Group Signatures: Better Efficiency and New Theoretical Aspects

Jan Camenisch* Jens Groth†

Abstract

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

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 [CvH91], 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 [CvH91, CP95, CS97, CM98, ACJT00, AST02, CL02a, BMW03, AdM03, BBS04, CL04]. Many of them also allow members to join and leave the group at arbitrary times [AST02, CL02a, TX03].

Group signatures have many applications in the area 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 [Tru03] uses the Ateniese et al. group signature scheme [ACJT00] 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 [ACJT00]. We prove that it satisfies a strong security definition very similar to [BMW03]. Security is proved in the random oracle model under the strong RSA assumption and a DDH assumption.

Our scheme is considerably faster than the state of the art scheme in [ACJT00]. 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 the goal we use methods similar to those of [CL02a] and [TX03]. 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

*IBM Research, Zurich Research Lab, jca@zurich.ibm.com

†Cryptomathic and University of Aarhus, Dept. of Computer Science, BRICS, jg@brics.dk

[BMW03] 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 [BMW03] 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 [BMW03] definition implies IND-CCA2 secure public key bit-encryption. As far as we know, the existence of one-way functions and NIZK arguments does not entail the existence of public key encryption. Therefore, it seems that to satisfy [BMW03] 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. [ACJT00]. While being reasonably efficient, this scheme does not support certificate revocation. An extension by Ateniese, Song and Tsudik [AST02] 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 [CL02a] and Tsudik and Xu [TX03] 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 [ACJT00]. [TX03] is more efficient than [CL02a], 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. [CL02a] 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 [AdM03] suggest a scheme that does not rely on knowledge of the factorization of the modulus, but this scheme is much less efficient than [ACJT00]. [BMW03] 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 [BBS04] as well as Camenisch and Lysyanskaya [CL04] presented groups signatures schemes based on bilinear 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 $(vk, \textit{gmsk}) \leftarrow \mathrm{KeyGen}()$ 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 $\vec{sk}$ 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 the end 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 $\sigma \leftarrow \mathrm{Sign}(sk_i, m)$. To verify a signature σ on message m one computes $\mathrm{Verify}(vk, m, \sigma)$. Furthermore, given a signature σ on m, the group manager can identify the originating member by computing $\mathrm{Open}(\textit{gmsk}, m, \sigma)$, which outputs the identity of the member who created the signature. Finally, using the Revoke algorithm $(vk, \textit{gmsk}) \leftarrow \mathrm{Revoke}(\textit{gmsk}, Y_i)$, the group manager can exclude the member relating to Y_i from the group.

Bellare, Micciancio, and Warinschi [BMW03] 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 [BMW03]. to include a dynamically changing membership.

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.

$\mathbf{Exp}_{\mathcal{A}}^{\mathrm{f-trace}}(k)$:

$(vk, \mathit{gmsk}, \vec{sk}) \leftarrow \mathrm{KeyGen}(k)$
$(m, \sigma) \leftarrow \mathcal{A}^{\mathrm{Sign}(sk_\cdot,\cdot),\mathrm{Corrupt}(\cdot)}(vk, \mathit{gmsk})$
If $\mathrm{Verify}(vk, m, \sigma) = 1$, $i = \mathrm{Open}(\mathit{gmsk}, m, \sigma) \in [k]$, i was not queried $\mathrm{Corrupt}(\cdot)$ and (i, m) was not queried to $\mathrm{Sign}(sk_\cdot, \cdot)$ then return 1
If $\mathrm{Verify}(vk, m, \sigma) = 1$ and $i = \mathrm{Open}(\mathit{gmsk}, m, \sigma) \notin [k]$ then return 1
Else return 0

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

[BMW03] 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 [ACJT00].

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.

$\mathbf{Exp}_{\mathcal{A}}^{\mathrm{f-anon}}(b, k)$:

$(vk, \mathit{gmsk}, \vec{sk}) \leftarrow \mathrm{KeyGen}(k)$
$(i_0, i_1, m) \leftarrow \mathcal{A}^{\mathrm{Open}(\mathit{gmsk},\cdot,\cdot)}(vk, \vec{sk}); \sigma \leftarrow \mathrm{Sign}(sk_{i_b}, m)$
$d \leftarrow \mathcal{A}^{\mathrm{Open}(\mathit{gmsk},\cdot,\cdot)}(\sigma)$
If $\mathcal{A}$ did not query m, σ return d, else return 0

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

[BMW03] argue that full-anonymity entails what is meant by the more informal notions of anonymity and unlinkability.

Anonymity. The [BMW03] 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 [ACJT00] and [CL02a] 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 [CvH91, AdM03, TX03] 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 [KTY04].

$\mathbf{Exp}^{\mathrm{anon}}_{\mathcal{A}}(b,k)$:

$(vk, \mathit{gmsk}, \vec{sk}) \leftarrow \mathrm{KeyGen}(k)$
$(i_0, i_1, m) \leftarrow \mathcal{A}^{\mathrm{Open}(\mathit{gmsk},\cdot,\cdot),\mathrm{Sign}(sk_\cdot,\cdot),\mathrm{Corrupt}(\cdot)}(vk); \sigma \leftarrow \mathrm{Sign}(sk_{i_b}, m)$
$d \leftarrow \mathcal{A}^{\mathrm{Open}(\mathit{gmsk},\cdot,\cdot),\mathrm{Sign}(sk_\cdot,\cdot)}(\sigma)$[1]
If $\mathcal{A}$ did not query m, σ to Open and did not query i_0, i_1 to $\mathrm{Corrupt}(\cdot)$ then return d, else return 0

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

As Bellare et al. [BMW03], 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 [CS97] for various proofs of knowledge of discrete logarithms and proofs of the validity of statements about discrete logarithms. For instance, $\mathit{PK}\{(\alpha,\beta,\gamma) : y = g^\alpha h^\beta \ \wedge \ \tilde{y} = \tilde{g}^\alpha \tilde{h}^\gamma \ \wedge \ (u \leq \alpha \leq v)\}$ denotes a "*zero-knowledge* P*roof of* K*nowledge of integers* α, β, *and* γ *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 [FS86, PS96]. We use the notation $\mathit{SPK}\{(\alpha) : y = g^\alpha\}(m)$ to denote a signature obtained in this way.

Cryptographic Assumptions. We prove security under the strong RSA assumption and the decisional Diffie-Hellman assumption.

Assumption 1 (Strong RSA Assumption) *The strong RSA (SRSA) assumption states that it is computationally infeasible, on input a random RSA modulus* n *and a random element* $u \in \mathbb{Z}_n^*$*, to compute values* $e > 1$ *and* v *such that* $v^e \equiv u \pmod{n}$.

The tuple (n, u) generated as above is called an *instance* of the *flexible* RSA problem.

Assumption 2 (DDH Assumption) *Let* p *be an* ℓ_p*-bit prime and* p *an* ℓ_q*-bit prime such that* $q|p-1$*. Let* $g \in \mathbb{Z}_p^*$ *be an element of order* q*. Then, for sufficiently large values of* ℓ_p *and* ℓ_q*, the distribution* $\{(g, g^a, g^b, g^{ab})\}$ *is computationally indistinguishable from the distribution* $\{(g, g^a, g^b, g^c)\}$*, where* a*,* b*, and* c *are random elements from* $[0, q-1]$

[1] 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.

The Camenisch-Lysyanskaya Signature Scheme. The group signature scheme is based on the Camenisch-Lysyanskaya (CL) signature scheme [CL02b, Lys02]. 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 [CL02b, Lys02]. 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, $g_1, \ldots, g_L, h, a \in \mathrm{QR}_n$. Output the public key $(n, g_1, \ldots, g_L, h, a)$ and the secret key p. Let ℓ_n be the length of n.

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

Signing algorithm. On input $m_1, \ldots, m_L$, 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 ℓ_s is a security parameter. Compute the value y such that $y^e \equiv a g_1^{m_1} \ldots g_L^{m_L} h^r \pmod{n}$. The signature on the message $(m_1, \ldots, m_L)$ consists of (e, y, r).

Verification algorithm. To verify that the tuple (e, y, r) is a signature on message $(m_1, \ldots, m_L)$, check that $y^e \equiv a g_1^{m_1} \ldots g_L^{m_L} h^r \pmod{n}$, and check that $2^{\ell_e} > e > 2^{\ell_e - 1}$.

Theorem 1 ([CL02b]) *The signature scheme is secure against adaptive chosen message attacks [GMR88] 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 $y^e = a g^m h^r \bmod n$, then we also have $(yh)^e = a g^m h^{r+e} \bmod n$. 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 $g^m h^{r'} \bmod n$, where $r' \leftarrow \mathbb{Z}_n$ 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 $y^e = a g^m h^{r' + r''}$ 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 $r \leftarrow \mathbb{Z}_e$. 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 $r \leftarrow \mathbb{Z}_n$.

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 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 $Y_i = G^{x_i} \bmod P$. 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 cannot compute the discrete logarithm x_i, and therefore cannot make a group signature pointing to the member.

In Figure 1, we present the actual protocol. Following the model of [BMW03], 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.

Parameters. The parameters of our schemes are as follows. We use ℓ_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. ℓ_c is the length of the output of the hash-function. ℓ_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 $\ell_n = \ell_P = 2048$, $\ell_E = 404$, $\ell_Q = 282$, $\ell_c = 160$, $\ell_e = \ell_s = 60$. This choice should ensure that factoring an ℓ_n bit number is about as hard as computing discrete logarithms modulo an ℓ_P-bit prime [LV01].

The proof of the following theorem follows from Lemma 7 and Lemma 8 that are found in Appendix A.

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

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 [ACJT00, CL02a, TX03] support members joining at arbitrary points in time. The schemes [CL02a, TX03] 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 gives 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 [CL02a] 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.[2]

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.

[2] 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 ℓ_n-bit RSA modulus $n = pq$ as a product of two safe primes $p = 2p' + 1, q = 2q' + 1$. Select at random $a, g, h \in \mathrm{QR}_n$.

Select at random ℓ_Q-bit and ℓ_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 $x_1, \ldots, x_k \in Z_Q$ and select at random $r_1, \ldots, r_k \in \mathbb{Z}_n$.

Choose different random ℓ_e-bit numbers $e_1, \ldots, e_k$ such that $E_1 = 2^{\ell_E} + e_1, \ldots, E_k = 2^{\ell_E} + e_k$ are primes. Compute $y_1, \ldots, y_k$ such that $y_1^{E_1} = ag^{x_1}h^{r_1} \bmod n, \ldots, y_k^{E_k} = ag^{x_k}h^{r_k} \bmod n$.

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

Group managers private key: $gmsk = (vk, X_G, Y_1 = G^{x_1} \bmod P, \ldots, Y_k = G^{x_k} \bmod P)$.

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 $U_3 = H^{R+e_i} \bmod P$.[a] Compute the (sub-)signature

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

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

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

Compute a challenge $c = \mathrm{hash}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$ and set $z_x = r_x + cx_i$, $z_r = r_r + c(-r_i - rE_i)$, $z_e = r_r + ce_i$, 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, σ)**:** 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

$$v = a^{-c} g^{-z_x} h^{z_r} u^{c2^{\ell_E}+z_e} \bmod n,$$

$$V_1 = U_1^{-c} F^{Z_R} \bmod P, \qquad V_2 = U_2^{-c} G^{Z_R+z_x} \bmod P, \qquad V_3 = U_3^{-c} H^{Z_R+z_e} \bmod P$$

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

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.

[a] Goldreich and Rosen [GR03] show that to someone not knowing the factorization of n the element $h^r \bmod n$ is indistinguishable from a random element in QR_n for $r \in \{0,1\}^{\ell_n/2}$. This means that u does not reveal y_i to outsiders.

Figure 1: The Basic Group Signature Scheme.

Security. As in the proof of Claim 7.3, we could have generated the elements w, a, g, h independently of the factorization of n such that we could still make signatures. Therefore, the adversary does not gain anything extra from the ability to adaptively join members and revoke members. The scheme still has full-anonymity.

With respect to full-traceability, we have a problem since the group manager's secret key now contains a factorization of n, and Claim 7.3 no longer holds. However, Claims 7.2 and 7.1 still hold so member i cannot be framed since only she knows the discrete logarithm of Y_i.

Remaining is the problem that the adversary may frame the group manager by creating a signature that cannot be opened to any member. However, here we can prove that without knowledge of the group manager's

Join and Revoke

KeyGen(): Run $\mathrm{KeyGen}(0)$ of the basic scheme. Choose also at random $w \in \mathrm{QR}_n$ 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 $y_i = (a g^{x_i} h^{r_i' + r_i''})^{E_i^{-1}} \bmod n$. 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

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

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

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

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

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

Verify(vk, m, σ)**:** 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

$$v = (aw)^{-c} g^{-z_x} h^{z_r} u^{c 2^{\ell_E} + z_e} \bmod n,$$
$$V_1 = U_1^{-c} F^{Z_R} \bmod P, \qquad V_2 = U_2^{-c} G^{Z_R + z_x} \bmod P, \qquad V_3 = U_3^{-c} H^{Z_R + z_e} \bmod P$$

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

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 α, β such that $\alpha E_i + \beta E_j = 1$. Then she computes the new $w_j \leftarrow w_i^{\beta E_i} w_j^{\alpha E_j} \bmod n$.

Figure 2: Protocol for Dynamic Join and Revoke.

secret key this is not possible.

Performance. We now discuss the performance of our group signature with join and revoke and compare it to the ACJT scheme [ACJT00] and its extension to revocation by Camenisch and Lysyanskaya [CL02b]. To compute a group-signature, one needs to do six exponentiations modulo P with exponents from $\mathbb{Z}_Q$, one exponentiation modulo n with 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 (ℓ_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 [ACJT00] group signature scheme. In order to achieve the same security as in our scheme, the modulus n used there needs to be about 4096 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 exponentiations 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.

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 $f \in \mathrm{QR}_n$ in the public key and give the member a CL-signature on the form (y_i, E_i, r_i), where $y_i^{E_i} = a f^{s_i} g^{x_i} h^{r_i} \bmod n$. 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.

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 $w \in_R \mathrm{QR}_n$.

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

$$SPK\{(\psi, \xi, \rho, \varepsilon, \tau) : aw = u^{2^{\ell_E}+\varepsilon} f^{-\psi} g^{-\xi} h^{\rho} \bmod n \ \wedge\ U_1 = F^{\tau} \bmod P \wedge$$
$$U_2 = G^{\tau+\xi} \bmod P \ \wedge\ U_3 = H^{\tau+\varepsilon} \bmod P \ \wedge\ U_4 = U_1^{\psi} \bmod P \wedge$$
$$\varepsilon \in \{-2^{\ell_e+\ell_c+\ell_s}, +2^{\ell_e+\ell_c+\ell_s}\} \ \wedge\ \psi, \xi \in \{-2^{\ell_Q+\ell_c+\ell_s}, 2^{\ell_Q+\ell_c+\ell_s}\}\}(m) \ ,$$

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

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

Compute a challenge $c = \mathrm{hash}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$ 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, σ)**:** 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

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

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

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.

Figure 3: Group Signature with Full Revocation.

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 [KTY04]'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 [BMW03] definition of security of a group signature scheme, let us see 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.*

Sketch of proof. We set the group signature scheme up with just two identities i_0, i_1. The secret keys corresponding to these two members is published, they are the public key of the cryptosystem. To encrypt a bit b, we use sk_{i_b} to sign the message $m = 0$. The group manager's key *gmsk* corresponds to the secret key of the cryptosystem. With *gmsk*, it is possible to open the signature to see whether it was signed with sk_{i_0} or sk_{i_1}. This means that the bit b can be recovered.

Let us now restrict ourselves to adversaries that output $(i_0, i_1, 0)$ in the first phase. Then the definition of full-anonymity corresponds exactly to the definition of IND-CCA2 security. □

[AdM03] speculate whether it is possible to construct a group signature scheme based only on one-way functions. Following [IR89] 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 [BMW03].

8 A Group Signature Scheme Based on One-Way Functions and NIZK Arguments

We will present a group signature scheme with full-traceability and anonymity based on the assumptions that one-way functions exist and that non-interactive zero-knowledge arguments exist.

Recall that one-way functions imply the existence of pseudorandom functions, signature schemes secure against existential forgery under adaptive chosen message attack and statistically binding commitment to any string. As shown in [DIO98] the statistically binding commitment scheme based on one-way functions from [Nao91] can be made non-interactive.

The central idea in the group signature scheme we are going to present is that a member can demonstrate membership by producing a signature on the message under an authorized verification key that is part of the public key.

There is the question how the group manager will be able to tell the members apart. We let the public key contain a commitment to a seed s_i for a pseudorandom function. By evaluating this function on a randomly chosen element r, the member allows the group manager to identify him. On the other hand, nobody else can

tell the pseudorandom string apart from a random string, and therefore cannot tell who signed the message. To force the member to use a correct seed we require him to produce a NIZK argument of correctness.

The protocol is described in Figure 4. For simplicity and to facilitate comparison with the [BMW03] definition we do not include the join protocol in this description.

Group Signature Based on One-Way Functions and NIZK Arguments

KeyGen: Generate signature keys (vk_{GM}, sk_{GM}). Let pk be a public key for an unconditionally binding commitment scheme. Select at random a seed s_{GM} and let c_{GM} be a commitment to s_{GM} using randomness r_{GM}.

For each member generate signature keys (vk_i, msk_i). Select also a seed s_i for the pseudorandom function. Let c_i be a commitment to s_i using randomness r_i.

The public key is $vk = (pk, c_{GM}, vk_{GM}, c_1, vk_1, \ldots, c_k, vk_k)$. The group manager's secret key is $gmsk = (vk, s_1, \ldots, s_k)$.

Member i's secret key is $sk_i = (vk, vk_i, msk_i, s_i, r_i)$.

Sign: The member wishing to sign message m does the following.

She generates a strong one-time signature key pair (vk_{OT}, sk_{OT}). She then forms $\sigma_{OT} = \mathrm{Sign}_{msk_i}(vk_{OT})$. She picks a random point r. She sets $e = \mathrm{PRF}_{s_i}(r) \oplus \sigma_{OT}$. Let p be a NIZK argument that e is indeed the pseudorandom function evaluated in r under a seed s_i in one of the commitments, XORed with a valid signature on vk_{OT} under key vk_i, or it is the pseudorandom function evaluated in r under seed s_{GM} XORed with vk_{OT} signed under key vk_{GM}. Having done all this she computes a one-time signature $\sigma_{all} = \mathrm{Sign}_{sk_{OT}}(m, vk_{OT}, r, e, p)$.

The signature consists of $\sigma = (vk_{OT}, r, e, p, \sigma_{all})$.

Verify: The verifier wishing to verify a signature $(vk_{OT}, r, e, p, \sigma_{all})$ on message m checks the signature σ_{all} and that the proof p is valid.

Open: The group manager, wishing to open a signature $(vk_{OT}, r, e, p, \sigma_{all})$ on message m first verifies the signature. Then he goes through the list of seeds s_i that he has given members computing $\mathrm{PRF}_{s_j}(r)$. When he reaches $j = i$ he can extract a signature σ_{OT} on vk_{OT}. Then he knows that i is the member and he may demonstrate this by revealing σ_{OT} together with the key vk_i.

Figure 4: Group signature based on any one-way function and any general NIZK argument

We will now argue the following theorem.

Theorem 4 *The group signature scheme described in Figure 4 has full-traceability and anonymity.*

Sketch of proof.
Full-traceability. The zero-knowledge argument implies that either a valid signature points to one of the members or e contains a signature under vk_{GM}. But if the latter is the case, then we would be able to forge signatures. Therefore, e must be such that it points to one of the members.

Suppose now that the adversary is able to generate a valid signature pointing to member i, without corrupting party i or querying (i, m) to the signing oracle. We will use the adversary to forge a signature under vk_i or vk_{GM}. First, since we are using a strong one-time signature scheme the adversary cannot reuse a public key vk_{OT} that it has received from $\mathrm{Sign}(sk_\cdot, \cdot)$. By the NIZK argument p we have $e = \mathrm{PRF}_{s_i}(r) \oplus \sigma_{OT}$, where σ_{OT} is a signature under vk_i or vk_{GM}. However, with the knowledge of s_i and s we may then extract this signature and therefore we have made a forgery.

Anonymity. Imagine the two experiments defining anonymity where we use respectively sk_{i_0} or sk_{i_1} to generate the signature are distinguishable. In that case, the two experiments where we first guess i_0, i_1 at random, then run the two experiments but only output d if we guessed correctly and the adversary did not query m, σ to Open or i_0, i_1 to Corrupt, are also distinguishable. Now, set these two experiments up such that for i_0 and i_1 we simulate the NIZK proofs when making signatures. By the zero-knowledge property of the NIZK arguments, this means that we still have two distinguishable experiments. Imagine further, that we form c_{i_0} and c_{i_1} as commitments to 0 but still use s_{i_0} and s_{i_1} when making signatures. By the hiding property of the commitment scheme, the two experiments are again indistinguishable. Now, modify further the experiments such that we use $e = \mathrm{PRF}_s(r) \oplus \mathrm{Sign}_{sk_{GM}}(vk_{OT})$ whenever Sign outputs a signature. Since PRF is a pseudorandom function, the two experiments are still indistinguishable. Finally, instead of simulating proofs make instead proofs using vk and s. Now, the two experiments are both distinguishable and identical, a clear contradiction. □

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 Figure 4 can easily be extended to a traceable signature scheme [KTY04]. Theorems 3 and 4 can then be seen as indications that group signatures require stronger assumptions than traceable signature schemes.

9 Conclusion

We have contributed in two directions. On the practical side, we have suggested a new group signature scheme that is efficient and can be extended to support revocation and full revocation. Of a more theoretical nature, we have noted that full-anonymity may require stronger assumptions than what is needed to achieve anonymity.

This leaves an interesting open problem of suggesting a group signature scheme that supports full revocation but at the same time has full-anonymity.

While we have shown that in the static model our group signature realizes security model of [BMW03], but for the dynamic case we gave only sketches of security proofs in this extended abstract.

References

[ACJT00] Guiseppe 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.

[AdM03] 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`.

[AST02] Giuseppe Ateniese, Dawn Xiaodong Song, and Gene Tsudik. Quasi-efficient revocation in group signatures. In *proceedings of Financial Cryptography '02*, pages 183–197, 2002.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures using strong diffie hellman. In *Advances in Cryptology — CRYPTO 2004 (to appear)*, LNCS. Springer Verlag, 2004.

[BMW03] 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.

[CL02a] 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.

[CL02b] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *SCN '02, LNCS series, volume 2576*, pages 268–289, 2002.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Advances in Cryptology — CRYPTO 2004 (to appear)*, LNCS. Springer Verlag, 2004.

[CM98] 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.

[CP95] 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.

[CS] Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. Full version of [CS03]. Available from IACR eprint archive.

[CS97] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In Burt Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, volume 1296 of *LNCS*, pages 410–424. Springer Verlag, 1997.

[CS98] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. In *proceedings of CRYPTO '98, LNCS series, volume 1462*, pages 13–25, 1998. Full paper available at `http://eprint.iacr.org/2001/108`.

[CS03] Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. In Dan Boneh, editor, *Advances in Cryptology — CRYPTO 2003*, volume 2729 of *LNCS*, pages 126–144, 2003.

[CvH91] David Chaum and Eugène van Heyst. Group signatures. In *proceedings of EUROCRYPT '91, LNCS series, volume 547*, pages 257–265, 1991.

[DIO98] Giovanni Di Crescenzo, Yvail Ishai, and Rafail Ostrovsky. Non-interactive and non-malleable commitment. In *proceedings of STOC '98)*, pages 141–150, 1998.

[FS86] 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.

[GMR88] 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.

[GR03] 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.

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *proceedings of STOC '89*, pages 44–61, 1989.

[KTY04] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable signatures. Cryptology ePrint Archive, Report 2004/007, 2004. `http://eprint.iacr.org/`.

[LV01] Arjen K. Lenstra and Eric K. Verheul. Selecting cryptographic key sizes. *Journal of Cryptology*, 14(4):255–293, 2001.

[Lys02] Anna Lysyanskaya. *Signature schemes and applications to cryptographic protocol design*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, September 2002.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *proceedings of STOC '90*, pages 427–437, 1990.

[PS96] 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.

[Sah01] Amit Sahai. Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. In *proceedings of FOCS '01*, pages 543–553, 2001.

[Tru03] Trusted Computing Group. TCG TPM specification 1.2. Available at `www.trustedcomputinggroup.org`, 2003.

[TX03] Gene Tsudik and Shouhuai Xu. Accumulating composites and improved group signing. In *proceedings of ASIACRYPT '03, LNCS series, volume 2894*, pages 269–286, 2003.

A Security Proof of Basic Group Signature Scheme

Lemma 5 *Let $n = pq$, where $p = 2p' + 1$ and $q = 2q' + 1$ are safe primes. We assume the strong RSA assumption holds for $\mathbb{Z}_n^*$. Let $s, t \in \mathbb{Z}_n^*$ and $d > 1$ be produced by an adversary $\mathcal{A}$ such that $s^d = t^d \bmod n$. If d is odd we have $s = t$ and if d is even we have $s = \pm t$.*

Proof. Assume to start with that d is odd. Then we have $(s/t)^d = 1 \bmod n$. If $p'q'|d$ then we can break the strong RSA assumption, since for any h we have $h = h^{2d+1} \bmod n$. If $p'|d$ and $q' \not| d$ then $\gcd(n, 2^{2d} - 1) = p$ and we have factored n. Likewise we get a non-trivial factorization of n if $p' \not| d$ and $q'|d$. Finally, if $p' \not| d$ and $q' \not| d$ then we are looking at a normal RSA-exponentiation and therefore $s = t$.

If d is even, then we can use the theorem to take care of the odd part. Assume therefore WLOG that $d = 2^e$. Since $\gcd(2, p'q') = 1$ we get $(s/t)^{2^e} = 1 \bmod n$ implies $(s/t)^2 = 1 \bmod n$. Then $(s/t)^2 - 1 = ((s/d) - 1)((s/d) + 1) \bmod n$ gives us $s = \pm t$ or a non-trivial factorization of n. □

Lemma 6 *Let $n = pq$, where $p = 2p' + 1$ and $q = 2q' + 1$. We will assume the strong RSA assumption holds. Pick $g_1, \ldots, g_l$ at random from QR_n. Let an adversary produce $u \in \mathbb{Z}_n^*$ and $x, x_1, \ldots, x_l$ such that $u^x = g_1^{x_1} \cdots g_l^{x_l} \bmod n$. Then with overwhelming probability $x \mid x_1, \ldots, x \mid x_l$. In particular, if $x = 0$ then $x_1 = x_2 = \ldots = x_l = 0$*

We refer to Camenisch and Shoup for the proof of this Lemma [CS].

Lemma 7 *The basic group signature scheme has full-traceability.*

Proof. In any group signature the pair (U_1, U_2) defines a unique x such that $G^{\frac{P-1}{Q}x} = (U_2 U_1^{-X_G})^{\frac{P-1}{Q}} \bmod P$, where X_G is such that $G = F^{X_G} \bmod P$. A group signature contains an argument of knowledge of this x and an argument of knowledge of a signature on x.

We will first argue in Claim 7.1 that we can use the group manager's secret key to make perfect simulations of group signatures without knowing the x_i's. This means that the signing oracle does not reveal anything about the x_i belonging to a member.

Next, we will argue in Claim 7.2 that if an adversary successfully produces a group signature on a new message m not queried before, and this signature has (U_1, U_2) encrypting Y_i, then this implies knowledge of x_i. This means that we can use $\mathcal{A}$ in an algorithm to break the DDH problem in $\langle F \rangle$.

Finally, we will argue in Claim 7.3 that any valid signature must contain (U_1, U_2) pointing to one of the x_i's.

Claim 7.1 *Given Y_i $(= G^{x_i} \bmod P)$, e_i, and y_i we can make a perfect simulation of a group signature for member i.*

Proof. Pick $r \in \{0,1\}^{\ell_n/2}$ at random from and set $u = h^r y_i \bmod n$. Choose $R \in \mathbb{Z}_Q$ at random and set $U_1 = F^R \bmod P, U_2 = G^R Y_i \bmod P, U_3 = H^{R+e_i} \bmod P$.

Choose $c \in \{0,1\}^{\ell_c}$ at random and choose $z_x \in \{0,1\}^{\ell_Q+\ell_c+\ell_s}$, $z_r \in \{0,1\}^{\ell_n/2+\ell_c+\ell_s}$, $z_e \in \{0,1\}^{\ell_e+\ell_c+\ell_s}$, and $Z_R \in \mathbb{Z}_Q$ at random. Set $v = a^{-c} g^{-z_x} h^{z_r} u^{c2^{\ell_E}+z_e} \bmod n$ $V_1 = U_1^{-c} F^{Z_R} \bmod P$, $V_2 = U_2^{-c} G^{Z_R+z_x} \bmod P$, and $V_3 = U_3^{-c} H^{Z_R+z_e} \bmod P$

Define the random oracle to output c on query $(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$.

Claim 7.2 *If the x_i of an uncorrupted member is inside the group signature, then the group signature contains a proof of knowledge of x_i.*

Suppose $(U_2 U_1^{-X_G})^{\frac{P-1}{Q}} = G^{\frac{P-1}{Q}x_i} \bmod P$. In a valid signature we have, among other things, $V_1 = F^{Z_R} U_1^{-c} \bmod P$ and $V_2 = G^{Z_R+z_x} U_2^{-c} \bmod P$.

The c matches that of the hash-function evaluated in, among other things, U_1, U_2, V_1, V_2. Modeling the hash-function as a random oracle the adversary can only have negligible chance of guessing this, so we can assume that the adversary actually at some point made a query containing U_1, U_2, V_1, V_2. As an answer to this query, it received a random challenge c and was able to produce Z_R and z_x. To have noticeable chance of success it should also have had noticeable chance of answering another challenge c', i.e., produce satisfactory Z'_R, z'_x such that $V_1 = F^{Z'_R} U_1^{-c'} \bmod P$ and $V_2 = G^{Z'_R+z'_x} U_2^{-c'} \bmod P$.

We deduce that $U_1^{c-c'} = F^{Z_R - Z'_R} \bmod P$ and $U_2^{c-c'} = G^{Z_R - Z'_R} G^{z_x - z'_x} \bmod P$. This implies that $((U_2 U_1^{-X_G})^{\frac{P-1}{Q}})^{c-c'} = G^{\frac{P-1}{Q}(z_x - z'_x)} \bmod P$. So, if (U_1, U_2) point to x_i, then we must have $x_i = \frac{z_x - z'_x}{c-c'} \bmod Q$. Thus, we have successfully computed the discrete logarithm x_i of $Y_i \bmod P$.

Claim 7.3 *A valid group signature contains one of the G^{x_i}'s inside (U_1, U_2).*

Proof. Define x as the element from $\mathbb{Z}_Q$ such that $(U_2 U_1^{-X_G})^{\frac{P-1}{Q}} = G^{\frac{P-1}{Q}x} \bmod P$. We will show that the strong RSA assumption implies that a valid group signature with x not being one of the x_i's implies that the adversary is capable of an existential forgery attack on the CL-signature.

First, we have to make sure that the key generation algorithm does not reveal anything about the factorization of n, otherwise we could not use the strong RSA assumption in the proof.

We choose $a_{\text{base}}, g_{\text{base}}, h_{\text{base}}$ at random from QR_n, and choose $E_1, \ldots, E_k$ as random primes with the correct distribution. We may now compute $a = a_{\text{base}}^{\prod_{j=1}^{k} E_j} \bmod n, g = g_{\text{base}}^{\prod_{j=1}^{k} E_j} \bmod n, h = h_{\text{base}}^{\prod_{j=1}^{k} E_j} \bmod$

n. This means that we do not need the factorization of n to produce the signatures (y_i, E_i, r_i) on the x_i that we give to the members.

Consider now $\mathcal{A}$ trying to form a valid signature that does not point to one of the members. Define $z_E := c2^{\ell_E} + z_e$. From correct proofs to two different challenges c, c' we have $v = a^{-c} g^{-z_x} h^{z_r} u^{z_E} \bmod n$ and $v = a^{-c'} g^{-z'_x} h^{z'_r} u^{z'_E} \bmod n$. This implies that $u^{z_E - z'_E} = a_{\text{base}}^{(c-c')\prod_{j=1}^{k} E_j} g_{\text{base}}^{(z_x - z'_x)\prod_{j=1}^{k} E_j} h_{\text{base}}^{(z'_r - z_r)\prod_{j=1}^{k} E_j} \bmod n$. By Lemma 6, this implies that $(z_E - z'_E) \mid (c - c')\prod_{j=1}^{k} E_j$ and thus $(c - c')2^{\ell_E} + (z_e - z'_e) \mid (c - c')\prod_{j=1}^{k} E_i$. Considering the sizes of $z_E - z'_E, E_1, \ldots, E_k$, and $c - c'$ we get that $(c - c') \mid (z_e - z'_e)$ and hence $(z_E - z'_E) = \pm E_i(c - c')$ for some E_i. From Lemma 6 we also get $(c - c') \mid (z_x - z'_x)$ and $(c - c') \mid (z_r - z'_r)$. We define $x = \frac{z_x - z'_x}{c - c'}$ and $r = \frac{z'_r - z_r}{c - c'}$. Removing the $c - c'$ factor in the exponent we get $(\pm u^{\pm 1})^{E_i} = \pm u^{\pm E_i} = a g^x h^r \bmod n$, which is a CL-signature on x.

In a real execution we do not set up the protocol with known E_i-roots of a, g, h. Rather we just reveal a set of signatures. What we have shown above is just that the adversary will generate a CL-signature on x using one of the E_i's belonging to a member. This means that we have an existential forgery on the CL-signature scheme unless $x = x_i \bmod E_i$ for some i. Considering the sizes of z_x, z'_x relative to E_i we then have $x = x_i$ for some E_i.

At the same time we have already seen in Claim 7.2 that a signature contains a proof of knowledge of x such that $(U_2 U_1^{-X_G})^{\frac{P-1}{Q}} = G^{\frac{P-1}{Q}x} \bmod P$, where $x = \frac{z_x - z'_x}{c - c'} \bmod Q$. Since $\frac{z_x - z'_x}{c - c'} = x_i$ we therefore deduce that a valid group signature does indeed point out a member $i \in [k]$. □

It is worth noting that Claim 7.1 and Claim 7.2 hold even if we know the factorization of n. This matters when we consider scenarios where the group manager generates n and therefore may generate it maliciously and with knowledge of a lot of extra information about it.

Lemma 8 *The basic group signature scheme has full-anonymity.*

Proof. The intuitive reason that we have full-anonymity is that since everything is encrypted and proved in zero-knowledge, a signature does not reveal anything about the signer's secret key. The obstacle is that the adversary can ask the group manager to open messages. Essentially this gives a chosen ciphertext attack on the cryptosystem used to hide the identities. We will argue that a signature actually is a CCA2 secure encryption of the identity of the signer.

A valid group signature actually contains two encryptions of the member's identity. We have $(U_1, U_2) = (F^R \bmod P, G^R Y_i \bmod P)$ and also $(U_1, U_3) = (F^R \bmod P, H^R H^{e_i} \bmod P)$. This means we have two ElGamal encryptions of messages that will allow the group manager to identify the signer. In addition to that, the proof of Claim 7.3 shows that a group signature actually contains a proof of knowledge that (U_1, U_2) and (U_1, U_3) point to the same member. We therefore have a setup of a CCA2 secure cryptosystem much like in [NY90, CS98, Sah01].

Let us start with the expectation of $\mathbf{Exp}_{\mathcal{A}}^{\text{f-anon}}(1, k)$. We may set up a, g, h as in Claim 7.3, such that we do not reveal the factorization of n. By Claim 7.1, we may simulate the group signature when generating the challenge group signature by choosing the challenge in an appropriate way.

Since we are simulating the proof, we may form (U_1, U_3) as a ciphertext encrypting $H^{e_{i_0}} \bmod P$ instead of $H^{e_{i_1}} \bmod P$. If $\mathcal{A}$ can notice the difference, then it means that it can break the semantic security of the ElGamal encryption.

The next step we take is to switch the key we are using to decrypt. We set up the signature scheme with knowledge of X_H instead of X_G, and use decryption of (U_1, U_3) to find out who signed the message. Since a proof that (U_1, U_2) and (U_1, U_3) point to the same member is included in the signatures $\mathcal{A}$ queries to $\text{Open}(\textit{gmsk}, \cdot, \cdot)$ this oracle answers the same on all queries where $(u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$ is not the same as in the challenge signature. In other words, $\mathcal{A}$ does not learn from the openings whether it is X_G or X_H that we use to decrypt.

The only thing we have to guard against is that $\mathcal{A}$ reuses $(m, u, v, U_1, U_2, U_3, V_1, V_2, V_3)$ from the challenge signature. Call the answers used in the simulated signatures for z_x, z_r, z_e, Z_r, and let the answers produced by $\mathcal{A}$ be z'_x, z'_r, z'_e, Z'_r.

Since $U_1^c V_1 = F^{Z_R} \bmod P$ we must have that $Z'_R = Z_R$. $U_3^c V_3 H^{-Z_R} = H^{z_r} = H^{z'_r} \bmod P$ implies that $z_e = z'_e \bmod Q$. Since z_e and z'_e are smaller than Q this shows that $z_e = z'_e$.

The equation $v = a^{-c} g^{-z_x} h^{z_r} u^{c2^{\ell_E} + z_e} = a^c g^{z'_x} h^{z'_r} u^{c2^{\ell_E} + z_e} \bmod n$ shows that $1 = g^{z'_x - z_x} h^{z'_r - z_r} \bmod n$. By Lemma 6 we then have $z_x = z'_x$ and $z_r = z'_r$.

Overall, we therefore see that the adversary cannot recycle $(u, v, U_1, U_2, U_3, V_1, V_2, V_3, m)$ without recycling the entire challenge group signature. Therefore, the adversary cannot tell the difference between using X_G and X_H to decrypt.

We now have a hybrid signature with simulated proof and $G^{x_{i_1}} \bmod P$ in (U_1, U_2) and $H^{e_{i_0}} \bmod P$ in (U_1, U_3) and we use X_H to reveal the identity of parties. We now consider the experiment where we use $U_2 = G^{R + x_{i_0}} \bmod P$. By the semantic security of ElGamal encryption, this is not something that $\mathcal{A}$ will notice. The experiment has now been modified so much that it is simply $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{f-anon}}(0, k)$ with simulated proof. □