Anonymizable Ring Signature Without Pairing

Olivier Blazy, Xavier Bultel, Pascal Lafourcade

Anonymizable Ring Signature Without Pairing *

Olivier Blazy[1], Xavier Bultel[2], and Pascal Lafourcade[2]

[1] Université de Limoges, Xlim, Limoges, France
[2] Université Clermont Auvergne, LIMOS, Clermont-Ferrand, France

Abstract. Ring signature is a well-known cryptographic primitive that allows any user who has a signing key to anonymously sign a message according to a group of users. Some years ago, Hoshino *et al.* propose a new kind of ring signature where anybody can transform a digital signature into an anonymous signature according to a chosen group of users; authors present a pairing-based construction that is secure under the gap Diffie-Hellman assumption in the random oracle model. However this scheme is quite inefficient for large group since the generation of the anonymous signature requires a number of pairing computations that is linear in the size of the group. In this paper, we give a more efficient anonymizable signature scheme without pairing. Our anonymization algorithm requires n exponentiations in a prime order group where n is the group size. Our proposal is secure under the discrete logarithm assumption in the random oracle model, which is a more standard assumption.

1 Introduction

Anonymizable Signature [7] is a kind of ring signature where anybody who has a signature produced by a group member can transform it into an anonymous signature within the group: someone can check that the anonymous signature has been produced by the group member signature but it is not possible to guess who is he. In practice, such a scheme allows a user to delegate to a proxy the task to anonymize a given signature. For example, during the reviews of an academic conference, each reviewer can sign his review before sending it to the program chair. Then the program chair anonymizes the given signature for the program committee and sends the review and the anonymous signature to the author of the paper. Then the author is convinced that the review comes from one of the member of the program committee but do not know who is the reviewer. The reviewer does not need to know the other members of the program committee.

Authors of [7] propose a pairing-based scheme secure under the gap Diffie-Hellman assumption in the random oracle model. In this scheme, the anonymous signature is a proof of knowledge of a valid signature within the group. However, this scheme is quite inefficient for large groups: the anonymization requires a number of pairing computation which is linear on the size of the group. In this paper, we propose GAWP(for *Get Anonymizable signature Without Pairing*),

* This research was conducted with the support of the "Digital Trust" Chair from the University of Auvergne Foundation.

an efficient pairing-free anonymous signature scheme. This scheme is based on the Schnorr signature [9] and uses the same methodology as [7]. Moreover, our scheme is provably secure under the discrete logarithm assumption in the random oracle model.

Related works: Ring signatures have been introduced by Rivest *et al.* in [8]. Such a signature scheme allows a user to sign a message anonymously within a group. Since the user only needs the public keys of all the members of the group and his secret key, this primitive does not require any group manager as in group signatures [2]. More recently, formal security definitions for ring signatures have been proposed [1]. In [7], Hoshino *et al.* define *anonymizable signatures* that extend the concept of ring signatures adding the possibility to transform any signature into an anonymous signature within a chosen group. Authors formally define the security models of this new primitive and propose a secure instantiation based on the BLS signature [3]. This scheme requires pairing and is proven secure in the random oracle model. To the best of our knowledge, it is the only one anonymizable signature scheme of the literature. Finally, *relinkable signatures* [10] are close to anonymous signatures: this primitive allows a proxy who have the *relink key* to change the group of an anonymous signature. However, a signature cannot be anonymized by anybody. Moreover, the signatures are anonymous for everybody in anonymous signature, but they are not anonymous for the proxy in relinkable signatures.

Outline In the next section, we present the cryptographic background required for our work. In Section 3 gives the formal definitions of an anonymizable signature and the corresponding security models. Then we present the scheme GAWP in Section 4 and we analyze its security before concluding in the last section.

2 Background

In this section, we recall some definitions and cryptographic notions.

Definition 1 (Discrete Logarithm). *Let $\mathbb{G}$ be a multiplicative group of prime order p and $g \in \mathbb{G}$ be a generator. The* discrete logarithm problem *(DL) is to compute x given (g, g^x). The* discrete Logarithm hypothesis *states that there exists no polynomial time algorithm that solves DL with non-negligible advantage.*

Zero-knowledge proofs [6] A *proof of knowledge* is a two-party protocol between two polynomial time algorithms P (the *prover*) and V (the *verifier*). It allows the prover P to convince the verifier V that he knows a solution s to the instance $\mathcal{I}$ of a problem $\mathcal{P}$. Such a protocol is said *zero-knowledge proof of knowledge* (ZKP) if it satisfies the following properties:

Completeness: If P knows s, then he is able to convince V (*i.e.*, V outputs `"accept"`).

Soundness: If P does not know s, then he is not able to convince V (*i.e.*, V outputs `"reject"`) except with negligible probability.

Zero-knowledge: V learns *nothing* about s except $\mathcal{I}$.
Honest-verifier ZKP (HZKP) is a weaker notion of ZKP which is restricted to case where the verifier is *honest*, *i.e.*, V correctly runs the protocol.

If we only have one flow from the prover to the verifier, we say that the ZKP is *non-interactive* (NIZKP). In the litterature, *sigma protocols* are ZKP with three exchanges between the prover and the verifier: a commitment, a challenge, and a response. By example, the Schnorr protocol [9] is a signma protocol that allows to prove the knowledge of the discrete logarithm of an instance $(g, k = g^x)$: the prover chooses $r \xleftarrow{\$} \mathbb{Z}_p^*$ and sends $R = g^r$. Then the verifier sends the challenge $c \xleftarrow{\$} \mathbb{Z}_p^*$ to the prover, who responds with the value $z = r + x \cdot c$. The verifier accepts the proof iff $g^z = h \cdot k^c$.

If the challenge is chosen on a large set, it is possible to transform a sigma protocol into a NIZKP using the Fiat-Shamir heuristic [5] replacing the challenge by the digest of a hash function on the commitment. It is also possible to transform such a NIZKP into a signature scheme in the random oracle model by using the message together with the commitment as input in the hash function to compute the challenge. For example, the Schnorr signature is obtained using this transformation on the Schnorr protocol: to sign a message m with the secret key $x \in \mathbb{Z}_p^*$, the signer picks r and computes $h = g^z$ and $z = r + x \cdot H(h, m)$. Using the public key $k = g^x$ and the signature (h, z), anybody can check that $g^z = h \cdot k^{H(h,m)}$ to validate the signature on m.

Finally, our scheme uses the generic transformation of ZKP designed by [4]. The authors propose a generic transformation from the ZKP of the solution to some problem instance to a ZKP of the solution to one problem instance out of n problem instances (without revealing this problem instance). This transformation holds with any sigma protocol and works as follows: consider n instances $\{I_i\}_{1 \leq i \leq n}$ and a prover who only knows the solution s_1 of the instance I_1. The prover sends n commitment h_i for $1 \leq i \leq n$ and receives an unique challenge c. For all the instances $\{I_i\}_{2 \leq i \leq n}$, the prover chooses a challenge c_i such that he is able to prove the knowledge of the solution of I_i using h_i and the challenge c_i as in the original sigma protocol (note that since he chooses the challenge by himself, he does not need to really know the corresponding secret). Finally, he computes $c_1 = c \oplus c_2 \oplus \ldots \oplus c_n$ and proves the knowledge of I_1 using h_1 and the challenge c_1 as in the original sigma protocol (note that, in this case, the prover must to know the secret s_1 to conclude the proof). Then the verifier check the proof for all pairs (h_i, c_i) and checks that $c = c_1 \oplus \ldots \oplus c_n$. The computational and space cost of the resulting ZKP is n times the cost of the primary ZKP. It is possible to use the Fiat-Shamir transformation on such a ZKP to obtain an equivalent NIZKP.

3 Security Models

We first give a formal definition of an *Anonymizable Ring Signature*.

Definition 2 (Anonymizable ring signature (ARS)). *An ARS is a tuple of algorithms* (Init, Gen, Sig, Ver, Ano, AVer) *such that:*

$\mathsf{Init}(1^t)$: *This algorithm outputs an* init *value from security parameter* t.

$\mathsf{Gen}(\mathsf{init})$: *This algorithm outputs a signing key pair* $(\mathsf{ssk}, \mathsf{svk})$ *from* init.

$\mathsf{Sig}(\mathsf{ssk}, m)$: *This algorithm outputs a signature* σ *on the message* m *using the signing key* ssk.

$\mathsf{Ver}(\mathsf{svk}, \sigma, m)$: *This algorithm returns* 1 *when* σ *is a valid signature of* m *for the verification key* svk. *Else it returns* 0.

$\mathsf{Ano}(L, \sigma, m, \mathsf{svk})$: *This algorithm outputs an anonymous signature* $\widehat{\sigma}$ *on the message* m *according to the set of public key* L *from the signature* σ *and the corresponding verification key* svk.

$\mathsf{AVer}(L, \widehat{\sigma}, m)$: *This algorithm returns* 1 *when* $\widehat{\sigma}$ *is a valid signature of* m *for the set of verification keys* L. *Else it returns* 0.

In what follow, we denote by $\mathtt{out}_{\mathcal{O}}$ the set of all the values outputted by the oracle $\mathcal{O}$ during an experiment. The first security requirement is the *unforgeability*. An ARS is unforgeable when it is not possible to forge a signature without the corresponding secret key, and to forge an anonymous signature without a signature valide for one of the group members. In this model, we give to the attacker the possibility to ask anonymous and regular signatures for chosen messages and chosen users to some signing oracle. Of course, to win the attack, the attacker must forge a signature that does not come from these oracles.

Definition 3 (EUF-CMA security). *Let* P *be an ARS of security parameter* t *and let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a polynomial time adversary. We define the existential unforgeability against chosen message attack experiment for* $\mathcal{A}$ *as follows:*

$\mathbf{Exp}^{\mathsf{euf\text{-}cma}}_{P,\mathcal{A}}(t, n)$:
$\mathsf{init} \leftarrow \mathsf{Init}(1^t)$
$\forall\ i \in \{0, \ldots, n\}, (\mathsf{ssk}_i, \mathsf{svk}_i) \leftarrow \mathsf{Gen}(\mathsf{init})$
$I \leftarrow \mathcal{A}_1^{\mathcal{GO}(\cdot), \mathcal{SO}(\cdot), \mathcal{AO}(\cdot)}(t, \{\mathsf{svk}_i\}_{0 \le i \le n})$
$(L_*, \widehat{\sigma}, m) \leftarrow \mathcal{A}_2^{\mathcal{GO}(\cdot), \mathcal{SO}(\cdot), \mathcal{AO}(\cdot)}(t, \{\mathsf{svk}_i\}_{0 \le i \le n}, \{\mathsf{ssk}_i\}_{i \in I})$
if $((\mathsf{AVer}(L_*, \widehat{\sigma}, m) = 1)$ *and* $(L^* \subset \{\mathsf{svk}_i\}_{0 \le i \le n, i \notin I})$
and $(\forall\ \mathsf{svk} \in L_*, (\mathsf{svk}, m, *) \notin \mathtt{out}_{\mathcal{SO}})$ *and* $((L_*, m, \widehat{\sigma}) \notin \mathtt{out}_{\mathcal{AO}}))$
then output 1 *else output* 0.

Where oracles are defined as follows:

$\mathcal{GO}(\cdot)$ *is a key generation oracle that increments* $n \leftarrow n+1$, *generates* $(\mathsf{ssk}_n, \mathsf{svk}_n) \leftarrow \mathsf{Gen}(\mathsf{init})$ *and returns it.*

$\mathcal{SO}(\cdot)$ *is a signing oracle that takes* (svk_i, m) *as input. It computes* $\sigma \leftarrow \mathsf{Sig}(\mathsf{ssk}_i, m)$ *and returns* $(\mathsf{svk}_i, m, \sigma)$.

$\mathcal{AO}(\cdot)$ *is an anonymization oracle that takes* (svk_i, m, L) *as input. It computes* $\sigma \leftarrow \mathsf{Sig}(\mathsf{ssk}_i, m)$ *and* $\widehat{\sigma} \leftarrow \mathsf{Ano}(L, \sigma, m, \mathsf{svk}_i)$ *and returns* $(L, m, \widehat{\sigma})$.

We define the advantage of the adversary $\mathcal{A}$ *against the EUF-CMA experiment by* $\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{P,\mathcal{A}}(t, n) = Pr[\mathsf{Exp}^{\mathsf{euf\text{-}cma}}_{P,\mathcal{A}}(t, n) = 1]$. *We define the advantage on EUF-CMA experiment by* $\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{P}(t, n) = \max_{\mathcal{A} \in \mathrm{POLY}(t)}\{\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{P,\mathcal{A}}(t, n)\}$. *We say that a ARS scheme* P *is EUF-CMA secure when the advantage* $\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{P}(t, n)$ *is negligible for any polynomially bounded* n.

The second security requirement is the anonymity. Loosely speaking, an ARS is anonymous when it is not possible to guess who has produced the signature

used to compute an anonymous signature. In this security model, the adversary chooses two users and a message m, and it receives an anonymous signature produced from the signatures on m computed by one of the two users included in a bigger set L. The goal is to guess who is the user chosen by the challenger. To help him, the adversary have access to some signing oracles.

Definition 4 (Anonymity). *Let P be an ARS of security parameter t and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be polynomial time adversary. We define the anonymity experiment for adversary $\mathcal{A}$ against P as follows:*

$\mathbf{Exp}^{\mathsf{anon}}_{P,\mathcal{A}}(t,n)$*:*
$b \leftarrow \{0,1\}$
$\mathit{init} \leftarrow \mathit{Init}(1^t)$
$\forall\ i \in \{0,\dots,n\}, (\mathit{ssk}_i, \mathit{svk}_i) \leftarrow \mathit{Gen}(\mathit{init})$
$(i_0, i_1, L, m) \leftarrow \mathcal{A}_1^{\mathcal{GO}(\cdot),\mathcal{SO}(\cdot),\mathcal{AO}(\cdot)}(t, \{(\mathit{ssk}_i, \mathit{svk}_i)\}_{0\le i\le n})$
If i_0 OR $i_1 \notin L$ then Abort
$\widehat{\sigma} \leftarrow \mathit{Ano}(L, \mathit{Sig}(\mathit{ssk}_{i_b}, m), m, \mathit{svk}_{i_b})$
$b' \leftarrow \mathcal{A}_2^{\mathcal{GO}(\cdot),\mathcal{SO}(\cdot),\mathcal{AO}(\cdot)}(t, \widehat{\sigma})$
output $b = b'$.

Where $\mathcal{GO}(\cdot)$, $\mathcal{SO}(\cdot)$ and $\mathcal{AO}(\cdot)$ are defined as in Definition 3.

The advantage of the adversary $\mathcal{A}$ against anonymity is $\mathsf{Adv}^{\mathit{anon}}_{P,\mathcal{A}}(t,n) = \left|Pr[\mathit{Exp}^{\mathit{anon}}_{P,\mathcal{A}}(t,n) = 1] - \frac{1}{2}\right|$. We define the advantage on anonymity experiment by $\mathsf{Adv}^{\mathit{anon}}_{P}(t,n) = \max_{\mathcal{A}\in \mathrm{POLY}(t)}\{\mathsf{Adv}^{\mathit{anon}}_{P,\mathcal{A}}(t,n)\}$ where $\mathrm{POLY}(t)$ is the set of all the algorithm that are polynomial in t. We say that a ARS scheme P is anonymous when the advantage $\mathsf{Adv}^{\mathit{anon}}_{P}(t,n)$ is negligible for any polynomially bounded n.

4 Constructions

In this section, we present our scheme called GAWP (for *Get ARS Without Pairing*). We use the same design methodology as in [7]: to anonymize the signature σ of a message m, a user computes a non-interactive zero knowledge proof of the knowledge of σ according to one verification key out of all the verification keys of the group. Our scheme is based on the well known Schnorr signature (see Section 2). Particularly, the signature and the verification algorithm are the same as in the Schnorr signature: let g be the generator of a prime order group, then the signature algorithm outputs $h = g^r$ and $z = r + \mathsf{ssk} \cdot H(h, m)$ where r is a random value, ssk the signing key and m the message. To validate a signature, a user checks that $g^z = h \cdot \mathsf{svk}^{H(h,m)}$ where svk is the public verification key such that $\mathsf{svk} = g^{\mathsf{ssk}}$. Then, our goal is to give a way to prove the knowledge of a valid Schnorr signature according to one of the verification keys of the group. Note that the first part of the signature $h = g^r$ does not leak any information about the signing key ssk. Then this value can be public in the anonymized signature. The last step is to prove the knowledge of the second part of the signature z according to h, m, H and the set of verification key L of all members of the group. More precisely, our aim is to prove the knowledge of z such that $g^z = h \cdot \mathsf{svk}^{H(h,m)}$ for one $\mathsf{svk} \in L$, h, m and H. In the following, we first give the zero-knowledge proof that allows to prove the knowledge of a Schnorr signature. Next, we give the concrete construction of GAWP, and finally, we analyze its security.

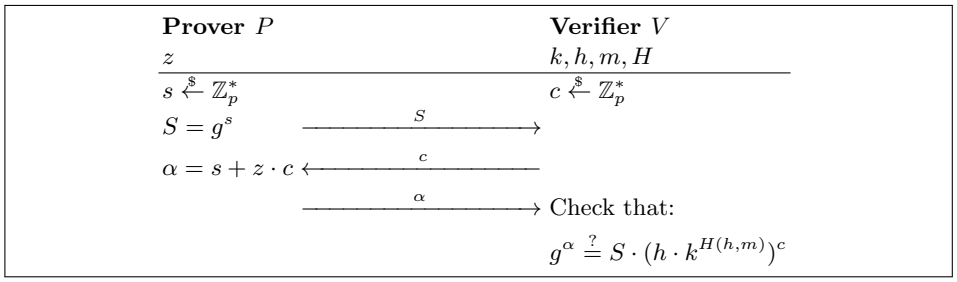

Fig. 1. Protocol Π_0

Proof of knowledge construction: Let $\mathbb{G}$ be a group of prime order p, g be a generator of $\mathbb{G}$ and n be an integer. Let k and h be two elements of $\mathbb{G}$, m be a bit-string and $H : \{0,1\}^* \to \mathbb{Z}_p^*$ be a hash function. Finally, for all $i \in \{1, \ldots, n\}$, we set the instance tuple $t_i = (k_i, h, m, H)$.

In the following, we show how to build Π, a non-interactive zero knowledge proof of knowledge of $z \in \mathbb{Z}_p^*$ such that there exists an instance $(k, h, m, H) \in \{t_i\}_{1 \leq i \leq n}$ such that $z = \log_g(h \cdot k^{H(h,m)})$. We first describe in Figure 1 the interactive case Π_0 where $n = 1$, hence there is only one instance $t = (k, h, m, H)$. It is a variant of the Schnorr protocol [9]. This proof is a sigma-protocol.

Lemma 1. *The ZKP Π_0 is complete, sound, and honest-verifier zero-knowledge.*

The proof of Lemma 1 is similar to the proof of the Schnorr protocol properties [9]. As Π_0 is honest-verifier zero knowledge and a sigma protocol, we can use the generic transformation of [4] to obtain the interactive version of our proof for any $n \geq 1$. Finally, using this transformation and the Fiat-Shamir heuristic on Π_0, we build the non-interactive proof Π in the random oracle model.

Theorem 1. *The NIZKP Π is complete, sound, and zero-knowledge in the random oracle model.*

Proof. It is a direct implication of [4] and Lemma 1.

Notation: We denote by $\Pi.\mathsf{Proof}(z, t, \{t_i\}_{1 \leq i \leq n})$ the algorithm that generates such a proof where z is the secret, $t = (k, h, m, H) \in \{t_i\}_{1 \leq i \leq n}$ is the instance corresponding to z such that $g^z = h \cdot k^{H(h,m)}$ and $\{t_i\}_{1 \leq i \leq n}$ is the set of all the instances. We denote by $\Pi.\mathsf{Verif}(\pi, \{t_i\}_{1 \leq i \leq n})$ the algorithm that checks the validity of the proof π according to the set of instances $\{t_i\}_{1 \leq i \leq n}$.

Scheme 1 (GAWP scheme) *$GAWP = (\mathsf{Init}, \mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{Ano}, \mathsf{AVer})$ is an ARS such that:*

$\mathsf{Init}(1^t)$**:** *This algorithm chooses a group $\mathbb{G}$ of prime order p according to the security parameter t. It then chooses a generator g of $\mathbb{G}$ and a hash function $H : \{0,1\}^* \to \mathbb{Z}_p^*$. It outputs $(\mathbb{G}, p, g, H)$.*

$\mathsf{Gen}(\mathsf{init})$**:** *This algorithm picks* $x \xleftarrow{\$} \mathbb{Z}_p^*$*, computes* $\mathsf{ssk} = x$ *and* $\mathsf{svk} = g^x$ *and returns* $(\mathsf{ssk}, \mathsf{svk})$*.*

$\mathsf{Sig}(\mathsf{ssk}, m)$**:** *This algorithm picks* $r \xleftarrow{\$} \mathbb{Z}_p^*$*, computes* $h = g^r$*,* $M = H(h, m)$ *and* $z = r + \mathsf{ssk} \cdot M$ *and returns* $\sigma = (h, z)$*.*

$\mathsf{Ver}(\mathsf{svk}, \sigma, m)$**:** *Using* $\sigma = (h, z)$*, if* $g^z = h \cdot \mathsf{svk}^{H(h,m)}$ *then this algorithm returns* 1*, else it returns* 0*.*

$\mathsf{Ano}(L, \sigma, m, \mathsf{svk})$**:** *Using* $\sigma = (h, z)$*, this algorithm computes a zero-knowledge proof of the knowledge of the witness* z *such that there exists* $k \in L$ *such that* $g^z = h \cdot k^{H(h,m)}$ *without revealing neither* z *nor* k*. More precisely, it uses the non-interactive zero-knowledge proof scheme* Π *to computes* $\widehat{\sigma} = \Pi.\mathsf{Proof}(z, (\mathsf{svk}, h, m, H), \{(k, h, m, H)\}_{k \in L})$ *and returns it.*

$\mathsf{AVer}(L, \widehat{\sigma}, m)$**:** *This algorithm computes* $b = \Pi.\mathsf{Verif}(\widehat{\sigma}, \{(k, h, m, H)\}_{k \in L})$ *and returns it.*

Security Analysis: We have the following theorem.

Theorem 2. *If there is no polynomial time algorithm* $\mathcal{A}$ *that solves the discrete logarithm problem with a non-negligible probability, then* $\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{GAWP}(t, n)$ *and* $\mathsf{Adv}^{\mathsf{anon}}_{GAWP}(t, n)$ *are both negligible in* t *for any polynomially bounded* n *in the random oracle model.*

We show this, through the two following lemmas

Lemma 2. *An ARSis unforgeable under the hardness of the discrete logarithm problem.*

Proof. We are going to show that is we have a polynomial adversary $\mathcal{A}$ able to forge our scheme in a polytnomial time with non negligible probability ϵ, then we can build a simulator $\mathcal{B}$ able to break the discrete logarithm problem with a similar polynomial time with probability ϵ/q where q is the number of users in the system.

Let assume $\mathcal{B}$ receives a discrete logarithm challenge $\mathsf{svk}_* \in \mathbb{G}$. We are going to build a sequence of games, allowing $\mathcal{B}$ to use adversary $\mathcal{A}$ to compute x such that $g^x = \mathsf{svk}_*$.

We first start the simulation by picking a random user and setting his public key as svk_*, all the user users have keys generated honestly (in other words the simulator $\mathcal{B}$ knows their corresponding signing keys). This means that if the adversary wants to corrupt some users, we can give him the correspondign secret keys.

When answering signing queries:

- If the signer is an honest user, $\mathcal{B}$ simply computes the signature honestly using the associated (known) secret signing keys.
- If the user, is the expected challenge user, then $\mathcal{B}$ picks a random $r \in \mathbb{Z}_p^*$, computes h^r, and simulates the Zero-Knowledge proof $\hat{\sigma}$ to say that this is indeed a valid signature for a set of users L containing the challenge user i_*. Under the Zero Knowledge property (hence the programmabilty of the ROM), this simulation is indistinguishable from a real signature.

After a polynomial number of signing queries, the adversary picks a user i'_* and returns a signature on an unsigned message / set of users. For the forgery to be valid, the signature has to be valid, the set of users should only contain uncorrupted users, and never have signed the said message.

From the adversary point of view honest signatures, and simulated ones are indistinguishable, hence with probability $1/q$ the adversary is going to pick the expected user as his challenge one.

Now using the extractability of the random oracle, $\mathcal{B}$ can recover the value ssk_* used to generate the proof (this can simply be done by rewinding the random oracle on the final proof computation). Hence after a polynomial time $\mathcal{B}$ is able to recover the discrete logarithm associated with the challenge with probability ϵ/q. □

Lemma 3. *The previous scheme is anonymous in the Random Oracle Model.*

Proof. Let us assume there exists an adversary $\mathcal{A}$ against the Anonymity of our scheme, we are going to build a simulator $\mathcal{B}$ in the Random Oracle Model.

We start from a game $\mathcal{G}_{0,0}$ where the simulator does everything honestly and picks the identity i_0. This includes generating honestly all the secret keys and signing.

We now change the generation of the challenge signature. The user still picks a random $r \in \mathbb{Z}_p^*$ but now simulates the Zero-Knowledge proof $\hat{\sigma}$. This is done by programming the random oracle, so that the challenge value fits with the guess done in the first part of the proof. This leads to the game $\mathcal{G}_{1,0}$. Using Theorem 1, this game is indistinguishable from the previous one.

A simulated challenge signature for the identity i_1 is exactly the same (as the simulation does not require the knowledge of the key), hence game $\mathcal{G}_{1,1}$ is identical to the previous one.

Finally, the simulator $\mathcal{B}$ switches back to an honest signature this time made by using the secret key i_1 to generate $\hat{\sigma}$. Once again, under the random oracle model, this game is indistinguishable from the previous one. □

Efficiency: In GAWP, the signature algorithm requires one exponentiation, and the anonymization algorithm requires $2 \times n$ exponentiations where n is the size of the group. In the scheme [7], the signature requires one exponentiation, but the anonymization requires $n+1$ exponentiations and $2 \times n+1$ pairing computations. Moreover, this scheme requires $2 \times n$ pairing computations and n exponentiations in the verification algorithm of an anonymous signature when our scheme requires only $2 \times n + 1$ exponentiations. Thus GAWP is more efficient than the scheme [7] that becomes impractical when the group contains a lot of members.

5 Conclusion

In this paper, we show that pairings are not needed in anonymizable ring signature: we design a paring-free scheme that is more efficient and secure under a more standard assumption as the previous scheme in [7]. Particularly, the

anonymization algorithm and the anonymous verification algorithm in [7] are very inefficient for large groups comparing to ours since it requires a number of pairing computations that is linear in the size of the group. The next step will be to design an anonymizable ring signature that can be proven secure without the random oracle heuristic.

References

1. A. Bender, J. Katz, and R. Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In S. Halevi and T. Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 60–79. Springer, Mar. 2006.
2. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Aug. 2004.
3. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, Sept. 2004.
4. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO94*, volume 839 of *LNCS*. Springer, 1994.
5. A. Fiat and A. Shamir. *Advances in Cryptology — CRYPTO' 86: Proceedings*, chapter How To Prove Yourself: Practical Solutions to Identification and Signature Problems, pages 186–194. Springer Berlin Heidelberg, Berlin, Heidelberg, 1987.
6. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, pages 186–208, 1989.
7. F. Hoshino, T. Kobayashi, and K. Suzuki. Anonymizable signature and its construction from pairings. In M. Joye, A. Miyaji, and A. Otsuka, editors, *PAIRING 2010*, volume 6487 of *LNCS*, pages 62–77. Springer, Dec. 2010.
8. R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In C. Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer, Dec. 2001.
9. C.-P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Aug. 1990.
10. K. Suzuki, F. Hoshino, and T. Kobayashi. Relinkable ring signature. In *Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings*, pages 518–536, 2009.