) to the machine $\mathcal{F}$, we have that, by executing two times $\mathcal{F}$, we will obtain in time $T' \leq 2T_{\mathcal{F}}$ and with probability $\tilde{\varepsilon}' \geq \frac{\varepsilon_{\mathcal{F}}^2}{66V_{Q_2,\hat{d}}}$ two valid ring signatures $(\mathcal{U}, m, R_1, \ldots, R_d, h_1, \ldots, h_d, \sigma)$ and $(\mathcal{U}, m, R_1, \ldots, R_d, h'_1, \ldots, h'_d, \sigma')$ such that $h_j \neq h'_j$, for some $j \in \{1, \ldots, d\}$ and $h_i = h'_i$ for all $i = 1, \ldots, d$ such that $i \neq j$.

By definition of valid forgery against a distributed ring signature scheme, there exists at least one non-corrupted user in each subset $\mathcal{U}_i \in \mathcal{U}$; in particular there exists a non-corrupted user $U_z \in \mathcal{U}_j \backslash B$ in the subset $\mathcal{U}_j$. Remember that $Y_j = \gamma_j P + \delta_j(bP)$, where γ_j and δ_j are values known by the machine $\mathcal{F}$.

For this non-corrupted user $U_z \in \mathcal{U}_j$, we have $c_z = 1$ with probability $1-\mu$, which means that $PK_z = \alpha_z(bP)$. So the value α_z is one of the terms added in the factor δ_j that appears in Y_j. If this is the case, then with overwhelming probability we will have that $\delta_j \neq 0 \bmod q$.

If now we come back to the two forged signatures, and we write the corresponding verification equations, we have:

$$e(\sigma, P) = R_1 \cdot \ldots \cdot R_d \cdot e(Y, h_1 Y_1) \cdot \ldots \cdot e(Y, h_d Y_d)$$

$$e(\sigma', P) = R_1 \cdot \ldots \cdot R_d \cdot e(Y, h'_1 Y_1) \cdot \ldots \cdot e(Y, h'_d Y_d)$$

Dividing these two equations, we obtain $e(\sigma - \sigma', P) = e(Y, (h_j - h'_j)Y_j) = e(aP, (h_j - h'_j)(\gamma_j P + \delta_j(bP))) = e(aP, (h_j - h'_j)\gamma_j P) \cdot e(aP, (h_j - h'_j)\delta_j(bP))$.

We can conclude from this relation the equality

$$e(ab\delta_j(h_j - h'_j)P, P) = e(\sigma - \sigma' - [a\gamma_j(h_j - h'_j)]P, P).$$

Since the pairing is non-degenerate, this implies that $ab\delta_j(h_j - h'_j)P = \sigma - \sigma' - [a\gamma_j(h_j - h'_j)]P$. Therefore, one can compute the solution of the given instance of the Computational Diffie-Hellman problem:

$$abP = \frac{1}{\delta_j(h_j - h'_j)}(\sigma - \sigma') - \frac{a\gamma_j}{\delta_j}P \ .$$

The inverses are computed modulo q, and they always exists because $h_j \neq h'_j$ and $\delta_j \neq 0 \bmod q$ with overwhelming probability.

Summing up, the machine $\mathcal{F}$ has solved the Computational Diffie-Hellman problem with probability

$$\varepsilon' = (1-\mu)\hat{\varepsilon}' \geq (1-\mu)\frac{\varepsilon_{\mathcal{F}}^2}{66V_{Q_2,\hat{n}}} \geq (1-\mu)\frac{((1-\mu)^{\hat{d}}\ 7\varepsilon/12)^2}{66V_{Q_2,\hat{d}}} \geq \frac{(1-\mu)^{2\hat{d}+1}}{200V_{Q_2,\hat{d}}}\ \varepsilon^2.$$

And the total time needed to solve the problem has been $T' \leq 2T_{\mathcal{F}} \leq 2T + 2Q_1 + 2Q_2 + 2(\hat{d} + \hat{n})Q_s$.

□

5 An ID-Based Distributed Ring Signature Scheme for Threshold Access Structures

We next propose a different scheme for computing threshold ring signatures in a more efficient way, in an ID-based scenario. The proposal follows the ideas introduced in [19], where threshold ring signatures are designed for PKI scenarios (with users having either Disc-Log or RSA keys, for example).

In the design of the new scheme, Shamir's threshold secret sharing scheme [16] is used as a primitive. We will assume, again, that any specific set of users can always have access to a private and authenticated broadcast channel. The protocols of our proposed scheme are described below.

Key generation: let $\mathbb{G}_1$ be an additive group of prime order q, generated by some element P. Let $\mathbb{G}_2$ be a multiplicative group with the same order q. We need $q \geq 2^k$, where k is the security parameter of the scheme. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear pairing as defined in Section 2.1. Let $H_1 : \{0,1\}^* \to \mathbb{G}_1^*$ and $H_2 : \{0,1\}^* \to \mathbb{Z}_q$ be two hash functions.

The master entity chooses at random his secret key $x \in \mathbb{Z}_q^*$ and publishes the value $Y = xP$.

Secret key extraction: any user U_i of the system, with identity ID_i, has public key $PK_i = H_1(ID_i)$. When he requests the master for his matching secret key, he obtains the value $SK_i = xPK_i$.

Threshold ring signature generation: assume that a subset of users $\{U_1, U_2, \dots, U_t\}$ want to compute an anonymous signature on behalf of a set $\mathcal{U} = \{U_1, \dots, U_t, U_{t+1}, \dots, U_\ell\}$, where $1 \leq t \leq \ell$. The t signing users jointly execute the following protocol:

1. For non-signing users $U_i \in \mathcal{U}$, with $i = t+1, \dots, \ell$, they choose uniformly at random $c_i \in \mathbb{Z}_q$ and $A_i \in \mathbb{G}_1$; they compute and broadcast the value
$$z_i = e(A_i, P) \cdot e(Y, c_i PK_i).$$
2. The signing users U_j, with $j = 1, \dots, t$, choose uniformly at random $T_j \in \mathbb{G}_1$; they compute and broadcast the value
$$z_j = e(T_j, P).$$
3. They compute $c = H_2(\mathcal{U}, m, z_1, \dots, z_\ell)$.
4. They construct, by using Lagrange interpolation, the only polynomial $f(x) \in \mathbb{Z}_q[X]$ of degree $\ell - t$ which verifies $f(0) = c$ and $f(i) = c_i$, for $i = t+1, \dots, \ell$.
5. For $j = 1, \dots, t$, player U_j computes $c_j = f(j)$ and then computes and broadcasts the value
$$A_j = T_j - c_j SK_j.$$
6. The resulting signature is $(\mathcal{U}, m, f(x), A_1, \dots, A_\ell)$.

Verification of a threshold ring signature: the recipient of the message first verifies that the degree of $f(x)$ is exactly $\ell - t$. Then he computes $c_i = f(i)$, for every user $U_i \in \mathcal{U}$, with $i = 1, \dots, \ell$, and the values
$$z_i = e(A_i, P) \cdot e(Y, c_i PK_i).$$
The signature is valid if $f(0) = H_2(\mathcal{U}, m, z_1, \dots, z_\ell)$.

5.1 Correctness and Anonymity of the Scheme

A signature which has been generated following the above method is correct, because $z_i = e(A_i, P) \cdot e(Y, c_i PK_i)$ for $i = t+1, \dots, \ell$, by construction. On the other hand, for $j = 1, \dots, t$, we have that
$$z_j = e(T_j, P) = e(A_j + c_j SK_j, P) = e(A_j, P) \cdot e(c_j x PK_j, P) = e(A_j, P) \cdot e(c_j PK_j, Y),$$
as desired. Therefore, the signature satisfies that $c = f(0) = H_2(\mathcal{U}, m, z_1, \dots, z_\ell)$.

With respect to anonymity, the reasoning is similar to the one that we have already used in Section 4.2: given a valid threshold ring signature $(\mathcal{U}, m, f(x), A_1, \dots, A_\ell)$

on behalf of a set of users $\mathcal{U}$, the probability that a particular subset $B \subset \mathcal{U}$ of t users have computed this signature is exactly

$$\frac{1}{q^{2(\ell-t)}} \cdot \frac{1}{q^t} = \frac{1}{q^{2\ell-t}}.$$

This probability depends only on ℓ and t. Therefore, all the subsets of $\mathcal{U}$ with t users have the same probability to be the actual authors of the signature.

5.2 Unforgeability of the Scheme

In the particular case of threshold access structures, the definition of a $(T, \varepsilon, Q_1, Q_2, Q_e, Q_s)$-successful attacker against a ID-based threshold ring signature scheme is the following: it receives as input the identities of a set of users, then it runs in time T, makes Q_1 queries to the random oracle H_1, Q_2 queries to the random oracle H_2, asks for Q_e secret keys of different users and asks for Q_s valid threshold ring signatures. With probability ε, this algorithm obtains a valid new signature for a pair $(\mathcal{U}, m)$ and a threshold t, such that it has asked for the secret key of at most $t-1$ of the users in $\mathcal{U}$.

In the following theorem, we prove the unforgeability of our ID-based threshold ring signature scheme, by reducing the problem of forging a signature to the Computational Diffie-Hellman problem.

Theorem 3. *Let $\mathcal{A}$ be a $(T, \varepsilon, Q_1, Q_2, Q_e, Q_s)$-successful adversary against the proposed ID-based threshold ring signature scheme, such that the success probability ε of $\mathcal{A}$ is non-negligible in the security parameter k.*

We denote by $\hat{\ell}$ the maximum cardinality of the sets for which $\mathcal{A}$ asks for a valid signature.

Let μ be any value satisfying $\left(1 - \frac{\varepsilon}{6}\right)^{1/Q_e} \leq \mu < 1$.

Then the Computational Diffie-Hellman problem in $\mathbb{G}_1$ can be solved in time $T' \leq 2T + 2Q_1 + 2Q_2 + 2\hat{\ell}Q_s$ and with probability $\varepsilon' \geq \frac{(1-\mu)\varepsilon^2}{128 Q_2}$.

Proof. The first thing to remark is the fact that we can bound $\varepsilon \geq \frac{3(Q_s+Q_2)^2}{2^k}$. Otherwise, the success probability ε would be negligible in the security parameter k.

We are going to construct a probabilistic polynomial time Turing machine $\mathcal{F}$ which will use the attacker $\mathcal{A}$ as a sub-routine in order to solve the given instance of the Computational Diffie-Hellman problem. Therefore, $\mathcal{F}$ must perfectly simulate the environment of the attacker $\mathcal{A}$.

The machine $\mathcal{F}$ receives a list of identities and the public data (P, aP, bP), and its goal is to compute the value abP. The public key of the master entity is defined to be $Y = aP$. Then $\mathcal{F}$ runs the attacker $\mathcal{A}$ against the threshold ID-based ring signature scheme, answering to all the queries that $\mathcal{A}$ makes. The public key $Y = aP$ is also sent to the attacker $\mathcal{A}$.

Without loss of generality, we can assume that $\mathcal{A}$ asks the random oracle H_1 for the value $H_1(ID)$ before asking for the secret key of ID.

The machine $\mathcal{F}$ constructs a table TAB_{H_1} to simulate the random oracle H_1. Every time an identity ID_i is asked by $\mathcal{A}$ to the oracle H_1, the machine $\mathcal{F}$ acts as follows: first $\mathcal{F}$ checks if this input is already in the table; if this is the case, then $\mathcal{F}$ sends to $\mathcal{A}$ the corresponding relation $H_1(ID_i) = PK_i$. Otherwise, with probability μ, the machine $\mathcal{F}$ chooses the bit $d_i = 0$ and a different $x_i \in \mathbb{Z}_q^*$ at random, and defines $PK_i = x_i P$ and $SK_i = x_i Y$. On the other hand, with probability $1 - \mu$, the machine $\mathcal{F}$ chooses the bit $d_i = 1$ and a different $\alpha_i \in \mathbb{Z}_q^*$ at random, and defines $PK_i = (\alpha_i)bP$ and $SK_i = \perp$. The values $(ID_i, PK_i, x_i$ or $\alpha_i, SK_i, d_i)$ are stored in a new entry of TAB_{H_1}, and the relation $H_1(ID_i) = PK_i$ is sent to $\mathcal{A}$. The condition $PK_i \neq PK_j$ must be satisfied for all the different entries $i \neq j$ of the table; if this is not the case, the process is repeated for one of these users.

Since we are assuming that H_1 behaves as a random function, and the values PK_i are all randomly chosen, this simulation of the hash function H_1 is consistent.

Later, every time $\mathcal{A}$ asks for the secret key corresponding to an identity ID_i, the machine $\mathcal{F}$ looks for ID_i in the table TAB_{H_1}. If $d_i = 0$, then $\mathcal{F}$ sends $SK_i = x_i Y$ to $\mathcal{A}$. If $d_i = 1$, the machine $\mathcal{F}$ cannot answer and halts. The probability that $\mathcal{F}$ halts in this process is less than $1 - \mu^{Q_e} \leq \varepsilon/6$.

As well, $\mathcal{F}$ constructs a table TAB_{H_2} to simulate the random oracle H_2. Every time $\mathcal{A}$ makes a query to this oracle, $\mathcal{F}$ looks for this value in the table. If it is already there, then $\mathcal{F}$ sends the corresponding relation to $\mathcal{A}$; if not, $\mathcal{F}$ chooses at random an output of the random oracle for the queried input, different from the outputs which are already in the table, sends the relation to $\mathcal{A}$ and stores it in the table TAB_{H_2}.

Finally, the attacker $\mathcal{A}$ can ask Q_s times for valid threshold ring signatures for messages m', sets $\mathcal{U}'$ of ℓ' users and thresholds t'. To answer such queries, the machine $\mathcal{F}$ proceeds as follows:

1. Choose at random $\ell' - t' + 1$ values $c', c'_{t'+1}, \ldots, c'_{\ell'} \in \mathbb{Z}_q$.
2. Using Lagrange interpolation, construct the only polynomial $f'(x) \in \mathbb{Z}_q[X]$ with degree $\ell' - t'$ such that $f'(0) = c'$ and $f'(i) = c'_i$, for $i = t' + 1, \ldots, \ell'$.
3. Compute the values $c'_j = f'(j)$, for $j = 1, \ldots, t'$.
4. Choose at random ℓ' values $A'_1, \ldots, A'_{\ell'} \in \mathbb{G}_1$.
5. Compute, for $i = 1, \ldots, \ell'$, the values $z'_i = e(A'_i, P) \cdot e(Y, c'_i PK_i)$..
6. Impose and store in the table TAB_{H_2} the new relation $H_2(\mathcal{U}', m', z'_1, \ldots, z'_\ell) = c'$.
7. Define the signature to be $(\mathcal{U}', m', f'(x), A'_1, \ldots, A'_\ell)$.

The process results in a valid threshold ring signature, because we are assuming that H_2 behaves as a random function, and c' is taken uniformly at random in $\mathbb{Z}_q$. However, the assignment $H_2(\mathcal{U}', m', z'_1, \ldots, z'_\ell) = c'$ can produce some collisions in the management of the table TAB_{H_2} that simulates the random oracle H_2.

A first possible collision occurs if a tuple $(\mathcal{U}', m', z'_1, \ldots, z'_\ell)$ produced in the simulation of a signature has been already queried to the random oracle H_2. The probability of this event is less than $\frac{Q_s Q_2}{q}$.

A second possible collision occurs when the same tuple $(\mathcal{U}', m', z'_1, \ldots, z'_\ell)$ is produced in two different signature simulations. The probability of this event is less than $\frac{Q_s^2}{2q}$.

We denote by ω the whole set of random tapes that take part in an attack by $\mathcal{A}$, with the environment simulated by $\mathcal{F}$, but excluding the randomness related to the oracle H_2. The success probability of $\mathcal{A}$ in forging a valid ring signature scheme is then taken over the space (ω, H_2).

In an execution of the attacker $\mathcal{A}$, we use the notation $\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_{Q_2}$ for the different queries that $\mathcal{A}$ makes to the random oracle H_2. If $\mathcal{A}$ produces a valid forged signature $(\mathcal{U}, m, f(x), A_1, \ldots, A_\ell)$, by the ideal randomness of the oracle H_2, the probability that $\mathcal{A}$ has not asked for the tuple $(\mathcal{U}, m, z_1, \ldots, z_\ell)$ to this oracle (and so $\mathcal{A}$ must have guessed the corresponding output), is less than $\frac{1}{q}$. We define $\beta = \infty$ in this case; otherwise, β denotes the index of the query where the tuple above was asked. That is, $\mathcal{Q}_\beta = (\mathcal{U}, m, z_1, \ldots, z_\ell)$.

We denote by $\mathcal{S}$ the set of successful executions of $\mathcal{A}$, with $\mathcal{F}$ simulating its environment, and such that $\beta \neq \infty$. We also define the following subsets of $\mathcal{S}$: for every $i = 1, 2, \ldots, Q_2$, the set $\mathcal{S}_i$ contains the successful executions such that $\beta = i$.

This gives us a partition $\{\mathcal{S}_i\}_{i=1,\ldots,Q_2}$ of $\mathcal{S}$ in exactly Q_2 classes.

The probability that an execution (ω, H_2) of $\mathcal{A}$ with the environment simulated by $\mathcal{F}$ results in a valid forgery with $\beta \neq \infty$ is

$$\tilde{\varepsilon} = \Pr[(\omega, H_2) \in \mathcal{S}] \geq \varepsilon - (1 - \mu^{Q_e}) - \frac{Q_s Q_2}{q} - \frac{Q_s^2}{2q} - \frac{1}{q} \geq$$

$$\geq \varepsilon - \frac{\varepsilon}{6} - \frac{\varepsilon}{6} - \frac{\varepsilon}{6} = \frac{\varepsilon}{2}.$$

Now we define the set of indexes which are more likely to appear as

$$I = \{i \text{ s.t. } \Pr[(\omega, H_2) \in \mathcal{S}_i \mid (\omega, H_2) \in \mathcal{S}] \geq \frac{1}{2Q_2}\}.$$

And the corresponding subset of successful executions as $\mathcal{S}_I = \{(\omega, H_2) \in \mathcal{S}_i \text{ s.t. } i \in I\}$.

For a specific index $i \in I$, we have that

$$\Pr[(\omega, H_2) \in \mathcal{S}_i] = \Pr[(\omega, H_2) \in \mathcal{S}] \cdot \Pr[(\omega, H_2) \in \mathcal{S}_i \mid (\omega, H_2) \in \mathcal{S}] \geq$$

$$\geq \tilde{\varepsilon} \cdot \frac{1}{2Q_2}.$$

Lemma 2. *It holds that* $\Pr[(\omega, H_2) \in \mathcal{S}_I \mid (\omega, H_2) \in \mathcal{S}] \geq 1/2$.

Proof. Since the sets $\mathcal{S}_i$ are disjoint, we have

$$\Pr[(\omega, H_2) \in \mathcal{S}_I \mid (\omega, H_2) \in \mathcal{S}] = \sum_{i \in I} \Pr[(\omega, H_2) \in \mathcal{S}_i \mid (\omega, H_2) \in \mathcal{S}] =$$

$$1 - \sum_{i \notin I} \Pr[(\omega, H_2) \in \mathcal{S}_i \mid (\omega, H_2) \in \mathcal{S}].$$

Since the complement of I contains at most Q_2 indexes, we have that this probability is greater than $1 - Q_2 \cdot \frac{1}{2Q_2} = 1/2$. $\square$

We come back to the execution of $\mathcal{A}$ with the environment simulated by $\mathcal{F}$. With probability at least $\tilde{\varepsilon}$, such an execution (ω, H_2) results in a valid forgery with $\beta \neq \infty$. In this case, applying Lemma 2, we know that this successful execution belongs to $\mathcal{S}_I$ with probability at least $1/2$.

Now we split H_2 as (H_2', c), where H_2' corresponds to the answers of all the queries to H_2 except the query $\mathcal{Q}_\beta$, whose answer is denoted as c.

We apply the Splitting Lemma (lemma 1), taking $X = (\omega, H_2')$, $Y = c$, $A = \mathcal{S}_\beta$, $\delta = \frac{\tilde{\varepsilon}}{2Q_2}$ and $\alpha = \frac{\tilde{\varepsilon}}{4Q_2}$. The lemma says that there exists a subset of executions Ω_β such that

$$\Pr[(\omega, H_2) \in \Omega_\beta \mid (\omega, H_2) \in \mathcal{S}_\beta] \geq \frac{\alpha}{\delta} = \frac{1}{2}$$

and such that, for any $(\omega, H_2) \in \Omega_\beta$:

$$\Pr_{\tilde{c}}[(\omega, H_2', \tilde{c}) \in \mathcal{S}_\beta] \geq \delta - \alpha = \frac{\tilde{\varepsilon}}{4Q_2}.$$

With probability at least $\frac{\tilde{\varepsilon}}{2}$, the first execution (ω, H_2', c) of $\mathcal{A}$ simulated by $\mathcal{F}$ is successful and the index β belongs to the set I. Furthermore, in this case we have that $(\omega, H_2', c) \in \Omega_\beta$ with probability at least $1/2$. If we now repeat this simulated execution of $\mathcal{A}$ with fixed (ω, H_2') and randomly chosen $\tilde{c} \in \mathbb{Z}_q$, we know that $(\omega, H_2', \tilde{c}) \in \mathcal{S}_\beta$ and furthermore $\tilde{c} \neq c$ with probability at least $\frac{\tilde{\varepsilon}}{4Q_2} - \frac{1}{q}$.

Now consider the two successful executions of the attack, (ω, H_2', c) and $(\omega, H_2, \tilde{c})$, that the algorithm $\mathcal{F}$ has obtained by executing the attack $\mathcal{A}$. We denote by $(\mathcal{U}, m, f(x), A_1, \ldots, A_\ell)$ and $(\tilde{\mathcal{U}}, \tilde{m}, \tilde{f}(x), \tilde{A}_1, \ldots, \tilde{A}_\ell)$, respectively, the forged threshold ring signatures. Since the random tapes and H_1 are identical, and the answers of the random oracle H_2 are the same until the query $\mathcal{Q}_\beta = (\mathcal{U}, m, z_1, \ldots, z_\ell)$, we have in particular that $\tilde{\mathcal{U}} = \mathcal{U}$, $\tilde{m} = m$ and $\tilde{z}_i = z_i$, for $i = 1, \ldots, \ell$.

Since $f(0) = c \neq \tilde{c} = \tilde{f}(0)$ and the degree of both $f(x)$ and $\tilde{f}(x)$ is $\ell - t$, the two polynomials $f(x)$ and $\tilde{f}(x)$ can coincide at most at $\ell - t$ points. Therefore, there are at least t values $j_1, \ldots, j_t \in \{1, \ldots, \ell\}$ such that $f(j_i) \neq \tilde{f}(j_i)$, for $i = 1, \ldots, t$. Furthermore, the forgery against the threshold ring signature scheme has been valid, so the attacker $\mathcal{A}$ has asked for the secret key of at most $t - 1$ members of the signing ring $\mathcal{U}$. This means that there is at least one member $U_j \in \mathcal{U}$ such that $c_j = f(j) \neq \tilde{f}(j) = \tilde{c}_j$ and such that the secret key of U_j has not been asked by $\mathcal{A}$. In this case, with probability $1 - \mu$ we have $d_j = 1$ and so $PK_j = \alpha_j bP$.

The equality $\tilde{z}_j = z_j$ becomes $e(A_j, P) \cdot e(Y, c_j PK_j) = e(\tilde{A}_j, P) \cdot e(Y, \tilde{c}_j PK_j)$. This is equivalent to

$$e(A_j - \tilde{A}_j, P) = e(Y, (\tilde{c}_j - c_j)PK_j) = e(aP, (\tilde{c}_j - c_j)\alpha_j bP) = e(a(\tilde{c}_j - c_j)\alpha_j bP, P).$$

This implies that $A_j - \tilde{A}_j = a(\tilde{c}_j - c_j)\alpha_j bP$. Therefore, the machine $\mathcal{F}$ obtains the solution of the given instance of the Computational Diffie-Hellman problem as

$$abP = \frac{1}{(\tilde{c}_j - c_j)\alpha_j}(A_j - \tilde{A}_j).$$

The inverse can be taken modulo q, since $\alpha_j \in \mathbb{Z}_q^*$ and $c_j \neq \tilde{c}_j$.

The total success probability ε' of the attack performed by $\mathcal{F}$ is

$$\varepsilon' \geq (1-\mu)\frac{\tilde{\varepsilon}}{2} \cdot \frac{1}{2}\left(\frac{\tilde{\varepsilon}}{4Q_2} - \frac{1}{q}\right) \geq (1-\mu)\frac{\tilde{\varepsilon}}{4} \cdot \frac{\tilde{\varepsilon}}{8Q_2} \geq$$
$$\geq \frac{(1-\mu)\tilde{\varepsilon}^2}{32Q_2} \geq \frac{(1-\mu)(\varepsilon/2)^2}{32Q_2} \geq \frac{(1-\mu)\varepsilon^2}{128Q_2}.$$

The total execution time T' of the machine $\mathcal{F}$ consists of running two times the machine $\mathcal{A}$, simulating its environment. That is, $T' \leq 2(T + Q_1 + Q_2 + \hat{\ell} Q_s)$.

$\square$

This last proposal, apart from being more efficient for the case of threshold structures, enjoys a better security reduction, since the factor $V_{Q_2,\hat{d}}$ does not appear in the relation between the probabilities ε' and ε. This is due to the fact that the Ring Forking Lemma for generic ring signature schemes is not used in the proof of the security of this threshold proposal.

6 Conclusion

In this work we have dealt with distributed ring signature schemes in identity-based scenarios. Such schemes provide anonymity to a subset of users who want to sign a message on behalf of a larger set of users. Furthermore, in identity-based scenarios, public keys of the users are derived from publicly verifiable data (for example, an e-mail address), and so digital certificates are not necessary to authenticate the validity of public keys. This allows more efficient implementations of public key cryptographic systems, specially for those cases where basic operations involve many different public keys, as it happens in (distributed) ring signatures.

We have proposed the two first distributed ring signature schemes which run in an identity-based framework. The first one can be used for general families of possible signing subsets, whereas the second one is specific, and more efficient, for the case of threshold families. The design of the schemes uses different mathematical tools, as bilinear pairings or Shamir's secret sharing scheme. In the security analysis, we use some results of probability theory and we assume that the well-known Computational Diffie-Hellman problem is intractable.

References

[1] M. Abe, M. Ohkubo and K. Suzuki, $1-$out$-$of$-n$ signatures from a variety of keys. *Proceedings of Asiacrypt'02*, Springer-Verlag, Lecture Notes in Computer Science 2501: 415–432 (2002).

[2] M. Bellare, A. Boldyreva and A. Palacio, An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. *Proceedings of Eurocrypt'04*, Springer-Verlag, Lecture Notes in Computer Science 3027: 171–188 (2004).

[3] M. Bellare and P. Rogaway, Random oracles are practical: a paradigm for designing efficient protocols. *Proceedings of 1st Conference on Computer and Communications Security*, ACM: 62–73 (1993).

[4] E. Bresson, J. Stern and M. Szydlo, Threshold ring signatures for ad-hoc groups. *Proceedings of Crypt0'02*, Springer-Verlag, Lecture Notes in Computer Science 2442: 465–480 (2002).

[5] R. Canetti, O. Goldreich and S. Halevi, The random oracle methodology, revisited. *Proceedings of STOC'98*, ACM: 209–218 (1998).

[6] Y. Dodis, A. Kiayias, A. Nicolosi and V. Shoup, Annonymous identification in ad hoc groups. *Proceedings of Eurocrypt'04*, Springer-Verlag, Lecture Notes in Computer Science 3027: 609–626 (2004).

[7] A. Fiat and M. Naor, Broadcast encryption. *Proceedings of Crypto'93*, Springer-Verlag, Lecture Notes in Computer Science 773: 480–491 (1993).

[8] R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin, Robust Threshold DSS signatures. *Proceedings of Eurocrypt'96*, Springer-Verlag, Lecture Notes in Computer Science 1070: 354–371 (1996).

[9] S. Goldwasser, S. Micali and R. Rivest, A digital signature scheme secure against adaptative chosen-message attacks. *SIAM Journal of Computing*, 17 (2): 281–308 (1988).

[10] J. Herranz and G. Sáez, Forking lemmas for ring signature schemes. *Proceedings of Indocrypt'03*, Springer-Verlag, Lecture Notes in Computer Science 2904: 266–279 (2003).

[11] J. Herranz and G. Sáez, Ring signature schemes for general access structures. *Proceedings of ESAS'04*, Springer-Verlag, Lecture Notes in Computer Science (to appear, 2004).

[12] J. Herranz and G. Sáez, New ID-based ring signature schemes. *Proceedings of ICICS'04*, Springer-Verlag, Lecture Notes in Computer Science (to appear, 2004).

[13] J.B. Nielsen, Separating random oracle proofs from complexity theoretic proofs: the non-committing encryption case. *Proceedings of Crypto'02*, Springer-Verlag, Lecture Notes in Computer Science 2442: 111–126 (2002).

[14] D. Pointcheval and J. Stern, Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13 (3): 361–396 (2000).

[15] R. Rivest, A. Shamir and Y. Tauman, How to leak a secret. *Proceedings of Asiacrypt'01*, Springer-Verlag, Lecture Notes in Computer Science 2248: 552–565 (2002).

[16] A. Shamir, How to share a secret. *Communications of the ACM*, 22: 612–613 (1979).

[17] A. Shamir, Identity-based cryptosystems and signature schemes. *Proceedings of Crypto'84*, Springer-Verlag, Lecture Notes in Computer Science 196: 47–53 (1984).

[18] V. Shoup, Practical threshold signatures. *Proceedings of Eurocrypt'00*, Springer-Verlag, Lecture Notes in Computer Science 1807: 207–220 (2000).

[19] J.K. Sui Liu, V.K. Wei and D.S. Wong, A separable threshold ring signature scheme. *Proceedings of ICISC'03*, Springer-Verlag, Lecture Notes in Computer Science 2971: 12–26 (2004).

[20] F. Zhang and K. Kim, ID-based blind signature and ring signature from pairings. *Proceedings of Asiacrypt'02*, Springer-Verlag, Lecture Notes in Computer Science 2501: 533–547 (2002).

[21] The Pairing-Based Crypto Lounge, Web page maintained by Paulo Barreto: `http://planeta.terra.com.br/informatica/paulobarreto/pblounge.html`