

# Group Signatures: Better Efficiency and New Theoretical Aspects

Jan Camenisch<sup>1</sup> and Jens Groth<sup>2,3</sup>

<sup>1</sup> IBM Research, Zurich Research Lab  
E-mail: [jca@zurich.ibm.com](mailto:jca@zurich.ibm.com)

<sup>2</sup> Cryptomathic

<sup>3</sup> BRICS\*, Dept. of computer Science, University of Aarhus  
E-mail: [jg@brics.dk](mailto:jg@brics.dk)

**Abstract.** A group signature scheme allows members of a group to anonymously sign messages. To counter misuse, the anonymity can be revoked by the so-called group manager.

This paper contributes two results to the area of group signatures. First, we improve the state-of-the-art scheme by Ateniese et al. by an order of magnitude. Our new scheme satisfies the recent security definition by Bellare et al. Second, and of a more theoretical nature, we study the Bellare et al. definitions and show that their notion of full-anonymity may require stronger assumptions than what is needed to achieve a relaxed but reasonable notion of anonymity.

## 1 Introduction

Group signatures, introduced by Chaum and van Heyst [11], allow a member to anonymously sign on behalf of the group. More precisely, distinguishing whether or not two group-signatures originated by the same or by different group members is infeasible to everyone but the group manager. A number of group signature schemes are proposed in the literature [11, 12, 10, 9, 3, 2, 6, 4, 1, 5, 8]. Many of them also allow members to join and leave the group at arbitrary times [2, 6, 22].

Group signatures have many applications in the space of privacy protection. The most prominent one is probably in trusted computing, where a computing device is required to authenticate as proper (i.e., secure) device, i.e., that it has obtained *attestation* by some third party. To protect privacy of the device's user, this authentication should not allow identification of the device. In fact, the protocol standardized by the Trusted Computing Group to achieve this [21] basically uses the Ateniese et al. group signature scheme [3] but without its anonymity revocation feature.

In this paper, we present a new practical group signature scheme that is related to the Ateniese et al. scheme [3]. We prove that it satisfies a strong security definition very similar to [4]. Security is proved in the random oracle model under the strong RSA assumption and a DDH assumption.

---

\* Basic Research in Computer Science ([www.brics.dk](http://www.brics.dk)), funded by the Danish National Research Foundation.

Our scheme is considerably faster than the state of the art scheme in [3]. Moreover, in our scheme the protocol to join the group only takes two rounds. The prospective member sends a join request to the group manager. The group manager sends a certificate back to the member.

The scheme supports dynamically joining new members to the group without changing the public key. Furthermore, it is possible to revoke a secret key such that it can no longer be used to sign messages. Revocation of a membership does require the public key to be modified. However, the modification is of constant size and allows group members in good standing to update their secret keys easily. To accomplish this goal we use methods similar to those of [6] and [22]. Their schemes are not as efficient as our scheme.

We present a modification of our scheme that with only a small loss of efficiency also allows us to make a full revocation, i.e., reveal all signatures signed with a revoked key. This scheme does not satisfy the [4] definition of security though. The problem is that given a private signature key it *is* possible to determine which signatures belong to the member in question.

As a separate theoretical contribution we show that the existence of one-way functions and NIZK arguments can be used to construct a group signature scheme. Again, we obtain a scheme that does not satisfy the [4] definition because a member's secret key does make it possible to identify signatures made by this member. We propose how to define security of group signature schemes when compromise of members' secret keys does matter.

We prove that the [4] definition implies IND-CCA2 secure public key bit-encryption. The existence of one-way functions and NIZK arguments does to our knowledge not entail the existence of public key encryption. Therefore, it seems that to satisfy [4] one must use stronger security assumptions than what is needed for just making a group signature scheme.

*State of the art.* The current state of the art group signature scheme is due to Ateniese et al. [3]. While being reasonably efficient, this scheme does not support certificate revocation. An extension by Ateniese, Song and Tsudik [2] implements the full revocation mentioned before, i.e., all bad signatures by the revoked member are revealed. Unfortunately, this scheme is rather inefficient. Camenisch and Lysyanskaya [6] and Tsudik and Xu [22] propose schemes with dynamic revocation. This means that after a certificate has been revoked the member cannot any longer make signatures. Both schemes are less efficient than [3]. [22] is more efficient than [6], but relies on a trusted third party to generate some of the data, and need to update the key both when members join and leave the group. [6] can easily be modified to only updating the verification key when memberships are revoked.

All the schemes mentioned here include in their assumptions the strong RSA assumption and the random oracle model. Ateniese and de Medeiros [1] suggest a scheme that does not rely on knowledge of the factorization of the modulus, but this scheme is much less efficient than [3]. [4] suggest a scheme based on any trapdoor permutation and without the random oracle model. This scheme is only a proof of concept; it is very inefficient.

Concurrent with our work, Boneh, Boyen, and Shacham [5] as well as Camenisch and Lysyanskaya [8] presented group signatures schemes based on bi-linear maps. While these schemes are more efficient, they are based on new and alternative number theoretic assumptions.

## 2 Definitions

A group signature scheme involves three types of parties: members, non-members and a group manager. It further consists of five algorithms KeyGen, Join, Sign, Verify, Open, and Revoke. The key generation algorithm produces  $(vk, gmsk) \leftarrow \text{KeyGen}()$  as output, where  $vk$  is a public verification key and  $gmsk$  is the group managers secret key. If the group of members is fixed, we may assume that the algorithm also outputs a vector  $sk$  of secret keys to be used by the members. If, however, the group of members is dynamic, KeyGen does not output secret keys for the members. Instead the Join protocol can be used to let non-members join the group. As a result of this protocol, a new member obtains a secret key  $sk_i$ , while the group manager obtains some information  $Y_i$  related to the new member that he includes into his secret key  $gmsk$ . To sign a message  $m$  the member runs  $\sigma \leftarrow \text{Sign}(sk_i, m)$ . To verify a signature  $\sigma$  on message  $m$  one computes  $\text{Verify}(vk, m, \sigma)$ . Furthermore, given a signature  $\sigma$  on  $m$ , the group manager can identify the originating member by computing  $\text{Open}(gmsk, m, \sigma)$ , which outputs the identity of the member who created the signature. Finally, using the Revoke algorithm  $(vk, gmsk) \leftarrow \text{Revoke}(gmsk, Y_i)$ , the group manager can exclude the member relating to  $Y_i$  from the group.

Bellare, Micciancio, and Warinschi [4] propose two properties, full-traceability and full-anonymity, that capture the security requirements of group signatures. These definition assume that the key generation is run by a trusted party and do not consider members joining or leaving the group after the key generation [4].

*Full-traceability.* The short description of full-traceability is that without a member's secret key it must be infeasible to create a valid signature that frames this member. This must hold even if the group manager's secret key and an arbitrary number of the members' secret keys are exposed.

Formally, we say that the group signature scheme has full-traceability if the expectation of the following experiment is negligible.

$\text{Exp}_A^{\text{f-trace}}(k) :$   
 $(vk, gmsk, sk) \leftarrow \text{KeyGen}(k)$   
 $(m, \sigma) \leftarrow \mathcal{A}^{\text{Sign}(sk, \cdot), \text{Corrupt}(\cdot)}(vk, gmsk)$   
 If  $\text{Verify}(vk, m, \sigma) = 1$ ,  $i = \text{Open}(gmsk, m, \sigma) \in [k]$ ,  $i$  was not queried  
 $\text{Corrupt}(\cdot)$  and  $(i, m)$  was not queried to  $\text{Sign}(sk, \cdot)$  then return 1  
 If  $\text{Verify}(vk, m, \sigma) = 1$  and  $i = \text{Open}(gmsk, m, \sigma) \notin [k]$  then return 1  
 Else return 0

Here  $\text{Corrupt}(\cdot)$  is an oracle that on query  $i \in [k]$  returns  $sk_i$ .

[4] argue that full-traceability implies what is meant by the more informal notions of unforgeability, no-framing, traceability, and coalition resistance as defined, e.g., in [3].

*Full-anonymity.* We want to avoid that signatures can be linked to group members or other signatures. For this purpose, we define full-anonymity as the notion that an adversary cannot distinguish signatures from two different members. This must hold even when we give the secret keys to the adversary. In other words, even if a member's key is exposed, then it is still not possible for the adversary to see whether this member signed some messages in the past, neither is it possible to see if any future messages are signed by this member.

$\mathbf{Exp}_{\mathcal{A}}^{\text{f-anon}}(b, k) :$   
 $(vk, gmsk, \mathbf{sk}) \leftarrow \text{KeyGen}(k)$   
 $(i_0, i_1, m) \leftarrow \mathcal{A}^{\text{Open}(gmsk, \cdot, \cdot)}(vk, \mathbf{sk}); \sigma \leftarrow \text{Sign}(sk_{i_b}, m)$   
 $d \leftarrow \mathcal{A}^{\text{Open}(gmsk, \cdot, \cdot)}(\sigma)$   
 If  $\mathcal{A}$  did not query  $m, \sigma$  return  $d$ , else return 0

We say the group signature scheme has full-anonymity if  $\Pr[\mathbf{Exp}_{\mathcal{A}}^{\text{f-anon}}(1, k) = 1] - \Pr[\mathbf{Exp}_{\mathcal{A}}^{\text{f-anon}}(0, k) = 1]$  is negligible.

[4] argue that full-anonymity entails what is meant by the more informal notions of anonymity and unlinkability.

*Anonymity.* The [4] model is strict in its anonymity requirements. It demands that even if a member's secret key is exposed it must still be impossible to tell which signatures are made by the member in question. This is a good definition of security in a threat model where parties may be corrupted adaptively but can erase data. The schemes in [3] and [6] have this strong anonymity property as does our new scheme with Join and Revoke.

In other threat models, this may be aiming too high. Consider for instance a static adversary, then the key is exposed before any messages are signed or it is never exposed. Or consider an adaptive adversary where parties cannot erase data, in this case full-anonymity does not buy us more security. We therefore define a weaker type of anonymity that is satisfied if both the group manager's secret key and the member's secret key are not exposed. We note that for instance the scheme in [11, 1, 22] satisfy only this weaker property. One positive effect of not requiring full-anonymity is that potentially it makes it possible for the member to claim a signature she made, i.e., prove that she signed a particular signature, without having to store specific data such as randomness, etc., used to generate the signature. This latter property is called claiming in [17].

$\mathbf{Exp}_{\mathcal{A}}^{\text{anon}}(b, k) :$   
 $(vk, gmsk, \mathbf{sk}) \leftarrow \text{KeyGen}(k)$   
 $(i_0, i_1, m) \leftarrow \mathcal{A}^{\text{Open}(gmsk, \cdot, \cdot), \text{Sign}(sk_{\cdot, \cdot}), \text{Corrupt}(\cdot)}(vk); \sigma \leftarrow \text{Sign}(sk_{i_b}, m)$   
 $d \leftarrow \mathcal{A}^{\text{Open}(gmsk, \cdot, \cdot), \text{Sign}(sk_{\cdot, \cdot})}(\sigma)$ <sup>4</sup>  
 If  $\mathcal{A}$  did not query  $m, \sigma$  to Open and did not query  $i_0, i_1$  to Corrupt( $\cdot$ ) then return  $d$ , else return 0

<sup>4</sup> We do not allow  $\mathcal{A}$  to corrupt member's in the second phase. This is simply because we WLOG may assume that it corrupts all other members than  $i_0$  and  $i_1$  before getting the challenge signature.

We say the group signature scheme is anonymous if  $\Pr[\text{Exp}_{\mathcal{A}}^{\text{anon}}(1, k) = 1] - \Pr[\text{Exp}_{\mathcal{A}}^{\text{anon}}(0, k) = 1]$  is negligible.

As Bellare et al. [4], we can argue that anonymity implies the informal notions of anonymity and unlinkability mentioned in the introduction.

### 3 Preliminaries

*Protocols to Prove Knowledge of and Relations among Discrete Logarithms.* In our scheme we will use various protocols to prove knowledge of and relations among discrete logarithms. To describe these protocols, we use notation introduced by Camenisch and Stadler [10] for various proofs of knowledge of discrete logarithms and proofs of the validity of statements about discrete logarithms. For instance,  $PK\{(\alpha, \beta, \gamma) : y = g^\alpha h^\beta \wedge \tilde{y} = \tilde{g}^\alpha \tilde{h}^\gamma \wedge (u \leq \alpha \leq v)\}$  denotes a “zero-knowledge Proof of Knowledge of integers  $\alpha, \beta,$  and  $\gamma$  such that  $y = g^\alpha h^\beta$  and  $\tilde{y} = \tilde{g}^\alpha \tilde{h}^\gamma$  holds, where  $u \leq \alpha \leq v,$ ” where  $y, g, h, \tilde{y}, \tilde{g},$  and  $\tilde{h}$  are elements of some groups  $G = \langle g \rangle = \langle h \rangle$  and  $\tilde{G} = \langle \tilde{g} \rangle = \langle \tilde{h} \rangle$ . The convention is that Greek letters denote the quantities the knowledge of which is being proved, while all other parameters are known to the verifier. Using this notation, a proof protocol can be described by just pointing out its aim while hiding all details.

In the random oracle model, such protocols can be turned into signature schemes using the Fiat-Shamir heuristic [13, 20]. We use the notation  $SPK\{(\alpha) : y = g^\alpha\}(m)$  to denote a signature obtained in this way.

*The Camenisch-Lysyanskaya Signature Scheme.* The group signature scheme is based on the Camenisch-Lysyanskaya (CL) signature scheme [7, 19]. Unlike most signature schemes, this one is particularly suited for our purposes as it allows for efficient protocols to prove knowledge of a signature and to retrieve signatures on secret messages efficiently using discrete logarithm based proofs of knowledge [7, 19]. We recall the signature scheme here.

*Key generation.* On input  $1^k$ , choose an RSA modulus  $n = pq, p = 2p' + 1, q = 2q' + 1$  as a product of safe primes. Choose, uniformly at random,  $g_1, \dots, g_L, h, a \in \text{QR}_n$ . Output the public key  $(n, g_1, \dots, g_L, h, a)$  and the secret key  $p$ . Let  $\ell_n$  be the length of  $n$ .

*Message space.* Let  $\ell_m$  be a parameter. The message space is the set  $\{(m_1, \dots, m_L) : m_i \in \pm\{0, 1\}^{\ell_m}\}$ .

*Signing algorithm.* On input  $m_1, \dots, m_L$ , choose a random prime number  $e$  of length  $\ell_e > \ell_m + 2$ , and a random number  $r$  of length  $\ell_r = \ell_n + \ell_m + \ell_s$ , where  $\ell_s$  is a security parameter. Compute the value  $y$  such that  $y^e \equiv ag_1^{m_1} \dots g_L^{m_L} h^r \pmod{n}$ . The signature on the message  $(m_1, \dots, m_L)$  consists of  $(e, y, r)$ .

*Verification algorithm.* To verify that the tuple  $(e, y, r)$  is a signature on message  $(m_1, \dots, m_L)$ , check that  $y^e \equiv ag_1^{m_1} \dots g_L^{m_L} h^r \pmod{n}$ , and check that  $2^{\ell_e} > e > 2^{\ell_e - 1}$ .

**Theorem 1 ([7]).** *The signature scheme is secure against adaptive chosen message attacks [15] under the strong RSA assumption.*

*Remarks.* The original scheme considered messages in the interval  $[0, 2^{\ell_m} - 1]$ . Here, however, we allow messages from  $[-2^{\ell_m} + 1, 2^{\ell_m} - 1]$ . The only consequence of this is that we need to require that  $\ell_e > \ell_m + 2$  holds instead of  $\ell_e > \ell_m + 1$ .

Further note that a signature can be randomized: It is clear that if  $y^e = ag^mh^r \pmod n$ , then we also have  $(yh)^e = ag^mh^{r+e} \pmod n$ . Thus the signature scheme is not strong but it is secure against chosen message attack.

The CL-signature scheme makes it possible to sign a committed message. One party computes the commitment  $g^mh^{r'}$  mod  $n$ , where  $r' \in_R \mathbb{Z}_n$  such that  $m$  is statistically hidden. This party also proves knowledge of  $m, r'$ . The signer now picks  $e$  as a random  $\ell_e = \ell_2$ -bit prime, and picks  $r'' \in \mathbb{Z}_{E_i}$ . He then computes  $y$  so  $y^e = ag^mh^{r'+r''}$  and returns  $(y, e, r'')$ . Now the party has a signature on  $m$  without the signer having any knowledge about which message was signed.

We note that careful analysis of the signature scheme's security proof shows that in fact the requirement of Camenisch and Lysyanskaya that  $\ell_r = \ell_n + \ell_m + \ell_s$  holds can be relaxed to  $\ell_r = \ell_e$ , by picking  $r \in_R \mathbb{Z}_e$ . However, if the goal is to sign a commitment message that shall be kept secret from the signer, one requires a larger  $r$ , for instance  $r \in_R \mathbb{Z}_n$ .

## 4 The Basic Group Signature Scheme

*The ideas underlying our group signature scheme.* We base our group signature scheme on two groups. One group is  $\text{QR}_n$ , where  $n$  is an RSA modulus chosen as a safe-prime product. The other group is of order  $Q$  in  $\mathbb{Z}_P^*$ , where  $Q|P-1$ .

Each member receives a CL-signature  $(y_i, e_i, r_i)$  on a message  $x_i$ . As part of a group signature they will prove knowledge of such a CL-signature. Since outsiders cannot forge CL-signatures this ensures that the signer is member of the group. As the group manager must be able to open signatures and identify the signer we include in the group signature also an encryption of  $Y_i = G^{x_i} \pmod P$ . The signer proves knowledge of  $x_i$  and that it is the same  $x_i$  that she knows a CL-signature on. The group manager knowing the secret key can decrypt and identify the signer. Because the group manager does not know  $x_i$  we avoid members being framed by malicious group managers. The group manager simply cannot compute the discrete logarithm  $x_i$ , and therefore not make a group signature pointing to the member.

In Figure 1 we present the actual protocol. Following the model of [4], it assumes that the key generation algorithm is run by a trusted third party. We later extend this scheme to include dynamic join and revocation such that this third party is not required.

The parameters of our schemes are as follows. We use  $\ell_s$  as a bit-length such that for any integer  $a$  when we pick  $r$  as a  $|a| + \ell_s$ -bit random number then  $a + r$  and  $r$  are statistically indistinguishable.  $\ell_c$  is the length of the output of the hash-function.  $\ell_e$  is a number large enough that we can assign all members different numbers and make the  $E_i$ 's prime.

It must be the case that  $\ell_c + \ell_e + \ell_s + 1 < \ell_Q$  and  $\ell_Q + \ell_c + \ell_s + 1 < \ell_E < \ell_n/2$ .

A suggestion for parameters is  $\ell_n = 2049$ ,  $\ell_P = 1600$ ,  $\ell_E = 404$ ,  $\ell_Q = 282$ ,  $\ell_c = 160$ ,  $\ell_e = \ell_s = 60$ . This choice should ensure that factoring an  $\ell_n$  bit number is

about as hard as computing discrete logarithms in a subgroups of size  $2^{\ell_Q}$  modulo an  $\ell_P$ -bit prime [18].

**Theorem 2.** *The basic group signature scheme has full-traceability and full-anonymity.*

The proof of Theorem 2 can be found in the full paper.

## 5 Join and Revoke

*Flexible Join.* It may be impractical to set up the signature scheme with all members in advance. Often groups are dynamic and we may have members joining after the public keys have been generated. The recent schemes [3, 6, 22] support members joining at arbitrary points in time. The schemes [6, 22] require that the public key be updated when a new member joins. However, they can easily be modified to the more attractive solution where the public key does not need to be updated when a member joins.

Our scheme supports members joining throughout the protocol. The idea is that the member generates  $Y_i = G^{x_i} \bmod P$  herself, so only she knows the discrete logarithm  $x_i$ . Jointly the group manager and the member generate  $ag^{x_i}h^{r_i} \bmod n$ , where  $r_i$  is so large that  $x_i$  is statistically hidden. Then she gives it to the group manager who generates  $(y_i, e_i)$  and give them to the member. Here we use that the CL-signature scheme is secure against adaptive chosen message attack such that members cannot forge signatures and thereby falsely join themselves.

*Revocation.* On occasions, it may be necessary to revoke a member's secret key. Since signatures are anonymous, the standard approach of using certificate revocation lists cannot be used. Following [6] we suggest using roots of some element  $w$  to implement revocation. A signature contains an argument of knowledge of a pair  $(w_i, E_i)$  such that  $w = w_i^{E_i} \bmod n$ . If we want to revoke a membership we update the public key to contain  $w \leftarrow w_i$ . Now this member may no longer prove knowledge of a root of  $w$  and thus she cannot sign messages any more.<sup>5</sup>

When changing the public key we need to communicate to the remaining members how they should update their secret keys. In our scheme, we do this by publishing  $e_i$  corresponding to the revoked member. Members in good standing may use this to obtain a root of the new  $w$  through a simple computation. This means that the change in the public key is of constant size, and old members may update their secret keys by downloading only a constant amount of public information.

The protocol is described in Figure 2.

*Performance.* We now discuss the performance of our group signature with join and revoke and compare it to the ACJT scheme [3] and it's extension to revocation by Camenisch and Lysyanskaya [7]. To compute a group-signature, one needs to do six exponentiations modulo  $P$  with exponents from  $\mathbb{Z}_Q$ , one exponentiation modulo  $n$  with

<sup>5</sup> A member with a revoked key can still sign messages under the old verification key and claim that they were signed when this key was valid. Whether such an attack makes sense depends on the application of the group signature scheme and is beyond the scope of the paper. One obvious solution is of course to add a time-stamp.

### Basic Group Signature Scheme

**KeyGen**( $k$ ): Choose an  $\ell_n$ -bit RSA modulus  $n = pq$  as a product of two safe primes  $p = 2p' + 1, q = 2q' + 1$ . Select at random  $a, g, h \in \text{QR}_n$ . Select at random  $\ell_Q$ -bit and  $\ell_P$ -bit primes  $Q, P$  such that  $Q|P - 1$ . Let  $F$  be an element of order  $Q$  in  $\mathbb{Z}_P^*$ . Choose at random  $X_G, X_H \in \mathbb{Z}_Q$  and set  $G = F^{X_G} \bmod P, H = F^{X_H} \bmod P$ .

Select at random  $x_1, \dots, x_k \in \mathbb{Z}_Q$  and select also at random  $r_1, \dots, r_k \in \mathbb{Z}_n$ .

Choose different random  $\ell_e$ -bit numbers  $e_1, \dots, e_k$  such that

$E_1 = 2^{\ell_E} + e_1, \dots, E_k = 2^{\ell_E} + e_k$  are primes. Compute  $y_1, \dots, y_k$  such that  $y_1^{E_1} = ag^{x_1}h^{r_1} \bmod n, \dots, y_k^{E_k} = ag^{x_k}h^{r_k} \bmod n$ .

Public key:  $vk = (n, a, g, h, Q, P, F, G, H)$ .

Group managers private key:

$gmsk = (vk, X_G, Y_1 = G^{x_1} \bmod P, \dots, Y_k = G^{x_k} \bmod P)$ .

Member  $i$ 's private key:  $sk_i = (vk, x_i, y_i, e_i, r_i)$ .

**Sign**( $sk_i, m$ ): Select at random  $r \in \{0, 1\}^{\ell_n/2}$  and  $R \in \mathbb{Z}_Q$ . Set  $u = h^r y_i \bmod n, U_1 = F^R \bmod P, U_2 = G^{R+x_i} = G^R Y_i \bmod P$ , and  $U_3 = H^{R+e_i} \bmod P$ .<sup>a</sup> Compute the (sub-)signature

$$\begin{aligned} SPK\{(\xi, \rho, \varepsilon, \tau) : a = u^{2^{\ell_E} + \varepsilon} g^{-\xi} h^\rho \bmod n \wedge U_1 = F^\tau \bmod P \wedge \\ U_2 = G^{\tau + \xi} \bmod P \wedge U_3 = H^{\tau + \varepsilon} \bmod P \wedge \\ \varepsilon \in \{-2^{\ell_e + \ell_c + \ell_s}, +2^{\ell_e + \ell_c + \ell_s}\} \wedge \xi \in \{-2^{\ell_Q + \ell_c + \ell_s}, 2^{\ell_Q + \ell_c + \ell_s}\}\}(m) \end{aligned}$$

i.e., choose  $r_x \in \{0, 1\}^{\ell_Q + \ell_c + \ell_s}, r_r \in \{0, 1\}^{\ell_n/2 + \ell_c + \ell_s}, r_e \in \{0, 1\}^{\ell_e + \ell_c + \ell_s}$ , and  $R_R \in \mathbb{Z}_Q$  and compute

$$\begin{aligned} v = u^{r_e} g^{-r_x} h^{r_r} \bmod n, & \quad V_1 = F^{R_R} \bmod P, \\ V_2 = G^{R_R + r_x} \bmod P, & \quad V_3 = H^{R_R + r_e} \bmod P. \end{aligned}$$

Compute a challenge  $c = \text{hash}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$  and set

$z_x = r_x + cx_i, z_r = r_r + c(-r_i - rE_i), z_e = r_e + ce_i$ , and  $Z_R = R_R + cR \bmod Q$ .

Signature:  $\sigma = (c, u, U_1, U_2, U_3, z_x, z_r, z_e, Z_R)$ .

**Verify**( $vk, m, \sigma$ ): Check that  $z_e \in \{0, 1\}^{\ell_e + \ell_c + \ell_s}$  and  $z_x \in \{0, 1\}^{\ell_Q + \ell_c + \ell_s}$ . Compute

$$\begin{aligned} v = a^{-c} g^{-z_x} h^{z_r} u^{c2^{\ell_E} + z_e} \bmod n, & \quad V_1 = U_1^{-c} F^{Z_R} \bmod P, \\ V_2 = U_2^{-c} G^{Z_R + z_x} \bmod P, & \quad V_3 = U_3^{-c} H^{Z_R + z_e} \bmod P \end{aligned}$$

and verify that  $c = \text{hash}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$

**Open**( $gmsk, m, \sigma$ ): Verify that the signature is valid.

Using  $X_G$  decrypt  $(U_1^{\frac{P-1}{Q}} \bmod P, U_2^{\frac{P-1}{Q}} \bmod P)$  to get  $G^{\frac{P-1}{Q} x_i} \bmod P$  and return  $i$ .

<sup>a</sup> Without knowledge of the factorization of  $n$ ,  $h^r \bmod n$  for  $r \in_R \{0, 1\}^{\ell_n/2}$  is indistinguishable from a random element in  $\text{QR}_n$  [14]. Therefore,  $u$  does not reveal  $y_i$  to outsiders.

**Fig. 1.** The Basic Group Signature Scheme.

### Join and Revoke

**KeyGen()**: Run KeyGen(0) of the basic scheme. Choose also at random  $w \in \text{QR}_n$  and include it in  $vk$ . Prove that  $g \in \langle h \rangle$  by running  $PK\{(\alpha) : g = h^\alpha\}$  using binary challenges. Set  $gmsk = (vk, p, q, X_G)$  where  $n = pq$ .

**Join**: The member selects at random  $x_i \leftarrow \mathbb{Z}_Q$  and computes  $Y_i = G^{x_i} \bmod P$ . She also forms a commitment to  $x_i, g^{x_i} h^{r'_i} \bmod n$  with  $r_i \in_R \mathbb{Z}_n$  and proves knowledge of  $x_i, r'_i$  fitting the above. She sends  $Y_i, g^{x_i} h^{r'_i} \bmod n$  and the proof to the group manager.

The group manager selects  $e_i \in \{0, 1\}^{\ell_e}$  such that  $E_i = 2^{\ell_e} + e_i$  is prime. He computes  $w_i = w^{E_i^{-1}} \bmod n$ . He selects at random  $r''_i \in \mathbb{Z}_e$  and sets

$y_i = (ag^{x_i} h^{r'_i + r''_i})^{E_i^{-1}} \bmod n$ . He sends  $w_i, y_i, E_i, r''_i$  back to the new member.

Her secret key is  $sk_i = (vk, w_i, x_i, r_i = r'_i + r''_i, y_i, e_i)$ .

**Sign**( $vk, sk_i, m$ ): Select at random  $r \in \{0, 1\}^{\ell_n/2}$  and  $R \in \mathbb{Z}_Q$ . Set  $u = h^r y_i w_i \bmod n$ ,  $U_1 = F^R \bmod P$ ,  $U_2 = G^{R+x_i} \bmod P$ , and  $U_3 = H^{R+e_i} \bmod P$ . Compute the (sub-)signature

$$\begin{aligned} SPK\{(\xi, \rho, \varepsilon, \tau) : & aw = u^{2^{\ell_e} + \varepsilon} g^{-\xi} h^\rho \bmod n \wedge U_1 = F^\tau \bmod P \wedge \\ & U_2 = G^{\tau + \xi} \bmod P \wedge U_3 = H^{\tau + \varepsilon} \bmod P \wedge \\ & \varepsilon \in \{-2^{\ell_e + \ell_c + \ell_s}, +2^{\ell_e + \ell_c + \ell_s}\} \wedge \xi \in \{-2^{\ell_Q + \ell_c + \ell_s}, 2^{\ell_Q + \ell_c + \ell_s}\}\}(m) \end{aligned}$$

i.e., choose  $r_x \in \{0, 1\}^{\ell_Q + \ell_c + \ell_s}$ ,  $r_r \in \{0, 1\}^{\ell_n/2 + \ell_c + \ell_s}$ ,  $r_e \in \{0, 1\}^{\ell_e + \ell_c + \ell_s}$ , and  $R_R \in \mathbb{Z}_Q$  and compute

$$\begin{aligned} v &= u^{r_e} g^{-r_x} h^{r_r} \bmod n, & V_1 &= F^{R_R} \bmod P, \\ V_2 &= G^{R_R + r_x} \bmod P, & V_3 &= H^{R_R + r_e} \bmod P. \end{aligned}$$

Compute a challenge  $c = \text{hash}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$  and  $z_x = r_x + cx_i$ ,  $z_r = r_r + c(-r_i - rE_i)$ ,  $z_e = r_e + ce_i$ ,  $Z_R = R_R + cR \bmod Q$ .

Signature:  $\sigma = (c, u, U_1, U_2, U_3, z_x, z_r, z_e, Z_R)$ .

**Verify**( $vk, m, \sigma$ ): Check that  $z_e \in \{0, 1\}^{\ell_e + \ell_c + \ell_s}$  and  $z_x \in \{0, 1\}^{\ell_Q + \ell_c + \ell_s}$ . Compute

$$\begin{aligned} v &= (aw)^{-c} g^{-z_x} h^{z_r} u^{c2^{\ell_e} + z_e} \bmod n, & V_1 &= U_1^{-c} F^{Z_R} \bmod P, \\ V_2 &= U_2^{-c} G^{Z_R + z_x} \bmod P, & V_3 &= U_3^{-c} H^{Z_R + z_e} \bmod P \end{aligned}$$

and verify that  $c = \text{hash}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$

**OpenProof**( $gmsk, i, m, \sigma$ ): This is the same as in the basic scheme.

**Revoke**( $gmsk, i$ ): Publish  $E_i$ . Replace in  $vk$  the element  $w$  with  $w_i$ .

Any member in good standing may update her secret key  $sk_j$  as follows. She selects  $\alpha, \beta$  such that  $\alpha E_i + \beta E_j = 1$ . Then she computes the new  $w_j \leftarrow w_i^{\beta E_i} w_j^{\alpha E_j} \bmod n$ .

**Fig. 2.** Protocol for Dynamic Join and Revoke.

an exponent of length  $\ell_n/2$ , and one multi-base exponentiation with one exponent of length  $\ell_n/2 + \ell_c + \ell_s$  and two of length at most  $\ell_Q + \ell_s + \ell_c$ . In a good implementation, the computation of the multi-base exponentiation takes about 10 percent more time than a single exponentiation with an exponent of length  $\ell_n/2 + \ell_c + \ell_s$ .

The verification of a signature requires three two-base exponentiations modulo  $P$  and one multi-base exponentiation modulo  $n$ . As one of the exponents of the two-base exponentiations modulo  $P$  is rather small ( $\ell_c$  bits), these three take roughly the same time as three ordinary exponentiations modulo  $P$ . Concerning the multi-base exponentiation modulo  $n$ , the same statements as for the multi-base exponentiation modulo  $n$  in the signature generation holds.

Let us compare this with the [3] group signature scheme. In order to achieve the same security as in our scheme, the modulus  $n$  used there needs to be about 3200 bits. The reason is that in their scheme, the group manager is given a value  $B_i = a^{x_i} a_0 \bmod n$  by a member, where  $x_i$  is the member's secret. As the group manager knows the factorization of  $n$ , he has an advantage when trying to compute discrete logarithms modulo  $n$  and hence to compute  $x_i$ .

Now, the computation of a signature in the ACJT scheme takes four exponentiations modulo  $n$  with exponents about the size of  $n^2$  and three multi-base exponentiations with exponents the size of about  $n^3$ . Assuming that all the exponentiations in the ACJT and our scheme were carried out with the same modulus (which is quite a bit in favor of the ACJT scheme), our scheme is about 20 times more efficient.) Moreover, our scheme also provides revocation which the ACJT scheme does not. The extension of the ACJT to revocation proposed by Camenisch and Lysyanskaya requires about four multi-base base exponentiation with a 2048-bit modulus and exponents, in which case our scheme is more than 26 times more efficient.

Finally we note that the ACJT scheme requires that the member are assured that the modulus  $n$  is a safe prime product while in our scheme it is sufficient that they are convinced that  $g \in \langle h \rangle$ . The latter can be achieved *much* more efficiently than the former.

*Separating the membership management and the anonymity revocation capability.* There may be cases where we want to separate the process of granting (and revoking) membership and the process of revoking anonymity of signatures. A simple modification to our scheme allows for this.

The idea is that  $n$  is generated by the membership manager who can produce the needed CL signatures that we use in our scheme. On the other hand we let the anonymity revocation manager generate  $G, H$ . The membership manager then registers  $G^{x_i} \bmod P$  and  $H^{e_i} \bmod P$  with the anonymity revocation managers.

Now, if the member that wants to sign a message picks  $r$  and  $r_r$  large enough (for instance from  $\{0, 1\}^{\ell_n + \ell_s}$ ), then in the group  $QR_n$  everything is statistically hidden. Furthermore in  $\mathbb{Z}_P^*$  everything is encrypted. Therefore, the membership manager can no longer see who signs a particular message. However, the membership manager needs to prove that  $y_i, g, w_i \in \langle h \rangle$ , otherwise the side-classes might leak information. We refer to the full paper for the details of this,

## 6 Full Revocation

*Revocation revisited.* The current method of revocation does not allow us to revoke signatures valid under an old key. It would be highly impractical to demand that all

members re-sign messages when the public key is updated. Instead, we would prefer a solution parallel to that of certificate revocation lists that allow us to publish information that marks signatures signed by the now distrusted member. Nevertheless, of course we still want to preserve the privacy of all other members so we cannot simply reveal the group manager's secret key.

We propose an addition that solves this problem. The idea is to pick a random element  $s_i \in \mathbb{Z}_Q$  when the member joins. The member can now form  $F^R \bmod P$  and  $F^{Rs_i} \bmod P$  and include them in a group signature. According to the DDH assumption this will just look like two random elements. However, if the group manager releases  $s_i$ , then all signatures suddenly become clearly marked as belonging to said member.

We do need to force the member to use  $s_i$ , otherwise the member could create group signatures that could not be full-revoked. Therefore, we include a random element  $f \in \text{QR}_n$  in the public key and give the member a CL-signature on the form  $(y_i, E_i, r_i)$ , where  $y_i^{E_i} = af^{s_i}g^{x_i}hr_i \bmod n$ . The member will form  $U_4, V_4$  as  $U_4 = F^{Rs_i} \bmod P$  and  $V_4 = F^{d_s} \bmod P$ , when making the signature and argues correctness of this together with an argument that  $s_i$  is included in the CL-signature that she knows.

The protocol is described in Figure 3.

*Security.* A member's secret key contains  $s_i$ . Therefore, if the secret key is exposed it is easy to link the member with the signatures she has made. We can therefore not hope to have full anonymity but must settle for anonymity.

In theory, it is possible to construct a signature scheme that supports full revocation and full-anonymity. One idea could be that the group manager selects elements  $A_i, B_i$  with  $B_i = A_i^{X_i} \bmod P$  and signs these elements. Then the member must produce in addition to the standard signature a pair  $(A_i^R \bmod P, B_i^{RX_i} \bmod P)$  and prove in zero-knowledge that it has been properly formed. Once the group manager wants to make a full revocation he publishes  $X_i$ . However, the member's secret key does not include  $X_i$  so exposure of this key does not reveal which messages she has signed. This method is not very efficient though. It is an open problem to come up with an efficient group signature scheme that has full-anonymity and supports full revocation.

On the flip side we note that it may be seen as a positive thing that the member's signing key reveals which messages she signed. In [17]'s notion of traceable signatures it is a requirement that the member should be able to claim his signature. When the member's secret key links him to her signatures then this can be done easily without her having to store old randomness used in specific signatures that she might later want to claim.

## 7 Separating Full-Anonymity and Anonymity

*Full-anonymity implies IND-CCA2 public key bit-encryption.* To appreciate the strength of the [4] definition of security of a group signature scheme, we note that full-anonymity implies CCA2 secure public key bit-encryption.

**Theorem 3.** *If a group signature scheme satisfying full-anonymity exists, then an IND-CCA2 public key cryptosystem for encrypting bits exists.*

### Group Signature with Full Revocation

**KeyGen()**: As in the basic scheme except we now also include a random element  $f$  from  $QR_n$  in the public key, as well as  $w \in_R QR_n$ .

**Join**: The Join protocol remains the same except now the member chooses a random element  $s_i \in \mathbb{Z}_Q$  and gets  $y_i = (af^{s_i}g^{x_i}h^{r'_i+r''_i})^{E_i^{-1}} \bmod n$ , while the group manager learns  $s_i$ .

**Sign**( $vk, sk_i, m$ ): Choose randomizers as in the Join and Revoke scheme and set  $u = h^r y_i w_i \bmod n$ ,  $U_1 = F^R \bmod P$ ,  $U_2 = G^{R+x_i} \bmod P$ ,  $U_3 = H^{R+e_i} \bmod P$ , and  $U_4 = U_1^{s_i} \bmod P$ .

Compute the (sub-)signature

$$\begin{aligned} SPK\{(\psi, \xi, \rho, \varepsilon, \tau) : & aw = u^{2^{\ell E + \varepsilon}} f^{-\psi} g^{-\xi} h^{\rho} \bmod n \wedge U_1 = F^{\tau} \bmod P \wedge \\ & U_2 = G^{\tau + \xi} \bmod P \wedge U_3 = H^{\tau + \varepsilon} \bmod P \wedge U_4 = U_1^{\psi} \bmod P \wedge \\ & \varepsilon \in \{-2^{\ell e + \ell c + \ell s}, +2^{\ell e + \ell c + \ell s}\} \wedge \psi, \xi \in \{-2^{\ell Q + \ell c + \ell s}, 2^{\ell Q + \ell c + \ell s}\}\}(m) \end{aligned}$$

i.e., choose  $r_s \in \{0, 1\}^{\ell Q + \ell c + \ell s}$ ,  $r_x \in \{0, 1\}^{\ell Q + \ell c + \ell s}$ ,  $r_r \in \{0, 1\}^{\ell n / 2 + \ell c + \ell s}$ ,  $r_e \in \{0, 1\}^{\ell e + \ell c + \ell s}$ , and  $R_R \in \mathbb{Z}_Q$  and compute

$$\begin{aligned} v &= u^{r_e} f^{-r_s} g^{-r_x} h^{r_r} \bmod n, & V_1 &= F^{R_R} \bmod P, \\ V_2 &= G^{R_R + r_x} \bmod P, & V_3 &= H^{R_R + r_e} \bmod P, & V_4 &= U_1^{r_s} \bmod P, \end{aligned}$$

Compute a challenge  $c = \text{hash}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$  and  $z_s = r_s + cs_i$ ,  $z_x = r_x + cx_i$ ,  $z_r = r_r + c(-r_i - rE_i)$ ,  $z_e = r_e + ce_i$ ,  $Z_R = R_R + cR \bmod Q$ .

**Signature**:  $\sigma = (c, u, U_1, U_2, U_3, U_4, z_s, z_r, z_x, z_e, Z_R)$ .

**Verify**( $vk, m, \sigma$ ): Check that  $z_e \in \{0, 1\}^{\ell e + \ell c + \ell s}$  and  $z_s, z_x \in \{0, 1\}^{\ell Q + \ell c + \ell s}$ . Compute

$$\begin{aligned} v &= (aw)^{-c} f^{-z_s} g^{-z_x} h^{z_r} u^{c2^{\ell E} + z_e} \bmod n, & V_1 &= U_1^{-c} F^{Z_R} \bmod P, \\ V_2 &= U_2^{-c} G^{Z_R + z_x} \bmod P, & V_3 &= U_3^{-c} H^{Z_R + z_e} \bmod P, & V_4 &= U_4^{-c} U_1^{z_s} \bmod P \end{aligned}$$

and verify that  $c = \text{hash}(vk, u, v, U_1, U_2, U_3, U_4, V_1, V_2, V_3, V_4, m)$

**Open**( $gmsk, m, \sigma$ ): The opening protocol remains the same.

**Revoke**( $gmsk, i$ ): The revocation protocol remains the same.

**FullRevoke**( $gmsk, i$ ): Look up  $s_i$  and publish it on the certificate revocation list. Execute **Revoke**( $gmsk, i$ ).

Since  $s_i$  is now public anybody may check in old signatures whether

$U_4^{\frac{P-1}{Q}} = U_1^{\frac{P-1}{Q} s_i} \bmod P$  and therefore whether the signatures have been formed by the fully revoked member.

**Fig. 3.** Group Signature with Full Revocation.

We refer to the full paper for the proof.

[1] speculate whether it is possible to construct a group signature scheme based only on one-way functions. Following [16] we believe it is not possible to construct public key encryption from one-way functions, and therefore not possible to construct a group signature scheme from one-way functions that satisfies the security definition of [4].

*Group signature from one-way function and NIZK argument.* From the full paper we get the following theorem.

**Theorem 4.** *If one-way functions and non-interactive zero-knowledge arguments exist for some suitable language, then group signature schemes with full-traceability and anonymity exist.*

We do not know of any construction of public key encryption from one-way functions and non-interactive zero-knowledge arguments. Theorems 3 and 4 therefore indicate that a group signature scheme having full-anonymity may require stronger assumptions than what is needed to obtain anonymity.

The scheme in the full paper can easily be extended to a traceable signature scheme [17]. Theorems 3 and 4 can then be seen as indications that group signatures require stronger assumptions than traceable signature schemes.

## References

1. Giuseppe Ateniese and Breno de Medeiros. Efficient group signatures without trapdoors. In *proceedings of ASIACRYPT '03, LNCS series, volume 2894*, pages 246–268, 2003. Revised paper available at <http://eprint.iacr.org/2002/173>.
2. Giuseppe Ateniese, Dawn Xiaodong Song, and Gene Tsudik. Quasi-efficient revocation in group signatures. In *proceedings of Financial Cryptography '02*, pages 183–197, 2002.
3. Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure group signature scheme. In *proceedings of CRYPTO '00, LNCS series, volume 1880*, pages 255–270, 2000.
4. Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *proceedings of EUROCRYPT '03, LNCS series, volume 2656*, pages 614–629, 2003.
5. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures using strong diffie hellman. In *Advances in Cryptology — CRYPTO 2004*, LNCS. Springer Verlag, 2004.
6. Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *proceedings of CRYPTO '02, LNCS series 2442, volume*, pages 61–76, 2002.
7. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *SCN '02, LNCS series, volume 2576*, pages 268–289, 2002.
8. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Advances in Cryptology — CRYPTO 2004*, LNCS. Springer Verlag, 2004.
9. Jan Camenisch and Markus Michels. A group signature scheme with improved efficiency. In Kazuo Ohta and Dinqyi Pei, editors, *Advances in Cryptology — ASIACRYPT '98*, volume 1514 of LNCS, pages 160–174. Springer Verlag, 1998.
10. Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In *proceedings of CRYPTO '97, LNCS series, volume 1294*, pages 410–424, 1997.
11. David Chaum and Eugène van Heyst. Group signatures. In *proceedings of EUROCRYPT '91, LNCS series, volume 547*, pages 257–265, 1991.
12. Lidong Chen and Torben Pryds Pedersen. New group signature schemes. In Alfredo De Santis, editor, *Advances in Cryptology — EUROCRYPT '94*, volume 950 of LNCS, pages 171–181. Springer-Verlag, 1995.

13. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *proceedings of CRYPTO '86, LNCS series, volume 263*, pages 186–194, 1986.
14. Oded Goldreich and Vered Rosen. On the security of modular exponentiation with application to the construction of pseudorandom generators. *Journal of Cryptology*, 16(2):71–93, 2003.
15. Shafi Goldwasser, Silvio Micali, and Ronald Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.
16. Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *proceedings of STOC '89*, pages 44–61, 1989.
17. Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable signatures. Cryptology ePrint Archive, Report 2004/007, 2004. <http://eprint.iacr.org/>.
18. Arjen K. Lenstra and Eric K. Verheul. Selecting cryptographic key sizes. *Journal of Cryptology*, 14(4):255–293, 2001.
19. Anna Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, LNCS, pages 597–612. Springer Verlag, 2002.
20. David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, volume 1070 of LNCS, pages 387–398. Springer Verlag, 1996.
21. Trusted Computing Group. TCG TPM specification 1.2. Available at [www.trustedcomputinggroup.org](http://www.trustedcomputinggroup.org), 2003.
22. Gene Tsudik and Shouhuai Xu. Accumulating composites and improved group signing. In *proceedings of ASIACRYPT '03, LNCS series, volume 2894*, pages 269–286, 2003.