Efficient Group Signatures without Trapdoors

Giuseppe Ateniese & Breno de Medeiros
Johns Hopkins University
Department of Computer Science
Baltimore, MD 21218, USA
{ ateniese, breno }@cs.jhu.edu

Working Draft, November 12, 2002

Abstract

Group signature schemes are fundamental cryptographic tools that enable unlinkably anonymous authentication, in the same fashion that digital signatures provide the basis for strong authentication protocols. In this paper we present the first group signature scheme with constant-size parameters that does not employ any trapdoor function. This novel type of group signature scheme allows public parameters to be shared among organizations. Such sharing represents a highly desirable simplification over existing schemes, which require each organization to maintain a separate cryptographic domain.

1 Introduction

Group signatures allow group members to anonymously sign arbitrary messages on behalf of the group. In addition, signatures generated from the same signer are unlinkable, i.e., it is difficult to determine whether two or more signatures were generated by the same group member. In case of dispute, a group manager will be able to *open* a signature and incontestably show the identity of the signer. At the same time, no one (including the group manager) will be able to falsely accuse any other member of the group.

Group signatures were introduced by D. Chaum and E. van Heyst [16] in 1991. That was followed by several other works, but only relatively recent ones [4, 10, 11] have group public keys and group signatures with sizes that do not depend on the number of group members. The scheme in [4] is the most efficient one and the only proven secure against an adaptive adversary.

Traditional signature schemes, such as DSA or RSA, can be used to build complex systems where both strong authentication and non-repudiation are needed. Analogously, group signatures could be used as a cryptographic primitive to build more sophisticated systems where privacy is of paramount importance such as digital payment schemes or credential transfer systems. However, all the existing group signature schemes providing constant-size parameters work in an RSA-type setting and require the group manager to know the factors of an RSA modulus. Sharing these factors would compromise the security and/or the trust assumptions of the entire scheme.

Previous work on trapdoor-free schemes have public parameters which increase linearly in size with the number of group members. In this paper, we introduce the first trapdoor-free group signature scheme with constant-size group public key and group signature. (Notice that, while in theory one always needs at least $\log n$ bits to uniquely identify n different users in any system,

in practice $\log n$ is orders of magnitude smaller than the bit length of keys used in public key cryptography.)

Our scheme is useful when several distinct groups or organizations must interact and exchange information about individuals while protecting the individuals' privacy. Credential transfer systems (CTS) [14, 15, 19, 17, 23, 9] are examples of such environments that can be built via group signature schemes [9, 3]. Real-world scenarios for the use of CTS include the health-care industry, electronic voting, and transportation systems. In such cases, the added manageability and improved optimization opportunities permitted by the use of a single cryptographic domain for all participating organizations may outweigh other efficiency considerations. In typical CTS, users authenticate themselves with anonymous credentials, protecting their privacy while exercising their right to vote, obtaining health services or renting a GPS-tracked automobile. The efficiency of a single signature generation or verification is measured in the human time scale. Consequently, theoretical computational advantages become less important, and instead the administrative complexity and related costs are likely to be the overwhelming concern of implementers. In these situations a trapdoor-free scheme has a definite advantage. The system proposed in this article is the first efficient, trapdoor-free group signature scheme.

1.1 Organization of this paper

The following section contains the definition of group signatures and the attending security requirements. In section §3 we give a high-level, intuitive description of our proposed scheme, and place it in the context of previous work. Section §4 is devoted to cryptographic primitives and building blocks which we use in a specific construction of our scheme which takes all of section §5. Section §6 explains how the system can be adapted to other groups. Section §7 of this paper lists the assumptions we make and contains a security analysis of the specific scheme, followed by some concluding observations.

2 Definition

In this section we present our characterization of group signature schemes. In general, a group signature scheme is defined by a family of procedures:

`SETUP`: A probabilistic algorithm that each group manager (entity responsible for establishing and managing a group signature scheme) must run once to generate the group-specific public and secret parameters. The input to `SETUP` is the set of public parameters, which includes a security parameter, and its output are the group public key $\mathcal{P}$ and associated secret key $\mathcal{S}$.

`JOIN`: A prospective member executes this protocol (interacting with the group manager) to join the group. The new member's output is a membership certificate and the corresponding secret.

`SIGN`: A probabilistic algorithm that outputs a group signature when given as input a message, the group public key, a membership certificate, and the associated membership secret.

`VERIFY`: A boolean-valued algorithm used to test the authenticity of signatures generated by `SIGN`.

`OPEN`: An algorithm that given as input a message, a group signature on it, and the group secret key, extracts the membership certificate used to issue the signature, and a non-interactive proof of the signature's authorship.

A group signature scheme must satisfy the following properties:

Correctness: A properly formed group signature must be accepted by the verification algorithm.

Unforgeability: Without possession of a membership certificate and knowledge of associated secret it is computationally infeasible to produce a signature that is accepted by the verification algorithm.

Anonymity/ Unlinkability: Given a group signature on a message, it is computationally infeasible to determine which member generated the signature. Moreover, given several group signatures on the same or different messages it is computationally infeasible to decide whether the signatures were issued by the same or by different group members.

Exculpability: A signature produced by a group member cannot be successfully attributed to another, and the group manager cannot generate signatures on behalf of other group members (non-framing).

Traceability: The group manager is "always" (with overwhelming probability) able to open a valid signature and determine which member signed it. Even if a coalition of group members collaborates to produce a signature on a message, possibly by combining their certificate secrets in some fashion, the group manager will succeed in attributing the signature to one of the colluding members (coalition-resistance).

In the literature it is usually only stated that the group manager should be able to determine the authorship of signatures. We prefer the stronger requirement that the OPEN protocol produce a publicly verifiable proof of the authorship. This stronger requirement is satisfied in practice by all proposed group signature schemes, and it underlies the non-framing property.

The requirements of unforgeability and coalition-resistance are equivalent to the requirements that group membership certificates be unforgeable under passive and active attacks, respectively, and only issuable by the group manager. In other words, a membership certificate must contain the equivalent of a digital signature by the group manager. Similarly, the requirements of traceability and exculpability imply that the group signature must hide a regular digital signature issued by the member.

3 The General Scheme

In the group authentication problem a holder U of a group certificate interacts with a verifier V to prove his status as a group member without revealing his certificate. If the interactive protocol can be made non-interactive through the Fiat-Shamir heuristic ([20]), then the resulting algorithm will be similar to the issuing of a group signature, except that U's identity may be unrecoverable from the signature alone. The issuing of a group signature requires, in addition to a proof of membership, that U *verifiably encrypts* some information about his certificate under the group manager's public key. U must provide the verifier with an encrypted token and prove to V that the group manager is able to decrypt the token to reveal U's authorship of the signature.

A group signature can be seen as a proof of knowledge of a group certificate which provides evidence of membership. The group certificate can be generated only by the group manager GM and should be difficult to forge. In addition, it has to contain some secret information generated by the group member and unknown to GM to avoid framing attacks in which GM signs on behalf of other members.

A Nyberg-Rueppel signature is an ElGamal-type signature that provides message recovery. This implies that the verifier does not have to compute any hash function in order to verify a signature.

This feature is very useful if we want to make the verification process unlinkable and so that the verifier does not learn anything about the signature itself. Generally, signatures with message recovery can be more effectively used in privacy protocols providing unlinkability. Indeed, the difficult task of randomizing a standard hash function can be avoided in such cases.

We now describe our scheme in a general arithmetic group $\mathcal{G}$. In section §5, we outline the specifics of the protocol in a suitable group, namely the subgroup of quadratic residues modulo a prime p, where p is simultaneously a *safe* prime, i.e, $p = 2q + 1$, with q also prime, and a *Sophie Germain* prime, that is the number $\hat{p} = 2p + 1$ is prime. There are other choices for the group $\mathcal{G}$, and we describe a variant in section §6. However, not all settings where the Nyberg-Rueppel signature makes sense have the characteristics needed by our scheme. We refer the reader to the security analysis (section §7) for the attending security assumptions.

Let $\mathcal{G}$ be a suitable group, and let q equal the order $|\mathcal{G}|$ of the group. Let g be a fixed, public generator for $\mathcal{G}$, and let $y = g^x$ be the public key of a party. A Nyberg-Rueppel signature on a message $m \in \mathcal{G}$ under the public key y is a pair (r, s), where r is a member of the group $\mathcal{G}$, and s is a non-zero integer, such that:

$$r = mg^k \quad , \quad s \equiv -\rho(r)x - k \mod q, \tag{1}$$

where $\rho(\cdot)$ is some integer-valued function defined on $\mathcal{G}$, called the *reduction function*. In both constructions we give in this paper, the group $\mathcal{G}$ itself is composed of integers modulo some number, and in this case it is safe to take the reduction function to be the identity.[1] We note that if the order of $\mathcal{G}$ is unknown, then s is not restricted to the interval $[1, q - 1]$.

To verify the signature, it is sufficient to check that the following equation holds in $\mathcal{G}$:

$$g^s y^{\rho(r)} r = m. \tag{2}$$

The basic idea is to define group-specific parameters to include a group public key y and a group identifier message $m = \mathcal{M}$, and define member certificates to be signature pairs (r, s) as above. These are issuable only by the group manager GM, who is privy to the secret key x associated to y. However, such a certificate would not work because no information is embedded into it that can be linked to a specific member, certainly not in a non-repudiatable manner. To solve this, we introduce more public group parameters: a second generator g_1 of $\mathcal{G}$ and also $y_1 = g_1^x$. In order to join the group, a user would generate a secret value u and compute $I = g_1^u$ and $J = y_1^u = I^x$. Once the prospective new member has proved knowledge of u given I, GM releases the signature:

$$r = \mathcal{M} I^k g^k \quad , \quad s = -\rho(r)x - k. \tag{3}$$

To verify the signature, it is sufficient to check whether:

$$(gI)^s (yJ)^{\rho(r)} r = \mathcal{M}. \tag{4}$$

At this point the group member signs a statement in which he certifies that (r, s) is his own certificate. Such a signature is stored by GM on a database and will be used to prove authorship of a group signature during the opening phase.

Notice that the congruence (4) is a proof of equality of discrete logarithms, i.e., a proof that the logarithm of y in the basis g equals that of J to basis I, or $\mathrm{Dlog}_g y = \mathrm{Dlog}_I J$. Such a proof

[1] If more general groups are used to realize the scheme, the choice of the reduction function may impact the security of the signature scheme. For ECDSA, it has been pointed out that the standard-specified reduction function makes the signature scheme malleable [26].

is necessary to show to a verifier that J indeed contains GM's secret x in the exponent. The correctness of this proof comes from the unfeasibility of finding two values w_1 and w_2 such that $s = \rho(r)x + w_1 = \rho(r)z + w_2$, for $x \neq z$, where r is a function of both w_1 and w_2, i.e., $r = \mathcal{M} I^{w_1} g^{w_2}$.

When generating a group signature, the member can prove knowledge of (r, s) and, in addition, prove knowledge of $u = \mathrm{Dlog}_{g_1} I$ in order to avoid framing attacks.

Although the new pair (r, s) defined in (3) is a valid group certificate, there are still two open problems left: The first is to prove knowledge of the group certificate without showing r and s so that any two executions of the protocol are unlinkable. The second problem is to allow GM to *open* the proof and show the identity of the group member.

Both problems can be solved by employing a *verifiable encryption* of digital signature schemes. Indeed, notice that the pair (r, s) in (3) can be seen as a traditional Nyberg-Rueppel signature under the public key $Jy = (Ig)^x$. Unlinkability is not a requirement of verifiable encryption therefore existing protocols for ElGamal-type signature schemes do not provide it. Hence, it would be possible to link two or more verifiable encryptions, which is equivalent to linking two or more group signatures from the same signer. This is because, in existing schemes, the first value r of the signature pair (r, s) is revealed and the actual protocol is applied only to the second value s reducing then the problem of verifiable encryption of a digital signature to the simpler problem of verifiably encrypting a discrete logarithm (see [8, 1, 22, 2] for details).

To solve this issue, we could use a secret exponent and protect in zero-knowledge the signature pair (r, s) along with other parameters required by our scheme. In particular, we could raise both terms of the congruence (4) to a randomly generated secret value v, thus achieving:

$$(gI)^{sv}(yJ)^{\rho(r)v} r^v = \mathcal{M}^v. \tag{5}$$

If we denote $(gI)^v$ by W and $(yJ)^v$ by X, we can rewrite the previous equation as:

$$W^s X^{\rho(r)} r^v = \mathcal{M}^v. \tag{6}$$

Remember that the scheme defines $\mathcal{M}$, g, g_1, y and y_1 as public, fixed parameters. The proof of knowledge of a group membership certificate starts with the release of the values W, X, g_1^v and y_1^v. The next step is to prove that W is well formed, through execution of a proof of knowledge of a representation of W with respect to the basis $\{g, g_1^v\}$, where the exponent of g in this representation equals $v = \mathrm{Dlog}_{g_1} g_1^v$. Similarly, X is shown to be correctly formed by repeating this proof with y substituted for g, and y_1 substituted for g_1.

To proceed, we must overcome a difficulty with equation (6), namely verifying the well-formedness of the term r^v, without revealing the basis r. The solution involves revealing an encryption of r. The encryption also provides a method for GM to open the transaction and reveal r, which uniquely identifies the group member who issued the given signature. The scheme specifies yet another generator $\bar{g}$ of the group $\mathcal{G}$, and a public key $\bar{y}$ for GM, used for ElGamal encryption purposes. The next step of the proof of knowledge of a membership certificate is to release:

$$Z = W^s X^{\rho(r)} = r^{-v}\mathcal{M}^v \ , \ A = r^{-1}\bar{y}^a \ , \ B = \bar{g}^a. \tag{7}$$

Z is shown to be well-formed through a proof of knowledge of a representation of Z w.r.t. the basis $\{W, X\}$. Next, A^v and $y' = \bar{y}^{-v}$ are released (plus a proof that $\mathrm{Dlog}_A A^v = \mathrm{Dlog}_{\bar{y}} y'^{-1} = \mathrm{Dlog}_{g_1} g_1^v$), and the value $ZA^{-v} = \mathcal{M}^v \bar{y}^{-av}$ is shown to be well-formed, by proving knowledge of a representation of ZA^{-v} w.r.t. the basis $\{\mathcal{M}, y'\}$, where the exponent of $\mathcal{M}$ in ZA^{-v} equals the logarithm $\mathrm{Dlog}_A A^v$, and where the exponent of y' in ZA^{-v} equals the logarithm $\mathrm{Dlog}_g B$.

The whole signature equation has been verified, except that we have not checked the relation between the value $\rho(r)$ (the exponent of X in the representation of Z w.r.t to basis $\{W, X\}$) and the value encrypted in A. For simple reduction functions $\rho(\cdot)$ (in our construction, $\rho(\cdot)$ is simply the identity) this can be achieved by combining the technique of verifiable ElGamal encryption of a logarithm (Stadler, [25]) and the proof of equality of discrete logarithms in different groups, due to Boudot[6]. We have described the whole procedure as if composed of several steps. However, the various proofs of knowledge described can be made non-interactive through the Fiat-Shamir heuristic and combined together in a single step. We describe this in detail in Section §5.

4 Proofs of knowledge

In this paper we make use of several types of proofs of knowledge about various relations between secrets. All the proofs of knowledge discussed here have been presented elsewhere, so we make no claims of originality in this section. More importantly, all the proofs of knowledge listed in this section have been proved zero-knowledge in a statistical or computational sense within the random oracle model, under the Decisional Diffie-Hellman assumption, and the Strong RSA assumption, explained below.

We now introduce some notation.

Notation 1 (Groups and generators)

- *$\mathcal{J}$ stands for an arithmetic group, such as an RSA ring with composite modulus n or the group $\mathbf{Z}_p^*$ of non-zero (multiplicative) residues modulo p.*
- *g stands for an element of $\mathcal{J}$ of unknown composite order or known prime order. Let q be the order of g.*
- *Let κ be the smallest integer such that 2^κ is larger than q. We assume that κ is known, even if q is not.*
- *g generates the subgroup $\mathcal{G}$ of $\mathcal{J}$.*

Let $\mathcal{H}$ stand for a secure hash function which maps arbitrarily long bit-strings into bit-strings of fixed length τ.

Definition 1 (Decisional Diffie-Hellman assumption (DDH)) *Let $\mathcal{J}$ be a group and g an element of known prime, or unknown composite, order q in $\mathcal{J}$. Let $\mathcal{G} = \langle g \rangle$ be the subgroup generated by g in $\mathcal{J}$. The DDH assumption for $\mathcal{G}$ is then there is no efficient (randomized, probabilistic) algorithm that can distinguish between the two following distributions in $\mathcal{G}$:*

$$Dist_1 = \{(h, i, j), \text{ where } h, i, j \text{ are independently randomly distributed (i.r.d.) in } \mathcal{G}\}$$

and

$$Dist_2 = \{(h', i', j'), \text{ where } h' = g^x, i' = g^y, j' = g^{xy} \text{ for i.r.d. } x, y \text{ with } 0 \leq x, y < q\}$$

A triple of group elements such as (h', i', j') above is called a *Diffie-Hellman triple.* The DDH assumption is thus the statement that there is no efficient algorithm to distinguish between Diffie-Hellman triples and randomly generated triples.

Definition 2 (Strong RSA assumption (SRSA)) *Let $n = pq$ be a composite modulus, where p and q are two large primes. The strong RSA assumption states that there is no efficient (randomized, probabilistic) algorithm that, given as input n and an integer y, but not the factorization of n, can produce two other integers u and e, where $e > 1$ and $u^e \equiv y \mod n$.*

SRSA underlies the security of the proof of equality of logarithms in distinct groups (8).

In the following we use a similar notation as in definition (1), and introduce a second security parameter ϵ.

Definition 3 (Proof of knowledge of a discrete logarithm) *With $\mathcal{G}$ and g as above, U can prove to a verifier V his knowledge of an integer x in $\{0, \ldots, 2^\kappa - 1\}$, such that $h = g^x$, by releasing integers s and c, with s in $\{-2^{\epsilon(\tau+\kappa)+1}, \ldots, 2^{\epsilon(\tau+\kappa)+1} - 1\}$ and c in $\{0, \ldots, 2^\tau - 1\}$, such that*

$$c = \mathcal{H}(g \mid\mid h \mid\mid g^s h^c),$$

where the symbol $\mid\mid$ denotes string concatenation.

In order to compute the pair (s, c) above, U generates a random integer k in $\{-2^{\epsilon(\tau+\kappa)}, \ldots, 2^{\epsilon(\tau+\kappa)} - 1\}$ and sets $c = \mathcal{H}(g||h||g^k)$, and $s = k - cx$ (as integer). We denote it by (notation introduced in [11]) :

$$PK[x \,:\, h = g^x],$$

This proof of knowledge can be transformed into a digital signature, with x being the secret key associated with public key h. To sign an arbitrary bitstring m, we just have to change the computation of c as follows:

$$c = \mathcal{H}(g \mid\mid h \mid\mid g^s h^c \mid\mid m).$$

We denote this *signature of knowledge* ([11]) by:

$$SPK[x \,:\, h = g^x](m).$$

Returning to the notation in definition (3), if the order q of the group $\mathcal{G}$ is known, then operations on the exponents should be computed modulo q, and some statements about the size of parameters can be simplified. In the above we would substitute:

- $x \in \{0, \ldots, 2^\kappa - 1\}$ by $x \in \{0, \ldots, q - 1\}$,
- $s \in \{-2^{\epsilon(\tau+\kappa)+1}, \ldots, 2^{\epsilon(\tau+\kappa)+1} - 1\}$ by $s \in \{0, \ldots, q - 1\}$, and
- $s = k - cx$ (in $\mathbf{Z}$) by $s = k - cx \mod q$.

In the constructions below we assume the group order q is unknown; as above, it is straightforward to adapt them to the case of known order.

Definition 4 (Proof of knowledge of a common discrete logarithm) *U can prove to a verifier V his knowledge of an x (with $0 \leq x < 2^\kappa$) such that two lists $g_1, g_2, \ldots, g_\ell$ and $h_1, h_2, \ldots, h_\ell$ (of elements of $\mathcal{G}$) satisfy $h_i = g_i^x, i = 1 \ldots \ell$, by releasing s and c ($-2^{\epsilon(\tau+\kappa)+1} \leq s < 2^{\epsilon(\tau+\kappa)+1}$ and $0 \leq c < 2^\tau$) such that*

$$c = \mathcal{H}(g_1 \mid\mid \ldots \mid\mid g_\ell \mid\mid h_1 \mid\mid \ldots \mid\mid h_\ell \mid\mid (g_1 \ldots g_\ell)^s (h_1 \ldots h_\ell)^c).$$

Again, U computes $c = \mathcal{H}(g_1 \mid\mid \ldots \mid\mid g_\ell \mid\mid h_1 \mid\mid \ldots \mid\mid h_\ell \mid\mid (g_1 \ldots g_\ell)^k)$ for a randomly chosen k ($-2^{\epsilon(\tau+\kappa)} \leq k < 2^{\epsilon(\tau+\kappa)}$), and sets $s = k - cx$. This proof of knowledge can be combined with the previous type. We denote it by:

$$PK[x \,:\, h_1 = g_1^x \wedge \cdots \wedge h_\ell = g_\ell^x].$$

Definition 5 (Proof of knowledge of a representation) *U can prove to a verifier V his knowledge of elements $x_1, \ldots, x_\ell$ (with $0 \leq x_i < 2^\kappa$) such that a given element A satisfies $A = g_1^{x_1} \cdots g_\ell^{x_\ell}$, by releasing s_i and c ($-2^{\epsilon(\tau+\kappa)+1} \leq s_i < 2^{\epsilon(\tau+\kappa)+1}$ and $0 \leq c < 2^\tau$) such that*

$$c = \mathcal{H}(g_1 \mid\mid \ldots \mid\mid g_\ell \mid\mid A \mid\mid g_1^{s_1} \ldots g_\ell^{s_\ell} A^c).$$

Again, U computes $c = \mathcal{H}(g_1 \mid\mid \ldots \mid\mid g_\ell \mid\mid A \mid\mid g_1^{k_1} \ldots g_\ell^{k_\ell})$ for randomly chosen k_i ($-2^{\epsilon(\tau+\kappa)} \leq k_i < 2^{\epsilon(\tau+\kappa)}$), and sets $s_i = k_i - cx_i$. This proof of knowledge can be combined with both previous types. We denote it by:

$$PK[x_1, \ldots, x_\ell \,:\, A = g_1^{x_1} \cdots g_\ell^{x_\ell}].$$

Let g, h be two elements of $\mathcal{G}$. Assume that g and h are constructed in a provably random way, for instance as consecutive images of a secure pseudo-random generator. Generating g and h in such a way ensures that no one knows the discrete logarithm of g to basis h, or that of h to basis g.

Definition 6 (Commitment to a secret value) *Let x be a secret value held by U. Let g and h be two provably random generators of $\mathcal{G}$. We say that $E = E(x, r) = g^x h^r$ is a commitment to the value x in $\mathcal{G}$, where r is a randomly generated value, $0 < r < q$.*

If q is unknown, then one must choose r in a larger interval, say $-2^{\kappa+\tau+1} < r < 2^{\kappa+\tau+1}$, to ensure that all elements in the interval $[0, q-1]$ are sampled nearly uniformly. The commitment reveals nothing about r in a statistical sense.

Let $\mathcal{E}$ be a distinct arithmetic group of unknown composite order n, in which the DDH assumption also holds. For instance, $\mathcal{E}$ can be chosen as the subgroup of quadratic residues in an RSA ring. Let $g = g_1$, g_2, $h = h_1$, and h_2 be provably random generators of $\mathcal{E}$. We assume that the smallest integer λ such that $2^\lambda > n$ is known. Assume U has published two commitments, $E = E_1(x, r) = g_1^x h_1^{r_1}$ in $\mathcal{G}$, and a second commitment $E_2(x, r_2) = g_2^x h_2^{r_2}$.

Let δ, σ_1 and σ_2 be other security parameters. Assume further that $x < b$.

Definition 7 (Proof of knowledge of a committed value) *With notation as in the paragraph above, U can prove in zero-knowledge to a verifier V knowledge of a number x committed in the value $E = E(x, r) = g^x h^r$, by sending V a triple (c, D, D_1) satisfying:*

$$c = \mathcal{H}(g \mid\mid h \mid\mid E \mid\mid g^D h^{D_1} E^{-c} \mod n).$$

To prepare the proof, U generates random $t \in [1, 2^{\delta+\tau/2} b + 1]$ and $s \in [1, 2^{\delta+\tau/2+\sigma_1} n - 1]$; computes $W = g^t h^s \mod n$; computes $c = \mathcal{H}(g \mid\mid h \mid\mid E \mid\mid W)$; and finally computes $D = t + cx$, $D_1 = s + cr$ (in $\mathbf{Z}$).

Definition 8 (Proof of equality of two committed values) *U can prove in zero-knowledge to a verifier V that two commitments $E_1 = E_1(x, r_1)$ and $E_2 = E_2(x, r_2)$ hide the same exponent x, by sending V a quadruple (c, D, D_1, D_2) satisfying:*

$$c = \mathcal{H}(g_1 \mid\mid h_1 \mid\mid g_2 \mid\mid h_2 \mid\mid E_1 \mid\mid E_2 \mid\mid g_1^D h_1^{D_1} E_1^{-c} \mod n \mid\mid g_2^D h_2^{D_2} E_2^{-c} \mod n).$$

To prepare the proof, U generates random $t \in [1, 2^{\delta+\tau/2}b + 1]$, $s_1 \in [1, 2^{\delta+\tau/2+\sigma_1}n - 1]$, and $s_2 \in [1, 2^{\delta+\tau/2+\sigma_2}n - 1]$; computes $W_1 = g_1^t h_1^{s_1} \mod n$, and $W_2 = g_2^t h_2^{s_2} \mod n$; computes $c = \mathcal{H}(g_1 \mid\mid h_1 \mid\mid g_2 \mid\mid h_2 \mid\mid E_1 \mid\mid W_1 \mid\mid W_2)$; and finally computes $D = t + cx$, $D_1 = s_1 + cr_1$, $D_2 = s_2 + cr_2$ (in $\mathbf{Z}$).

We denote the above proof by

$$PK[x, r_1, r_2 : E_1 = E_1(x, r_1) \wedge E_2 = E_2(x, r_2)].$$

The next definition uses a simplified notation; call $g = g_1$, $h = h_1$ and $\sigma = \sigma_1$.

Definition 9 (Proof that a committed number is a square) *A prover U can convince a verifier V that the commitment $E = E(x^2, r_1) = g^{x^2} h^{r_1} \mod n$ ($r_1 \in [-2^\sigma n + 1, 2^\sigma n - 1]$) contains the square of a number known to U, by sending V the quintuple (F, c, D, D_1, D_2), where*

$$c = \mathcal{H}(g \mid\mid h \mid\mid E \mid\mid F \mid\mid F^D h^{D_1} E^{-c} \mod n \mid\mid g^D h^{D_2} F^{-c} \mod n).$$

Indeed, U generates a random r_2 in $[-2^\sigma n + 1, 2^\sigma n - 1]$, and sets $F = g^x h^{r_2}$. Notice now that U can rewrite E in the basis $\{F, h\}$ as $E(x, r^3) = F^x h^{r_3} \mod n$, where $r_3 = r_1 - r_2 x$, and $r_3 \in [-2^\sigma bn + 1, 2^\sigma bn - 1]$. It is enough then for U to use the previous proof of equality of the exponent x committed though $E_1 = F = E(x, r_2)$ and $E_2 = E = E(x, r_3)$, i.e., execute $PK[x, r_2, r_3 : F = g^x h^{r_2} \wedge E = F^x h^{r_3}]$.

We denote the above proof by

$$PK[x, r_1 : E = E(x^2, r_1)].$$

The next two proofs of knowledge assert that a committed value lies in an interval. The first one was introduced in [12], and corrected in [13]. The second one, which uses the first as building block, was introduced in [6], and is used in our scheme.

Definition 10 (Proof that a committed number lies in a larger interval) *A prover U can convince a verifier V that a number $x \in [0, b]$ which is committed in $E = E(x, r) = g^x h^r \mod n$ ($r \in [-2^\sigma n + 1, 2^\sigma n - 1]$), lies in the much larger interval $[-2^{\sigma+\tau/2}b, 2^{\sigma+\tau/2}b]$, by sending V the triple (C, D_1, D_2), where*

$$D_1 \in [cb, 2^{\delta+\tau/2}b - 1], \text{ and}$$

$$C = \mathcal{H}(g \mid\mid h \mid\mid E \mid\mid g^{D_1} h^{D_2} E^{-c}); \quad c = C \mod 2^{\tau/2}.$$

To construct the proof, U generates randoms $s \in [0, 2^{\delta+\tau/2}b - 1]$, $t \in [-2^{\delta+\tau/2+\sigma}n + 1, 2^{\delta+\tau/2+\sigma}n - 1]$; computes $W = g^s h^t \mod n$; computes $C = \mathcal{H}(g \mid\mid h \mid\mid E \mid\mid W)$, and $c = C \mod 2^{\tau/2}$; and sets $D_1 = s + cx$, $D_2 = t + cr$, repeating the procedure from the beginning if $D_1 \notin [cb, 2^{\delta+\tau/2}b - 1]$.

We denote the above by

$$PK_{CFT}[x, r : E = E(x, r) \wedge x \in [-2^{\delta+\tau/2}b, 2^{\delta+\tau/2}b]].$$

Definition 11 (Proof that a committed number lies in a slightly larger interval) *A prover U can convince a verifier V that a number $x \in [a, b]$, committed in $E = E(x, r) = g^x h^r \mod n$ ($r \in [-2^\sigma n + 1, 2^\sigma n - 1]$) lies in the slightly larger interval $[a - \alpha, b + \alpha]$, where $\alpha = 2^{\delta+\tau/2+1}\sqrt{b - a}$, by releasing $\tilde{E}_1, \bar{E}_1$, and proving:*

$$PK[x, r : E = E(x, r)],$$

$$PK[\tilde{x}_1, \tilde{r}_1 : \tilde{E}_1 = E(\tilde{x}_1^2, \tilde{r}_1)],$$

$$PK[\bar{x}_1, \bar{r}_1 : \bar{E}_1 = E(\bar{x}_1^2, \bar{r}_1)],$$

$$PK_{CFT}[\tilde{x}_2, \tilde{r}_2 : \tilde{E}_2 = E(\tilde{x}_2, \tilde{r}_2) \wedge \tilde{x}_2 \in [-\alpha, \alpha]], \text{ where } \tilde{E}_2 = \frac{E}{g^a \tilde{E}_1} \mod n,$$

$$PK_{CFT}[\bar{x}_2, \bar{r}_2 : \bar{E}_2 = E(\bar{x}_2, \bar{r}_2) \wedge \bar{x}_2 \in [-\alpha, \alpha]], \text{ where } \bar{E}_2 = \frac{g^b}{E \bar{E}_1} \mod n.$$

To construct the above proof, U computes $\tilde{E} = E/g^a \mod n$, $\bar{E} = g^b/E \mod n$; sets $\tilde{x} = x - a$ and $\bar{x} = b - x$; computes $\tilde{x}_1 = \lfloor \sqrt{x-a} \rfloor$, $\tilde{x}_2 = \tilde{x} - \tilde{x}_1^2$, $\bar{x}_1 = \lfloor \sqrt{b-x} \rfloor$, $\bar{x}_2 = \bar{x} - \bar{x}_1^2$; generates random $\tilde{r}_1$ and $\tilde{r}_2$ in $[-2^\sigma n + 1, 2^\sigma n - 1]$ s.t. $\tilde{r}_1 + \tilde{r}_2 = r$, and similarly $\bar{r}_1$, $\bar{r}_2$ s.t. $\bar{r}_1 + \bar{r}_2 = -r$; computes the commitments $\tilde{E}_1 = E(\tilde{x}_1^2, \tilde{r}_1)$, $\tilde{E}_2 = E(\tilde{x}_2, \tilde{r}_2)$, $\bar{E}_1 = E(\bar{x}_1^2, \bar{r}_1)$, and $\bar{E}_2 = E(\bar{x}_2, \bar{r}_2)$; and executes the proofs of knowledge listed in the above definition.

We denote the above proof of knowledge by

$$PK[x, r : E = E(x, r) \wedge x \in [a - \alpha, b + \alpha].$$

The last cryptographic building block we need is the verifiable ElGamal encryption of an exponent.

Definition 12 (Verifiable ElGamal encryption of an exponent) *Assume U holds a secret r, and has published the value $\omega = \chi^r$. Here χ is a generator of a group $\mathcal{F}$ of order n, where n may be prime or composite, and $0 < r < n$. We assume that the DDH assumption holds in $\mathcal{F}$. It is possible for U to prove in zero-knowledge that a pair $(A = r^{-1}y^a, B = g^a) \mod n$, is an ElGamal encryption under public key y of the exponent of ω to basis χ.*

The proof can be found in [25], and we repeat it here for convenience. For i in $\{1, \ldots, \nu\}$, U generates random t_i, and computes $g_i = g^{t_i}$, $y_i = y^{t_i}$, and $\omega_i = \chi^{y_i}$. Next, U computes

$$c = \mathcal{H}(\chi \,||\, \omega \,||\, A \,||\, B \,||\, g_1 \,||\, \omega_1 \,||\, \cdots \,||\, g_\nu \,||\, \omega_\nu). \tag{8}$$

Next, U computes $s_i = t_i - c_i a$, where c_i stand for the ith-bit of c. The proof consists of c and s_i, $i = 1, \ldots, \nu$. In order to verify, V recomputes $g_i = g^{s_i} B^{c_i}$, $y_i' = y^{s_i} A^{c_i}$, and $\omega_i = \omega^{y_i'}$, and checks that (8) holds. The rationale for the proof is that, when $c_i = 0$, the verifier checks that g_i and ω_i are correctly constructed; when $c_i = 1$, the verifier checks that (A, B) is the ElGamal Encryption of the discrete logarithm of ω to basis χ, provided that g_i and ω_i are constructed correctly. If the statement were false, U could pass only one of the verification equations, for each i. In the random oracle model, the probability of U successfully proving a false statement is $2^{-\nu}$.

$$PK[r \,:\, \omega = \chi^r \wedge A = r^{-1}y^a \wedge B = g^a]$$

5 Realizing the scheme

We now describe the scheme more concretely, starting with $\mathcal{T}$, the set of shared public parameters. $\mathcal{T}$ specifies security parameters δ, ϵ, $\sigma = \sigma_1$, σ_2, and τ, and a secure hash function $\mathcal{H}$ that maps bit-strings of arbitrary length into bit-strings of fixed length τ. A typical set of choices would be $\delta = 40$, $\sigma = 40$, $\sigma_2 = 552$, $\tau = 160$, and $\mathcal{H}(\cdot) = \text{SHA-1}(\cdot)$. The parameter ϵ should be larger than 1 by a non-negligible amount. These security parameters impact the security and efficiency of the various proofs of knowledge used in the scheme, and appear with the same names in section §4.

$\mathcal{T}$ also specifies an arithmetic group $\mathcal{G}$ and three generators g, g_1 and g_2 of $\mathcal{G}$.

In this section we assume that $\mathcal{G}$ is the quadratic residues subgroup of the multiplicative residues module p, where p is simultaneously a safe prime, i.e., and $p = 2q + 1$, with q also prime, and a Sophie Germain prime, i.e., the number $\hat{p} = 2p + 1$ is prime. Primes $\hat{p}$ such that $\hat{p} = 2p + 1$, and $p = 2q + 1$, with p and q also prime are called *strong* primes. (More generally, if $\hat{p} = mp + 1$ and $p = nq + 1$ with small m, and n, are also called strong primes, but $m = n = 2$ gives the most efficient scheme.) See [18, 21] for efficient methods to generate such primes. In order to choose g it is enough to pick a random element g' in $\mathbf{Z}_p^*$ and set $g \equiv g'^2 \mod p$, provided that $g \not\equiv 1 \mod p$. The same procedure should be used to obtain g_1 and g_2.

The scheme also requires an auxiliary group $\mathcal{F}$ of order p, which in this section will be chosen as the quadratic subgroup of the multiplicative residues modulo $\hat{p}$. Furthermore, the scheme requires a second auxiliary group $\mathcal{E}$ of unknown composite order $\hat{n}$. A trusted party generates a composite modulus n, plus a proof P that n is the product of two safe primes. The group $\mathcal{E}$ is then by definition the quadratic residue subgroup of the multiplicative residues modulo n. The order of $\mathcal{E}$ is the universally unknown number $\phi(n)/4$. (Strictly speaking, it is known by the trusted party, which may safely forget it at its discretion, as the operation of the scheme does not require *anybody* to know the factorization of n.)

Shared parameters

Security parameters δ, ϵ, σ_1, σ_2, τ (integers);
Secure hash function $\mathcal{H}(\cdot) : \{0,1\}^* \longrightarrow \{0,1\}^\tau$;
$\hat{p}$, p, q, integer primes s.t. $\hat{p} = 2p + 1$ and $p = 2q + 1$;
$\mathcal{G} = \{x \in \mathbf{Z}_p^* : \exists\, a \in \mathbf{Z}_p^*$ s.t. $x \equiv a^2 \mod p\}$;
$\mathcal{F} = \{x \in \mathbf{Z}_{\hat{p}}^* : \exists\, a \in \mathbf{Z}_{\hat{p}}^*$ s.t. $x \equiv a^2 \mod \hat{p}\}$;
$\mathcal{E} = \{x \in \mathbf{Z}_n^* : \exists\, a \in \mathbf{Z}_n^*$ s.t. $x \equiv a^2 \mod n\}$;
P, a proof that n is a product of safe primes;
g, g_1, and g_2, generators of $\mathcal{G}$.

The above public parametes can be further certified if so desired. A proof of primality can be provided for each of the primes; as for g, g_1 and g_2, anybody can verify their correct generation by testing that each is not congruent to 0 or 1 modulo p, and then verifying that each is a square, by computing the Legendre symbol and checking that:

$$\left(\frac{g}{p}\right) = \left(\frac{g_1}{p}\right) = \left(\frac{g_2}{p}\right) = 1.$$

In order to setup a group using the shared parameters above, the group magnager GM chooses x and z at random among the numbers $[1, q-1]$ and set the public keys $y = g^x, y_1 = g_1^x, y_2 = g_2^z$. The group manager should proceed to register these group-specific parameters with some certification authority. The GM would prepare a statement $\mathcal{S}$ containing (minimally) a description of the group signature algorithms, a reference to the shared parameters, GM's name, the group-specific parameters y, y_1, and y_2, and some timed information, such as start and expiration dates. The GM should also select an element $\mathcal{M}$ of $\mathcal{G}$. One possibility is to use $\mathcal{S}$ to compute the element

$$\mathcal{M} = \mathcal{H}(\mathcal{S})^2 \mod p,$$

The GM should obtain a certificate $\mathrm{CERT}_{CA}(\mathcal{S}, \mathcal{M})$ from the CA establishing the group-specific parameters.

Group-specific parameters
$\mathcal{S}$, a string describing the group, including y, y_1 and y_2;
$\mathcal{M} \in \mathcal{G}$;
CA's signature $\mathrm{CERT}_{CA}(\mathcal{S}, \mathcal{M})$.

Notice that the CA's signature scheme can be any digital signature scheme without impacting the shared-domain properties of our scheme, as the CA's public data is (implicitly) shared information.

To join the group, a prospective member U chooses a random secret u in the interval $[1, q-1]$ and computes a *pseudonym* $I = g_1^u$, as well as $J = y_1^u = I^x$. In the first message of the JOIN protocol, U issues a signed membership REQUEST message, which includes a non-interactive proof of knowledge of the logarithm to basis g_1 of U's pseudonym (see section 4):

$$\texttt{JOIN-REQUEST:}\ \ U \longrightarrow GM : S_U(\text{U's name}; I; PK[u\ :\ I = g^u]), \tag{9}$$

In the above, $S_U(\cdot)$ denotes U's digital signature algorithm. It is assumed that U has secured a certificate on a public key (any general algorithm such as DSA or RSA+SHA-1 will suffice).

Once the group manager has verified the well-formedness of the request, it replies by issuing a new membership certificate for U.

$$\texttt{JOIN-ISSUE:}\ GM \longrightarrow U : (r, s), \text{ where } r \in \mathcal{G},\ 1 < r < b,\ 0 \leq s < q, \tag{10}$$

and r, s satisfy

$$(gI)^s (yJ)^{\rho(r)} r = \mathcal{M} \mod p. \tag{11}$$

Here the reduction function $\rho(\cdot)$ is the obvious one, i.e., $\rho(r) = r \mod q$. The number b has the value

$$b = p - 2^{\sigma+\tau/2+2}\sqrt{p}.$$

To generate the above signature, the GM generates k at random with $0 \leq k < q$ and computes $r = \mathcal{M}(gg_1)^k$. If r is larger than b, the GM tries a different value for k. Otherwise the GM computes s by $s = -k - \rho(r)x \mod q = -k - rx \mod q$. [2]

At this point, U should acknowledge having received the certificate, by signing r under the pseudonym:

$$\texttt{JOIN-ACK:}\ U \longrightarrow GM : S_u(I; r). \tag{12}$$

The group manager must store both signatures issued by U during the JOIN protocol. Later, when opening group signatures, the GM recovers r and must use the transcript from protocol JOIN to prove that r belongs to U. This protects U against framing attacks by the group manager. Notice that if U refuses to provide the acknowledgment in the third step of the protocol, the group manager must engage in (a potentially expensive) remedial action, such as revoking the certificate,

[2] Since the quadratic residues are nearly uniformly distributed in the interval $[1, p-1]$, we have that $r < b$ with probability equal to $1 - \frac{2^{\sigma+\tau/2+2}}{\sqrt{p}} > 1 - 2^{-645}$ if the security parameters have the typical values $\delta = 40$, $\tau = 160$ and p has at least 768 significant bits. So nearly always the GM will generate a signature at first try. Notice that the signature scheme by itself does *not* require that $r < b$, and if any $r < p$ is used the security of the scheme is unaffected. However, for efficiency reasons having to do with characteristics of the proof of equality of logarithms in distinct groups, it is convenient to add the above condition.

or otherwise U will have a valid group membership certificate but the group manager will not be able to attribute U's signatures in an non-repudiatable manner. These circumstances are hardly unique to our group signature scheme, and there are well-known ways to mitigate this problem; for instance, a careful GM might avoid such situations altogether by using a certified delivery service to send the certificate.

JOIN protocol messages

$$\begin{array}{lcl} U \longrightarrow GM: & & S_U(\texttt{U's name}; I; PK[u \,:\, I = g^u]) \\ \longleftarrow & & (r,s), \texttt{ where } r \in \mathcal{G},\ 1 < r < b,\ 0 \le s < q, \texttt{ and} \\ & & (gI)^s (yJ)^{\rho(r)} r = \mathcal{M} \mod p \\ \longrightarrow & & S_u(I; r) \end{array}$$

We now describe the protocol `SIGN`. One goal of this protocol is that U convince a verifier V of its knowledge of a membership certificate (r, s) as above. First, U chooses at random v, with $0 \le v < q$, which will be used to blind the membership certificate. In a few words, U will convince V that it knows (r, s) that satisfy the v-th power of the membership signature equation.

U computes the vth-power of some of the terms in the signature:

$$g' = g_1^v,\ y' = y_1^v,\ W = (gI)^v,\ X = (yJ)^v,\ Z = W^s X^r. \tag{13}$$

U then sends these terms to V: g', y', W, X, and Z, and proves knowledge of u, u' and v' such that v equals the discrete logarithm $\text{Dlog}_{g_1} g'$, s.t. u equals the exponent of g' and v the exponent of g in the representation of W with respect to the basis $\{g, g'\}$. U also proves that v equals the discrete logarithm $\text{Dlog}_{y_1} y'$, and that u' equals the exponent of y' and v the exponent of y in the representation of X w.r.t. the basis $\{y, y'\}$. U also proves knowledge of a representation of Z w.r.t. the basis $\{W, X\}$.

Notice that, while $u = u'$ in a correct membership certificate, it is not necessary for U to show that to V at this point: The verification of the membership certificate equation has the side-effect of proving the equality of u and u'. Please see section §3 for the argument.

Next, U choose a fresh, random generator χ of the group $\mathcal{F}$. U computes $\omega = \chi^r$. U sends to V : χ, ω, and proceeds to prove to V that the discrete logarithm of ω in the basis χ equals the exponent of X in the representation of Z with respect to basis $\{W, X\}$. This is a proof of equality of committed values in different groups, of different orders p and q, respectively. The efficient construction for such a proof can be found in [6]. We repeat it here for reasons of convenience.

First, notice that U can view Z as a commitment to the value r in the group $\mathcal{G}$, with respect to the basis $\{X, W\}$ and with random parameter s. In other words, $Z = E_1 = E_1(r, s_1 = s) = X^r W^s \mod p$. Next, U computes a second commitment to the value r in the group $\mathcal{E}$. First, U generates two provably random elements γ, β of $\mathcal{E}$, as the squares of two consecutive values of a secure pseudo-random generator $SPRNG$. Next, U generates a random element $s_2 \in [-2^{\kappa+\tau+1}, 2^{\kappa+\tau+1}]$, where $2^{\kappa-1} \le |\mathcal{E}| < 2^\kappa$, and computes $E_2 = (r, s_2) = \gamma^r \beta^{s_2} \mod n$. Finally, $\omega = \chi^r \mod \hat{p}$ can be seen as a "degenerate" commitment to r in the group $\mathcal{F}$: $\omega = E_3 = E_3(r, 0) = \chi^r \mod \hat{p}$.

U sends E_2 to V and executes two proofs which are *formally* identical to the proof of equality of two committed values (definition 8):

First, U sends V a quadruple (c, D, D_1, D_2) satisfying:

$$c = \mathcal{H}(X \,||\, W \,||\, \gamma \,||\, \beta \,||\, E_1 = Z \,||\, E_2 \,||X^D W^{D_1} E_1^{-c} \mod p \,||\, \gamma^D \beta^{D_2} E_2^{-c} \mod n).$$

Again, refer to definition (8) for how to build this proof.

The second proof is similar: U sends V a triple (c', D', D'_1) satisfying:

$$c' = \mathcal{H}(\gamma \mid\mid \beta \mid\mid \chi \mid\mid E_2 \mid\mid E_3 \mid\mid \gamma^{D'} \beta^{D'_1} E_2^{-c'} \mod n \mid\mid \chi^{D'} E_3^{-c'} \mod \hat{p}).$$

If all of the commitments E_1, E_2, and E_3 took place within the same group the above would be a proof of equality of the committed exponent in each of the commitments. However, as the order of the groups differ, we have only proved knowledge of an integer value r which satisfies

$$r \equiv r_1 \mod q, \text{ and } r \equiv r_3 \mod p, \tag{14}$$

where r_1 and r_3 are, respectively, the exponents committed in E_1 and E_3, while r is the exponent committed in E_2; since U does not know the order of $\mathcal{E}$ it cannot set up a modular equation that the exponent of E_2 should satisfy, and must use the full integer value r. U could cheat and pass the "proof" above for any two different values r_1 and r_3, by setting r in E_2 to equal the solution, computed via Chinese Remainder Theorem, to the pair of modular equations in (14).

Assume that U tries to cheat by solving the pair of equations (14) for two values r_1 and r_3, where r_1 is not the "same" as r_3, i.e., $r_1 \not\equiv r_3 \mod q$. In that case, U must compute the commitment r in E_2 as a value necessarily larger than p. Thus, if U is required to prove that the value r_2 committed in E_2 is within an interval of width at most p, then U cannot cheat, and the commitments must all hide the same value. So to complete the "proof of equality of commitments in different groups," U executes:

$$PK[r, s_2 : E_2 = E_2(r, s_2) \wedge r \in [-\alpha, b + \alpha]].$$

Here, as in definition (11), $\alpha = 2^{\delta+\tau/2+1}\sqrt{b}$, while r is in the interval $[0, b]$. Notice that as a result of the above proof, r is restricted to an interval of width $b + 2^{\delta+\tau/2+2}\sqrt{b} < b + 2^{\delta+\tau/2+2}\sqrt{p} = p$, by choice of b.

By executing the above proof of equality of logarithms, U commits to the value r modulo p through ω. This accomplishes the comparison of the reduced value of r as a signature exponent (and thus reduced $\mod q$) with the unreduced value of r modulo p. We need the value of r modulo p before we can make comparisons with the blinded signature term r^v modulo p.

U proceeds to generate yet another random number a and computes

$$A = r^{-1} y_2^a,\ B = g_2^a, \tag{15}$$

so that the pair (A, B) is an ElGamal encryption of the value r under the group manager's public key y_2. U sends A and B to V, and proves that (A, B) encrypt the logarithm $\mathrm{Dlog}_\chi \omega$, by executing (definition 12):

$$PK[r\ :\ \omega = \chi^r \wedge A = r^{-1} y_2^a \wedge B = g_2^a].$$

Finally, $A' = A^v$ is released, together with a proof that the logarithm $\mathrm{Dlog}_A A'$ equals $\mathrm{Dlog}_{g_1} g'$, and similarly for $y'_2 = y_2^v$. Notice that we must have:

$$ZA'^{-1} = (W^s X^r) A^{-v} = ((gI)^s (yJ)^r r)^v y_2^{-av} = \mathcal{M}^v y_2^{-av}, \tag{16}$$

where the last equality comes from the certificate equation (11). Next, U proves to V that ZA'^{-1} has a representation in the basis $\{\mathcal{M}, y_2'^{-1}\}$, where the exponent of $\mathcal{M}$ equals the logarithm $\mathrm{Dlog}_A A'$ and the exponent of $y_2'^{-1}$ equals the logarithm $\mathrm{Dlog}_g B$.

The proofs of knowledge above can be partially merged so that several inputs to different hash function computations are concatenated together and a single value c is used in all the proofs. The sequence of the proofs described above together constitute a *proof of knowledge of a group membership certificate*. If moreover an arbitrary message m is added as an argument to the last proof, we get the *signature of group membership* on message m.

To summarize the protocol `SIGN` can be written as (see section §4 for the notation):

SIGN protocol

`Proof arguments:`

g', y', W, X, Z, A, B, A', y_2', χ, ω, γ, β, E_2.

`Proofs of knowledge:`

$$\begin{aligned} SPK[u,v,a: \;& g' = g_1^v \;\wedge\; y' = y_1^v \;\wedge\; A' = A^v \;\wedge\; y_2' = y_2^v \\ & \wedge W = g^v g'^u \;\wedge\; X = y^v y'^u \\ & \wedge ZA'^{-1} = \mathcal{M}^v y_2'^{-a} \;\wedge\; B = g_2^a](m) \end{aligned}$$

$$PK[r,s,s_2 : Z = E_1(r,s) = X^r W^s \wedge E_2(r,s_2) = \gamma^r \beta^{s_2}]$$

$$PK[r,s_2 : E_3(r,0) = \omega = \chi^r \wedge E_2(r,s_2) = \gamma^r \beta^{s_2}]$$

$$PK[r,s_2 : E_2 = E_2(r,s_2) \wedge r \in [-\alpha, b+\alpha]]$$

$$PK[r,a : \omega = \chi^r \wedge A = r^{-1} y_2^a \wedge B = g_2^a]$$

Algorithm `VERIFY` can be derived immediately from the above formal description of `SIGN` as a proof of knowledge. As for `OPEN`, it is enough that the group manager decrypts the pair (A,B) to obtain the value r and the corresponding group membership certificate. Notice that it is possible for GM to also construct a proof that r is the value encrypted by (A,B) *without revealing the group secret* x:

$$\texttt{OPEN}: \; PK[x: \; A/r = B^x \;\wedge\; y = g^x], \tag{17}$$

and thus `OPEN` enables GM to extract a publicly verifiable *proof of authorship* of the signature.

6 An alternative construction in the RSA ring

In this section we briefly describe another possible realization of the scheme. Much of the notation and procedures are the same as in the previous section.

The shared parameters are chosen differently. We define $\mathcal{G}$ to be the group of quadratic residues in the RSA ring generated by a composite modulus which is a product of safe primes. Namely, a trusted party generates two safe primes p, q, and publishes $n = pq$. After constructing a proof that n is formed correctly, the third party may forget its factorization, as it is not needed for the scheme.

The group $\mathcal{F}$ is chosen as a group of order n. For that, one searches for a prime $\hat{p}$ so that $\hat{p} = mn + 1$, where m is a small number. One then sets $\mathcal{F}$ to be the subgroup of m-powers in the group $\mathbf{Z}_{\hat{p}}^*$. The group-specific parameters are almost the same, except that GM proves that $\mathcal{M}$ is a quadratic residue (if desired). The `JOIN` protocol is also little changed, the difference being that we place no extra restrictions on r and that s cannot be reduced modulo the uknown order of $\mathcal{G}$. As before, let κ denote the bitlength of $|\mathcal{G}|$.

Shared parameters

Security parameters δ, ϵ, σ_1, σ_2, τ (integers);
Secure hash function $\mathcal{H}(\cdot) : \{0,1\}^* \longrightarrow \{0,1\}^\tau$;
n, a composite integer, the product of safe primes;
$\hat{p}$, a prime satisfying $\hat{p} = mn + 1$, where m is small;
$\mathcal{G} = \{x \in \mathbf{Z}_n^* : \exists\, a \in \mathbf{Z}_n^*$ s.t. $x \equiv a^2 \mod n\}$;
$\mathcal{F} = \{x \in \mathbf{Z}_{\hat{p}}^* : \exists\, a \in \mathbf{Z}_{\hat{p}}^*$ s.t. $x \equiv a^m \mod \hat{p}\}$;
P, an (optional) proof that n is a product of safe primes;
g, g_1, and g_2, generators of $\mathcal{G}$;
P', an (optional) proof that g, g_1, and g_2 are quadratic residues.

Group-specific parameters

$\mathcal{S}$, a string describing the group, including y, y_1 and y_2;
$\mathcal{M}$, and an (optional) proof that $\mathcal{M} \in \mathcal{G}$;
CA's signature $\mathrm{CERT}_{CA}(\mathcal{S}, \mathcal{M})$.

JOIN protocol messages

$$
\begin{array}{lll}
U \longrightarrow GM: & & S_U(\texttt{U's name}; I; PK[u : I = g^u]) \\
\longleftarrow & & (r,s), \text{ where } r \in \mathcal{G},\ s \in [-2^{\kappa+\tau+1}, 2^{\kappa+\tau+1}], \text{ and} \\
& & (gI)^s (yJ)^r r = \mathcal{M} \mod p \\
\longrightarrow & & S_u(I; r)
\end{array}
$$

The SIGN protocol can be considerably simplified. There is no need to an extra commitment in a group of unknown order, as $\mathcal{F}$ itself has this property. Moreover, there is no need to prove that r in the term X^r is bounded in a certain interval, as a cheating U could not find a value that reduces to different values $r_1 \bmod n$ and $r_2 \mod \phi(n)$ while satisfying the signature equation, because $\phi(n)$ is unknown to U.

SIGN protocol

Proof arguments:

g', y', W, X, Z, A, B, A', y_2', χ, ω.

Proofs of knowledge:

$$
\begin{aligned}
SPK[u,v,a: \ & g' = g_1^v \ \wedge\ y' = y_1^v \ \wedge\ A' = A^v \ \wedge\ y_2' = y_2^v \\
& \wedge\, W = g^v g'^u \ \wedge\ X = y^v y'^u \\
& \wedge\, Z A'^{-1} = \mathcal{M}^v y_2'^{-a} \ \wedge\ B = g_2^a](m)
\end{aligned}
$$

$$
PK[r,s : Z = E_1(r,s) = X^r W^s \ \wedge\ E_2(r,0) = \omega = \chi^r]
$$

$$
PK[r,a : \omega = \chi^r \wedge A = r^{-1} y_2^a \wedge B = g_2^a]
$$

Protocol OPEN is unchanged from the previous case.

7 Security analysis

In section §4 we have introduced two assumptions that underlie the security and zero-knowledge properties of various types of proofs of knowledge that we use to build our scheme. These assumptions were:

1 The DDH assumption in the quadratic residues subgroup of either a prime order field or an RSA ring where the composite modulus is the product of two safe primes.

2 The Strong RSA assumption in the quadratic residue subgroup of an RSA ring where the composite modulus is the product of two safe primes.

The proofs of security of those proofs of knowledge can be obtained from the above assumptions within the *random oracle computational model*, introduced formally in [5].

We need two more security assumptions on the Nyberg-Rueppel signature scheme in order to derive the coalition-resistance property of our scheme,

3 Non-malleability.

4 *Rigidity* (defined below).

As with any signature that allows for message-recovery, Nyberg-Rueppel signatures are vulnerable to existential forgery: One can generate signatures by generating pairs (r', s') at random and computing the corresponding message m', namely the message recovered as the right side of equation (2). Unforgeability is achieved by fixing the message m to have a particular value (as in our scheme) or a particular format (for instance, m is the cryptographic hash/digest of a formatted string).

Non-malleability (see [26]) is a property of some randomized signature schemes. A signature scheme is randomized if its computation involves at least one random parameter, so that the signature value is not determined by the message alone. All discrete-log type signature schemes are randomized. For instance in the setting of section §5, for each choice of $k \mod q$, one can obtain a signature on m by putting $r = mg^k$ and $s = -k - rx \mod q$. We say that a randomized signature scheme is *non-malleable* if given one or more signatures on a single message m, it is infeasible to forge a *new* signature on the *same* message [3].

We define *rigidity* as the following property of discrete logarithm type signatures:

Definition 13 (Rigidity) *Given several signatures (r_i, s_i) on the same message m, each under a known homomorphic transform $(g^{a_i}, g^{a_i x})$ of a single public key (g, g^x), it is infeasible to forge a new signature on m under a public key which is also a known homomorphic transform (g^a, g^{ax}) of the public key (g, g^x).*

Rigidity and non-malleability are related: If a single signature on m is given, and a forged signature on m is required, then rigidity is a weaker assumption than non-malleability, the latter implying the former. This is less clear if multiple signatures are available to assist in the forgery attempt. We postulate that both assumptions hold for the Nyberg-Rueppel signature, when the

[3] It has been observed in [26] that, in the *generic group model* [7] the non-malleability of certain signature types comes for free from the proof of unforgeability under the *adaptive chosen-message attack*. As remarked in [26], that does not imply that any given specific signature type will be invulnerable to malleability attacks, and care must be taken when specifying the protocol's parameters.

group $\mathcal{G}$ is a quadratic residue subgroup of an RSA ring generated by a safe modulus, or the quadratic residue subgroup of a finite field with cardinality equal to that of a safe prime. Under these assumptions, we show that membership certificates are unforgeable.

Assume someone has built a "membership certificate forging machine" for $\mathcal{G}$, i.e., an efficient (randomized, probabilistic) algorithm that, given as input elements $\bar{g}$, $\bar{y}$, $\bar{g}_1$, $\bar{y}_1$, $\mathcal{M}$, and u (where $\bar{y} = \bar{g}^x$ and $\bar{y}_1 = \bar{g}_1^x$, but x is not given as an input), and a sufficiently large set $S = \{(r,\ s)|(\bar{g}\bar{g}_1^u)^s(\bar{y}\ \bar{y}_1^u)^{\rho(r)}r = \mathcal{M}\}$ of membership certificates for $I = \bar{g}_1^u$, produces a new certificate (r', s') with non-negligible probability. The number $|S|$ of certificates needed by the algorithm is assumed to be bounded above by a polynomial function of a security parameter. We claim that the same algorithm can be used to show that Nyberg-Rueppel signatures given by the formula $g^s y^{\rho(r)} r = m$ in $\mathcal{G}$ are malleable, i.e., given enough signatures on the same message m, under the same public key y, it is possible to produce a new signature on the same message m.

The proof is simple: Given enough Nyberg-Rueppel signatures (r, s) on the same message m,under the public key (g, y), choose two random numbers u, v, and compute $w = 1 - uv$. Let $\bar{g} = g^w$, $\bar{g}_1 = g^v$, $\bar{y} = y^w$, $\bar{y}_1 = y^v$. It is then clear that (r, s) and u satisfy the membership certificate equation: $(\bar{g}\bar{g}_1^u)^s(\bar{y}\bar{y}_1^u)^r r = m$. Run the certificate membership forging machine to obtain a new certificate (r', s'). This new certificate satisfies $g^{s'} y^{r'} r' = m$, contradicting non-malleability.

Consider now the case of a second forging machine that takes certificates (r_i, s_i), each for a pseudonym secret u_i, and produces a new membership certificate (r', s') and pseudonym secret u'. Let $(\bar{r}_i, \bar{s}_i)$ denote Nyberg-Rueppel signatures, each under different public keys $(g_i = g^{a_i}, y_i = g^{a_i x})$, where the a_i are known, i.e., each public key is a known homomorphic transform of a single public key $(g, y = g^x)$. Generate a random w, and compute each pseudonym secret u_i to solve the equation $w = a_i - u_i$. Next, let $\bar{g} = g^w$, $\bar{g}_1 = g$, $\bar{y} = y^w$, and $\bar{y}_1 = y$. Notice that $(\bar{r}_i, \bar{s}_i)$ satisfy the certificate equation $(\bar{g}\bar{g}_1^{u_i})^{\bar{r}_i}(\bar{y}\bar{y}_1^{u_i})^{\bar{s}_i} r = m$. Let r', s' be the forged certificate, under pseudonym u'. Then (r', s') is a Nyberg-Rueppel signature on the message m under the homomorphic public key $(g^{w+u'}, y^{w+u'})$. The rigidity assumption implies that such forgeries are infeasible,and consequently, that coalition attacks do not succeed in generating new certificates.

Notice that a signature scheme lacking rigidity could be quite problematic: The new public key in a forged certificate is a known function of the original public key, and indeed the forger can prove to another party that the new pair (g_i, g_i^x) encodes the same secret x as the original pair given. If the signature is non-rigid this implies that a forger is able to give a proof of knowledge of a secret x that it does not possess.

We now claim that the our group signature scheme is non-framing and supports `exculpability`. Notice that as part of the signature protocol, the member U proves knowledge of the pseudonym secret u. No other member, even the group manager, can complete the protocol with more than a non-negligible probability of success without knowledge of the pseudonym secret. Moreover, the `OPEN` protocol is a proof of r being the secret encrypted as part of the signature transcript; therefore the group manager cannot substitute another value for r. In order to link this r with a real user identity, the group manager will have to provide the content of `JOIN-ACK`, an unforgeable signature on r with the user's pseudonym, and the content of `JOIN-REQUEST`, an unforgeable signature on the pseudonym, issued under a public key belonging to the user.

To prove the `traceability` property we must show that the user is not able to sign a message without properly encrypting the tracing value r. In order to execute all the steps of the protocol `SHOW`, the user must provide the encryption of r such that (dropping the reduction function from the notation):

$$r^{-v}\mathcal{M}^v = (gI)^{sv}(yJ)^{rv}.$$

In order to substitute another value r' for the tracing value r, assuming a forged certificate is unavailable, it must be that r' satisfies $r' \equiv r \mod |\mathcal{G}|$ and $(r/r')^v = 1$ in $\mathcal{G}$. If the group $\mathcal{G}$ is the subgroup of quadratic residues in a finite field generated by a safe prime, or the quadratic residues subgroup of an RSA ring generated by a safe modulus, then the only value v that allows a different r' to be substituted for r is $v = 2$. In the RSA case, this implies computing a different square root r' for the value r^2, which moreover satisfies $r' \equiv r \mod |\mathcal{G}|$, where the order $|\mathcal{G}|$ is unknown; since both computing the square root and solving equations $\mod |\mathcal{G}|$ are considered intractable problems, this is infeasible. In the discrete logarithm case, the only other square root for r^2 equals $-r$. The condition $r' \equiv r \mod |\mathcal{G}|$ then implies $r \equiv 0 \mod |\mathcal{G}|$, because $|\mathcal{G}|$ is prime. But certificates with $r \equiv 0 \mod |\mathcal{G}|$ are clearly not anonymous or unlinkable, and thus prohibited. We conclude that in neither case it is feasible to substitute the tracing value.

As for `anonymity` and `unlinkability`, consider the terms in the protocol that contain I, the user's pseudonym. They are $W = (gI)^v$, $X = (yJ)^v = (yI^x)^v$, and $Z = W^s X^r$. Being able to link these terms in various executions of the protocol is equivalent to solving the Diffie-Hellman Decision problem. Similarly, r appears in the terms $Z = W^s X^r$, in the term $\omega = \chi^r$ and in two related ElGamal encryptions $A = r^{-1} y_2^a$ and its power A^v. (It may also appear in an auxiliary commitment, but that term can be disregarded as it involves an extra randomizing parameter and is statistically independent of r.) Being able to link these terms violates the DDH assumption.

8 Conclusions

In this paper we introduced the first group signature scheme with constant-size parameters that does not employ any trapdoor function. This allows public parameters to be shared among distinct groups providing a big advantage over existing schemes, which require each group to maintain a separate cryptographic domain. Our scheme is not bound to a specific setting but it can work in various groups where the Decision Diffie-Hellman assumption holds.

Our scheme is as efficient as the scheme in [11] but less efficient than the state-of-the-art scheme in [4]. This is due primarily to the high cost of the current implementation of the verifiable encryption scheme. It is still an open problem to find very efficient schemes for verifiable encryption of discrete-log type signatures that, in addition, provide unlinkability. Performance improvements of this important building block could most likely be applied to our scheme.

References

[1] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. In *Advances in Cryptology – EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 591–606, Springer-Verlag, 1998.

[2] G. Ateniese. Efficient Verifiable Encryption (and Fair Exchange) of Digital Signatures. In 6th ACM CCS, pp. 138-146, 1999.

[3] G. Ateniese, R. Curtmola and B. de Medeiros. Medical Information Privacy Assurance: Cryptographic and System Aspects. In Third Conference on Security in Communication Networks 2002 (SCN02), September 12-13, 2002, Amalfi (Italy).

[4] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In M. Bellare, editor, *Advances in Cryptology — CRYPTO '2000*, Lecture Notes in Computer Science. Springer-Verlag, 2000.

[5] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the First ACM Conference on Computer and Communications Security.* 1993.

[6] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *Advances in Cryptology – EUROCRYPT'00,* vol. 1807 of Lecture Notes in Computer Science, Springer Verlag, 2000

[7] Daniel R. L. Brown. The exact security of ECDSA. In *IEEE P1363: Standard Specifications for Public Key Cryptography*, August 1998, `http://grouper.ieee.org/groups/1363`.

[8] Jan Camenisch and Ivan Damgård. Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In *Advances in Cryptology – ASIACRYPT 2000,* vol. 1976 of Lecture Notes in Computer Science, Springer Verlag.

[9] Jan Camenisch and Anna Lysyanskaya. Efficient Non-transferable Anonymous Multi-show Credential System with Optional Anonymity Revocation. In Eurocrypt'01. Springer Verlag, 2001.

[10] Jan Camenisch and Markus Michels. A group signature scheme with improved efficiency. In K. Ohta and D. Pei, editors, *Advances in Cryptology — ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 160–174. Springer-Verlag, 1998.

[11] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *Advances in Cryptology – CRYPTO'97*, vol. 1296 of LNCS, Springer-Verlag, 1997.

[12] A. Chan, Y. Frankel, and Y. Tsiounis. Easy come – easy go divisible cash. In *Advances in Cryptology – Proceedings of EUROCRYPT'98*, LNCS 1403. Springer-Verlag, 1998.

[13] A. Chan, Y. Frankel, and Y. Tsiounis. Easy come – easy go divisible cash. Updated version with corrections, GTE Technical Report. 1998. Available at `http://www.ccs.neu.edu/home/yiannis`.

[14] D. Chaum, *Security Without Identification: Transactions Systems to Make Big Brother Obsolete*, CACM Vol. 28, No. 10, October 1985.

[15] D. Chaum and J. Evertse. A secure and privacy-protecting protocol for transmitting personal information between organizations. In *Advances in Cryptology*–CRYPTO'86, pp. 118-167. Springer-Verlag, 1986.

[16] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology, – CRYPTO'91*, vol. 547 of LNCS, Springer-Verlag, 1991.

[17] L. Chen. Access with pseudonyms. In Cryptography: Policy and Algorithms, pp. 232-243. Springer-Verlag, 1995.

[18] Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. In *ACM Transactions on Information and System Security,* 2000.

[19] I. Damgård. Payment systems and credential mechanisms with provable security against abuse by individuals. In *Advances in Cryptology – CRYPTO '88*, pp. 328–335, Springer-Verlag, 1988.

[20] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology* – CRYPTO'86, Springer Verlag, 1987.

[21] Marc Joye, Pascal Paillier and Serge Vaudenay. Efficient generation of prime numbers. In *Cryptographic Hardware and Embedded Systems – CHES 2000,* vol. 1965 of Lecture Notes in Computer Science, Springer Verlag, 2000.

[22] J. Kilian and E. Petrank. Identity escrow. In CRYPTO '98, vol.1642 of LNCS, pp. 169-185, Berlin, 1998. Springer-Verlag.

[23] A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf. Pseudonym Systems. In Selected Areas in Cryptography. Springer-Verlag 1999.

[24] Kaisa Nyberg and Rainer A. Rueppel. Message recovery for signature schemes based on the discrete logarithm problem. In *Advances in Cryptology – EUROCRYPT'94,* Springer Verlag, 1994.

[25] Markus Stadler. Publicly Verifiable Secret Sharing. In *Advances in Cryptology – EUROCRYPT'96,* vol. 1070 of Lecture Notes in Computer Science, Springer Verlag, 1996.

[26] Jacques Stern, David Pointcheval, John Malone-Lee, and Nigel P. Smart. Flaws in applying proof methodologies to signature schemes. In *Advances in Cryptology – CRYPTO 2002*, vol. 2442 of LNCS, Springer Verlag, 2002.