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

February 10, 2003

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.[1] The scheme in [4] is the most efficient one and the only proven secure against an adaptive adversary. 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. This paper provides the first, affirmative answer to the question of whether it is possible to design trapdoor-free group signature schemes with public parameters that do not increase linearly in size with the number of group members.

Our schemes are useful when several distinct groups or organizations must interact and exchange information about individuals while protecting their 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

[1] 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.

a single cryptographic domain for all participating organizations may outweigh other efficiency considerations. A CTS allows users to interact anonymously with several organizations so that it is possible to prove possession of a credential from one organization to another. Different transactions cannot be linked to real identities or even pseudonyms. It is then impossible to create profiles of users even if the organizations collude and, at the same time, users cannot falsely claim to possess credentials. Optionally, a privacy officer is able to retrieve user identities in case of disputes or emergencies. Users can thus 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 since it eliminates the need for specialized techniques such as the ones employed in [9].

Organization of this paper: The next 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. We follow with some concluding observations. A security analysis is included in the appendices.

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).

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 should contain the equivalent of a digital signature by the group manager. Similarly, the requirements of traceability and exculpability imply that the group signature should hide a regular digital signature issued by the member.

3 Description

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 other words, the group membership certificate has the effect of a signature issued by the group manager. 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.

3.1 The ElGamal signature variants

Our strategy to describe the system will be to start with a regular type of ElGamal signature for the group manager, and make minor modifications to achieve a membership certificate. (ElGamal signatures are the most standard signatures based on one-way functions, as opposed to trapdoor functions as in RSA-type signatures.) Let p and q be primes, with $p = 2q + 1$ and g be an element of order q in $\mathbf{Z}_p^*$, i.e., a quadratic residue generator modulo p. Assume moreover that the group manager has published the ElGamal public key $y = g^x \mod p$, for signing messages. Let m be a message, in a so far unspecified message space $\mathcal{M}$, and $h : \mathcal{M} \to \mathbf{Z}_q^*$ a pre-image resistant, collision-resistant hash function. The following table describes some basic types of ElGamal signing equations. ElGamal signatures are probabilistic functions so they use an auxiliary random input k, which must be different for each execution of the protocol. The signature

consists of a pair (r, s), where $r = g^k \mod p$ (except in the DSA case, where $r = (g^k \mod p) \mod q$) and s is computed according to the signature generation equations below:

Variant	Signing equation	Verification equation
I	$s = k^{-1}(h(m) - xr) \mod q$	$g^{h(m)} = y^r r^s \mod p$
II	$s = x^{-1}(h(m) - kr) \mod q$	$g^{h(m)} = y^s r^r \mod p$
III	$s = xr + kh(m) \mod q$	$g^s = y^r r^{h(m)} \mod p$
IV	$s = xh(m) + kr \mod q$	$g^s = y^{h(m)} r^r \mod p$
V	$s = x^{-1}(r - kh(m)) \mod q$	$g^r = y^s r^{h(m)} \mod p$
VI	$s = k^{-1}(r - xh(m)) \mod q$	$g^r = y^{h(m)} r^s \mod p$
DSA	$s = k^{-1}(h(m) + xr) \mod q$	$r = (g^{s^{-1}h(m)} y^{s^{-1}r} \mod p) \mod q$

Table 1: ElGamal signature variants.

DSA is essentially a variant-I ElGamal signature where the verification equation has been rewritten so as to permit the signer to further reduce r modulo q. In the terminology of Generalized ElGamal signatures, DSA is a variant in *short mode*. The scheme we have developed can be applied to any of the six variants above in normal (long) mode, but not in *short* mode. However, for reasons of efficiency of the resulting scheme, we will describe the protocols for versions of the ElGamal signatures called Nyberg-Rueppel signatures [25]. Nyberg-Rueppel schemes are variants of ElGamal designed to provide message recovery. Instead of a one-way hash function, message-recovery schemes use a redundancy function. The redundancy function R is an one-to-one mapping of messages into a so-called message-signing space $\mathcal{M}_S$. The image of R, denoted $\mathcal{M}_R$, must be sparse within $\mathcal{M}_S$ i.e., given a random element of $\mathcal{M}_S$, there is a negligible probability of it being in $\mathcal{M}_R$. Otherwise, the message-recovery scheme is vulnerable to existential forgery attacks, as redundancy functions are easy to invert: For an element z in $\mathcal{M}_R$, one can efficiently compute (recover) the unique pre-image message $m = R^{-1}(z)$. The following table assumes that $\mathcal{M}_S = \mathbf{Z}_p^*$. Again, the signature calls for a random input k, and the output is a pair (r, s), where $r = R(m)g^{-k} \mod p$, and s is computed as indicated in table 2.

Variant	Signing equation	Message recovery (verification)
I	$s = k^{-1}(1 + xr) \mod q$	$R(m) = ry^{rs^{-1}} g^{s^{-1}} \mod p$
II	$s = x^{-1}(-1 + kr) \mod q$	$R(m) = ry^{sr^{-1}} g^{r^{-1}} \mod p$
III	$s = -xr + k \mod q$	$R(m) = ry^r g^s \mod p$
IV	$s = -x + kr \mod q$	$R(m) = ry^{r^{-1}} g^{sr^{-1}} \mod p$
V	$s = x^{-1}(-r + k) \mod q$	$R(m) = ry^s g^r \mod p$
VI	$s = k^{-1}(x + r) \mod q$	$R(m) = ry^{s^{-1}} g^{s^{-1}r} \mod p$

Table 2: Nyberg-Rueppel signature variants.

If the redundancy function $R(\cdot)$ if replaced by an one-way, collision-resistant hash function $h(\cdot)$ in the equations above, the message-recovery property is lost, but the signing algorithm and its security properties remain unchanged. From our perspective, this Nyberg-Rueppel "without message recovery" is interesting because the hash function is "in the base", as opposed to "in the exponent", where they appear in basic ElGamal signatures. This improves the efficiency of our schemes, which must *randomize* the hash function

values. For reasons of conciseness and clarity, we describe our methods in detail only for the Nyberg-Rueppel setting.

3.2 The setting of our scheme

We now describe the setting of our scheme. Let $\mathcal{G}$ be some arithmetic group. Not all groups $\mathcal{G}$ where Nyberg-Rueppel (or ElGamal) signatures make sense have the characteristics needed by our scheme. In section §5, we outline the specifics of the protocols 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 appendix §C. In appendix §B we describe the security assumptions attending the choice of $\mathcal{G}$.

If $\mathcal{G}$ be a suitable group. The order of $\mathcal{G}$ may be a known prime or unknown composite number. In the first case, we let $q = |\mathcal{G}|$ be that prime. If the order is known, define $\rho : \mathcal{G} \rightarrow \mathbf{Z}_q$, the *reduction function*, to be simply the reduction $\mod q$. If the order of $\mathcal{G}$ is unknown, let $\rho(\cdot)$ be an integer-valued function defined on $\mathcal{G}$. In all constructions in this paper, $\mathcal{G}$ is the set of quadratic residues modulo a prime or composite, and in that case, one can take the reduction function simply to be the identity.

Let g and g_1 be fixed, public generators for $\mathcal{G}$; it is assumed that the discrete logarithm of g with respect to g_1 (and of g_1 w.r.t. g) is unknown to group members. Let $y = g^x$ be the public key of the group manager GM, with associated secret x. Finally, this signature scheme defines the message space $\mathcal{M}$ as the set of integers modulo q in the case of known order, and the set of integers smaller than some upper bound otherwise. The signing space is $\mathcal{M}_S = \mathcal{G}$, and the secure one-way function $h(\cdot) : \mathcal{M} \rightarrow \mathcal{M}_S$ is $h(m) = g_1^m$. Clearly, $h(\cdot)$ satisfies the requirements of a secure one-way (hash) function: $h(\cdot)$ is pre-image resistant by the hardness of computing discrete logarithms in $\mathcal{G}$. In the case of known order, it is further one-to-one, hence trivially collision-resistant. In the case of unknown order, finding a collision would reveal the order of $\mathcal{G}$, i.e., it is equivalent to factorization. On the other hand, $h(\cdot)$ is clearly *not* a redundancy function, as it is not efficiently invertible and the image is not sparse. We shall demonstrate the scheme for such variant-III Nyberg-Rueppel signatures.

$$\begin{array}{lrcll} \text{Signing:} & r & = & g_1^m g^{-k} \ \ (\text{in } \mathcal{G}); & (1) \\ & s & = & -\rho(x)r + k \ \ (\text{mod } q) = -xr + k \ \ (\text{mod } q); & (2) \\ \text{Verification:} & g_1^m & = & r y^r g^s \ \ (\text{in } \mathcal{G}). & (3) \end{array}$$

We have placed "mod q" within parenthesis as that reduction is only computed when the order of $\mathcal{G}$ is a known prime. These certificates are issuable only by the group manager GM, who is privy to the secret key x associated to y. So a membership certificate is simply a signature as such, where the message m identifies the group member.

3.3 The scheme

A prospective new member U who wishes to join the group must have first secured a digital signature certificate with some certification authority. U starts the join protocol by choosing a random, secret secret value u and computing $I_U = g_1^u$. More precisely, U and GM interact so that both contribute to the randomization of u, while its value remains secret from the GM. Then U constructs a zero-knowledge proof (of knowledge) of the discrete logarithm of the pseudonym I_U with respect to g_1. U signs the pseudonym and the proof of knowledge of the pseudonym secret, and sends it to the GM to request a group membership certificate.

GM verifies the signature against U's public certificate and the correctness of the zero-knowledge proof. If both are well-formed, GM responds with the signature pair (r, s) on I_U, which is technically GM' signature on an message u known only to U. This is safe from the GM's viewpoint because both GM and U contribute to the choice of the value u. It is imperative, however, that only U knows the value u, as it is in effect the secret key allowing U to use the membership certificate to issue signatures. The equations used by GM to generate (r, s) are:

$$r = I_U g^{-k} \pmod{\mathcal{G}};\ s = -\rho(x)r + k \pmod{q} = -xr + k \pmod{q}, \tag{4}$$

where k is a random parameter of GM's choice, and the reduction modulo q is applied only in the case of known order. U verifies the signature, checking that:

$$I_U = r y^r g^s \pmod{\mathcal{G}}. \tag{5}$$

In the equation above, and throughout the rest of this paper, we shall omit the reduction function $\rho(\cdot)$ from the notation.

The scheme must permit U to prove knowledge of this certificate pair (r, s) without revealing any linkable function of r, s, or u. It must also 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. However, unlinkability between different protocol executions is not a requirement of verifiable encryption schemes, and indeed 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, the `SIGN` protocol employs a secret exponent to protect in zero-knowledge the signature pair (r, s), along with other parameters required by our scheme. More concretely, every time the group member must use the certificate, she raises all terms of the congruence (5) to a freshly randomly generated secret value v, thus obtaining: $I_U^v = r^v y^{vr} g^{vs} \pmod{\mathcal{G}}$. This can be rewritten as:

$$R := I_U^v g^{-vs} = r^v y^{vr}. \tag{6}$$

For simplicity we denote R to be the value represented by either side of the equation above. In order to prove knowledge of a membership certificate, the member U will release the randomized value R and prove that it can be written in two different ways, so that the equation (6) holds for some value v which is not disclosed to the prover.

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 ElGamal encryption of r, and comparing that with an encryption of r^v. The encrypted value of r can then be re-used for comparison with the exponent of y^v appearing in R. However, this presents a new difficulty: The value in the exponent – actually $\rho(r)$, which we shorten to r – is an integer reduced modulo the order of the group $\mathcal{G}$, while the encrypted value r is an element of $\mathcal{G}$ itself. The reduction function does not preserve group operations, it is not multiplicative; and the method for proving equality between an ElGamal-encrypted value and a logarithm, due to Stadler [26], cannot be directly applied. The solution is to employ a technique due to Boudot [6] that permits efficient comparison between logarithms in different groups. So we use an auxiliary group $\mathcal{F}$ of order compatible with the operations in $\mathcal{G}$. We release a commitment to the value r as an

exponent of an element of $\mathcal{F}$, and we show that it equals (up to modular reduction), the exponent of y^v in R. Next, we use Stadler's technique to prove the equality of the encrypted value r (in $\mathcal{G}$), with the value committed as an exponent in $\mathcal{F}$.

To complete the sign protocol, we also compute a verifiable encryption of the value I_U associated to the certificate. This permits GM to open the signature with just an ElGamal decryption operation.

4 Proofs of knowledge

In this paper we make use of several types of proofs of knowledge about various relations between secrets. All these proofs of knowledge have been presented elsewhere; we make no claims of originality in this section, which we have added to make the paper self-contained and to harmonize the notation. More details about these proofs of knowledge have also been included in appendix §A, where we also discuss the underlying security assumptions.

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. Let ϵ denote a second security parameter

Definition 1 (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 $||$ 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 (1), 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:

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

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

Definition 2 (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 $h_i = g_i^x, i = 1 \ldots \ell$.*

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

Definition 3 (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 $A = g_1^{x_1} \cdots g_\ell^{x_\ell}$.*

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 4 (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. 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 5 (Proof of knowledge of a committed value) *With notation as in the paragraph above, U can prove to a verifier V knowledge of a number x committed in the value $E = E(x, r) = g^x h^r$.*

We denote it by: $PK[x, r \,:\, E = E(x, r)]$

Definition 6 (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.*

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 7 (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 $\alpha = 2^{\delta+\tau/2+1}\sqrt{b-a}$.*

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 8 (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 $0 < r < n$. 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.*

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

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 appendix §A. $\mathcal{T}$ 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.)

The above public parameters 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 manager GM chooses x and z at random among the numbers $[1, q - 1]$ and set the public keys $y = g^x$, and $y_2 = g_2^z$. The group

Shared parameters

Security parameters: δ, ϵ, σ_1, σ_2, τ;
Secure hash function: $\mathcal{H}(\cdot) : \{0,1\}^* \longrightarrow \{0,1\}^\tau$;
$\hat{p}$, p, q, 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. $x \equiv a^2 \mod n\}$;
g, g_1, and g_2, generators of $\mathcal{G}$.

Group-specific parameters

$\mathcal{S}$, a string including
y and y_2;
CA's signature: $\text{CERT}_{CA}(\mathcal{S})$.

Join protocol

$U \longrightarrow GM$:	$J_U = I^m \mod p$
$GM \longrightarrow U$:	$a,\ b \mod q$
$U \longrightarrow GM$:	$Sig_U(I_U = J_U^a g_1^b,\ PK[u : I_U = g_1^u])$
$GM \longrightarrow U$:	$r = I_U g^{-k} \mod p,\ s = -xr + k \mod q$

Table 3: Shared parameters, group specific parameters, and the JOIN protocol

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 obtain a certificate $\text{CERT}_{CA}(\mathcal{S})$ from the CA establishing the group-specific parameters.

Let $Sig_U(\cdot)$ denote U's signature algorithm. To join the group, a prospective member U chooses a random secret m in the interval $[1, q-1]$, computes $J_U = g_1^m$, and sends this value to GM, who responds with two values a, and b in $[1, q-1]$. U computes his pseudonym as $I_U = J_U^a g_1^b$, and its associated secret $u = am + b \mod q$. Next, U constructs a non-interactive proof of knowledge of the logarithm to basis g_1 of this pseudonym (see section 4), and also his signature $S = Sig_U(I_U, PK)$ on both the pseudonym and the proof-of-knowledge just constructed. U forwards to the GM this signature S.

The GM now verifies that the pseudonym incorporated his contribution, i.e., $I_U = J_U^a g_1^b$. This step is important because u is unknown to GM, who must sign it. Since the GM contributed to u's randomness, that does not constitute a threat to the GM's signature algorithm. The GM also verifies the correctness of the proof-of-knowledge and U's signature. If satisfied, the GM generates a random $k \mod q$, and computes $r = I_U g^{-k} \mod p$, checking that $r < c$, where c equals:

$$c = p - 2^{\sigma+\tau/2+2}\sqrt{p}, \tag{7}$$

and repeating the process of computing other random k and r until such an r is found. We remark that $r < c$ with overwhelming probability in a single attempt.[2] This very minor restriction on the possible values of r reflects requirements of the proof of equality of discrete logarithms in distinct groups, as we shall see later. After a suitable r is found, U computes $s = k - xr \mod q$, and sends the certificate (r, s) to U. The GM also records the signature S, which ties U's identity to the certificate's pseudonym. U verifies that the certificate (r, s) satisfies the verification equation, and if so, accepts it as valid.

[2]Since the quadratic residues are nearly uniformly distributed in the interval $[1, p-1]$, we have that $r < c$ 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.

SIGN protocol

Proof arguments:

Y, R, X_1, X_2, Y_1, Y_2, W_1, W_2, Z_1, Z_2, χ, γ, β, E_1, and E_2.

Signature of knowledge:

$$SPK[v, u', \ell', s', r, s_2, t, t_2 : Y = y^v \ \wedge \ Y_1 = X_1^v \ \wedge \ Y_2 = X_2^v \ \wedge \ Z_1 = W_1^v \ \wedge \ Z_2 = W_2^v$$
$$\wedge \ R = g_1^{u'} g^{s'} \ \wedge \ Y_1 = g_1^{u'} y_2^{\ell'} \ \wedge \ Y_2 = g_2^{\ell'}$$
$$\wedge \ E_1 = E_1(r, 0) = \chi^r \ \wedge \ W_1 = r^{-1} y_2^t \wedge W_2 = g_2^t$$
$$\wedge \ E_2 = E_2(r, s_2) = \gamma^r \beta^{s_2} \ \wedge \ r \in [-2^{\delta+\tau/2+1}\sqrt{c}, c + 2^{\delta+\tau/2+1}\sqrt{c}]$$
$$\wedge \ E_3 = E_3(r, t_2) = RZ_1 = Y^r y_2^{t_2}](\mathcal{M})$$

Table 4: The SIGN protocol

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 $0 \leq v < q$, 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. As in section §3, let R denote $I_U^v g^{-vs} = r^v y^{vr} \mod p$. U computes a computationally zero-knowledge commitment to the value v as $Y = y^v \mod p$. U then releases R, Y, and the following ElGamal encryption pairs:

$$(X_1, X_2) = (I_U y_2^{\ell}, g_2^{\ell}); \quad (Y_1, Y_2) = (I_U^v y_2^{v\ell}, g_2^{v\ell}) = (X_1^v, X_2^v),$$
$$(W_1, W_2) = (r^{-1} y_2^t, g_2^t); \quad (Z_1, Z_2) = (r^{-v} y_2^{vt}, g_2^{vt}) = (W_1^v, W_2^v).$$

At this point, U shows that the pairs (Y_1, Y_2) and (Z_1, Z_2) encrypt the v-power of the pairs (X_1, X_2) and (W_1, W_2), respectively, by executing $PK[v : Y = y^v \wedge Y_1 = X_1^v \wedge Y_2 = X_2^v \wedge Z_1 = W_1^v \wedge Z_2 = W_2^v]$.

Next, U demonstrates that the pseudonym I_U is encrypted by the pair (X_1, X_2), and proves knowledge of the pseudonym secret u, by executing $PK[u', \ell', s' : R = g_1^{u'} g^{s'} \ \wedge \ Y_1 = g_1^{u'} y_2^{\ell'} \ \wedge \ Y_2 = g_2^{\ell'}]$. This step is crucial to prevent framing attacks against U, as not even the group manager can execute it without knowledge of u.

Continuing with the SIGN protocol, U generates a fresh, random generator χ of the group $\mathcal{F}$, and computes a (computationally zero-knowledge) commitment to the value r as $E_1 = E_1(r, 0) = \chi^r)$. In the language of section §4, this is a (degenerate) commitment to the value r in the group $\mathcal{F}$, with respect to the generator χ.

U also generates a commitment to r in the auxiliary group $\mathcal{E}$ of unknown order. For that, U uses two generators β and γ of $\mathcal{E}$, where β and γ are provably randomly generated, so that U cannot know their relative discrete logarithm. For instance, γ and β can be generated as the squares of two consecutive values of a secure pseudo-random number generator *SPRNG*. The commitment is computed as $E_2 = E_2(r, s_2) = \gamma^r \beta^{s_2}$, where s_2 is a random parameter of U's choice: $s_2 \in [-2^{\kappa+\tau+1}, 2^{\kappa+\tau+1}]$, where $2^{\kappa-1} \leq |\mathcal{E}| < 2^{\kappa}$. Notice that the value $RZ_1 = y^{vr} y_2^{vt} = Y^r y_2^{vt}$ is also a commitment to the value r in the group $\mathcal{G}$, with generators Y, and y_2. We denote it by $E_3 = RZ_1$.

In the next step, U reveals the commitments E_1, E_2, and the respective generators γ, β, and χ. (In the case of γ and β, U must also reveal the seed of the *SPRNG* that leads to the computation of γ and β.) U then shows that E_1, E_2 and E_3 all are commitments to the same value r. Notice that we are following the efficient construction found in [6], repeated in detail here for reasons of convenience.

U executes two proofs of equality of two committed values (def. 6). In the first proof U sends V a triple (c', D', D_1') satisfying: $c' = \mathcal{H}(\chi \,||\, \gamma \,||\, \beta \,||\, E_1 \,||\, E_2 \,||\, \chi^{D'} E_1^{-c'} \mod \hat{p} \,||\, \gamma^{D'} \beta^{D_1'} E_2^{-c'} \mod n)$. Again,

refer to def. (6) for how to build these proofs. In agreement with the notation in section §4,we denote the above by $PK[r, s_2 : E_1 = E_1(r, 0) \ \wedge \ E_2 = E_2(r, s_2)]$. Then U sends V a quadruple (c, D, D_1, D_2) satisfying: $c = \mathcal{H}(\gamma \mid\mid \beta \mid\mid Y \mid\mid y_2 \mid\mid E_2 \mid\mid E_3 \mid\mid \gamma^D \beta^{D_1} E_2^{-c} \mod n \mid\mid Y^D y_2^{D_2} E_3^{-c} \mod p)$. We denote that by $PK[r, s_2, t_2 : E_2 = E_2(r, s_2) \ \wedge \ E_3 = E_3(r, t_2)]$, where $t_2 = vt$.

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

$$r \equiv r_1 \mod p, \text{ and } r \equiv r_3 \mod q, \tag{8}$$

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. [3] 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 the Chinese Remainder Theorem, to the pair of modular equations in (8). Thus, a non-member U' would be able to forge the proof of knowledge of a certificate, by choosing r_3 and s arbitrarily, computing the value r_1 that would make the certificate equation work, and then solving the pair of equations (8) for an r that reduces to $r_1 \mod p$ and $r_3 \mod q$, respectively. In the cheating case, however, because $r_1 \not\equiv r_3 \mod q$, U' computes a value $r > p$ as the solution of 8. 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, this forgery attack is prevented; and the commitments must all hide the same value. So to complete the "proof of equality of commitments in different groups," U must construct a proof that the value r is restricted to an interval of width at most p. For that, U uses the fact that $r < c$, and constructs the proof of knowledge that a committed value lies in a slightly larger interval, def. (7): $PK[r, s_2 : E_2 = E_2(r, s_2) \ \wedge \ r \in [-2^{\delta+\tau/2+1}\sqrt{c}, c + 2^{\delta+\tau/2+1}\sqrt{c}]$. To observe that the interval in question has width smaller than p, notice that its width equals $c + 2^{\delta+\tau/2+2}\sqrt{c} < c + 2^{\delta+\tau/2+2}\sqrt{p} = p$, by choice of c (see 7).

Finally, U must show that the exponent committed in E_1 equals the value encrypted in the pair (W_1, W_2), by executing (definition 8): $PK[r \ : \ E_1 = \chi^r \ \wedge \ W_1 = r^{-1} y_2^t \wedge W_2 = g_2^t]$. The actual protocol `SIGN` combines all the proofs of knowledge into a single signature of knowledge. This is done by simultaneously committing to all the inputs of the proofs and using the resulting challenge in all the verification equations (à la Fiat-Shamir). In addition, the message $\mathcal{M}$, that has to be signed, is used as an extra input of the hash function.

The protocol is summarized in table 5. Moreover, algorithm `VERIFY` can be derived immediately from the above formal description of `SIGN` as a proof of knowledge of a group certificate.

As for `OPEN`, it is enough that the group manager decrypts the pair (X_1, X_2) to obtain the value I_U and the corresponding group membership certificate. GM constructs a proof that I_U is indeed the value encrypted in (X_1, X_2) *without revealing the group secret* x: $PK[x : \ X_1 I_U^{-1} = X_2^x \wedge \ y = g^x]$, a publicly verifiable *proof of authorship* of the signature.

6 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.

[3] As 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.

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] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* Chapter §11, note 11.83, pp. 461–462. CRC Press, 1996.

[25] 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.

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

[27] 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.

A More proofs of knowledge

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. The notation is as in section §4.

Definition 9 (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 $\mathcal{G}$:*

$$\textit{Dist}_1 = \{(h, i, j), \textit{ where } h, i, j \textit{ are independently randomly distributed (i.r.d.) in } \mathcal{G}\}$$

and

$$\textit{Dist}_2 = \{(h', i', j'), \textit{ where } h' = g^x, i' = g^y, j' = g^{xy} \textit{ for i.r.d. } x, y \textit{ with } 0 \leq x, y < q\}$$

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 10 (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 (6).

Definition 11 (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 $s = k - cx$.

Definition 12 (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 $s_i = k_i - cx_i$.

Definition 13 (Proof of knowledge of a committed value) *With notation as in §4, 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 14 (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 $\mathbf{Z}$).

Definition 15 (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 16 (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 17 (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 18 (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 [26], 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

$$c = \mathcal{H}(\chi \,||\, \omega \,||\, A \,||\, B \,||\, g_1 \,||\, \omega_1 \,||\, \cdots \,||\, g_\nu \,||\, \omega_\nu). \tag{9}$$

Next, U computes $s_i = t_i - c_i a$, where c_i stand for the i^{th}-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 (9) 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 $2^{-\nu}$.

B 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].

The unforgeability of membership certificates is equivalent to the unforgeability of Nyberg-Rueppel signatures, where the redundancy function has been substituted by a one-way (hash) function. Such modification does not affect either the signing algorithm or its security properties. This is because the security of Nyberg-Rueppel signatures depend on the redundancy function having the following two properties [24]:

- Given a randomly chosen value in the set of quadratic residues, there is only a negligible probability that it belongs to its image of the redundancy function.
- If $R = R(m)$ is the redundancy function of a message, then by choosing random values z_1, and z_2 and computing $R' = Rg^{z_1}y^{z_2}$, there is only a negligible probability that the resulting value R' belongs to the image of the redundancy function.

When we substitute the redundancy function by the one-way function those properties do not hold. (As the one-way function has full image in the quadratic residues.) Yet, computing the pre-image is infeasible in the first case because of the discrete logarithm problem, and in the second case because the representation of g_1 with respect to g, and y, is unknown. Therefore, the substitution of the redundancy function by this hash function is legitimate: The correct computation of the hash function is verified when the signer proves knowledge of the pseudonym secret.

Finally, we point out that the scheme can be applied to the basic ElGamal signatures, though not without sacrifice in performance. This is because the hash function would be in the exponent, and we would have to prove knowledge of a double discrete logarithm to verify its correctness.

We now claim that the our group signature scheme is non-framing and supports `exculpability`. 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 I_U being the secret encrypted as part of the signature transcript; therefore the group manager cannot substitute another value for I_U. In order to link this I_U with a real user identity, the group manager will have to provide the transcript of the third message of the join protocol, U's public and unforgeable signature on the pseudonym.

To prove the `traceability` property we must show that the user is not able to sign a message without properly encrypting the tracing value I_U. During execution of the protocol `SIGN`, the user provides the encryption of I_U such that I_U^v satisfies the v^{th}-power of the verification equation (dropping the reduction function from the notation):

$$I_U{}^v = r^v y^{rv} g^*,$$

where $*$ is a don't-care value. Since the signature scheme itself is resistant to forgery, the only values $I_{U'}$ that can be substituted for I_U differ from I_U by a v^{th}-root of unity, i.e., $(I_U/I_{U'})^v = 1$ in $\mathcal{G}$. In the case where $\mathcal{G}$ is the quadratic residue subgroup in $\mathbf{Z}_p^*$, where p is a safe prime, there are no roots of unity apart from 1 itself. In the case that $\mathcal{G}$ is the quadratic residue subgroup in $\mathbf{Z}_n^*$, where n is a product of safe primes, the roots of unity (different from 1) are powers of $g^{(p-1)/2}$ and $g^{(q-1)/2}$, thus hard to compute without knowledge of the factorization of n. We conclude that substituting the tracing value is infeasible.

As for `anonymity` and `unlinkability`, they can be recovered from the fact that the signing protocol only reveals values which have been randomized with an exponent, or reveals semantically-secure encryptions of those values. Being able to link the same term in two different protocol executions is thus equivalent to being able to solve the Diffie-Hellman Decision problem.

C 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 section 5. 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 the same.

The `JOIN` protocol is little changed. There are no restrictions on the value of $r = I_U g^{-k} \mod n$, where k is chosen in the interval $[-2^{\tau+2\kappa}, 2^{\tau+2\kappa} - 1]$; as before, κ stands for the bitlength of $|\mathcal{G}|$. The terms a, b, and s cannot be reduced modulo the unknown order of $\mathcal{G}$, which is unknown.

The `SIGN` protocol can be considerably simplified. There is no need for an extra commitment in a group of unknown order, as the order of the group $\mathcal{G}$ is itself unknown. Moreover, there is no need to prove that the r in the commitment E_1 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.

Protocol `OPEN` is unchanged from the previous case.

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;

$\mathcal{G} = \{x \in \mathbf{Z}_n^* : \exists\, a \in \mathbf{Z}_n^* \text{ s.t. } x \equiv a^2 \mod n\}$;

$\mathcal{F} = \{x \in \mathbf{Z}_{\hat{p}}^* : \exists\, a \in \mathbf{Z}_{\hat{p}}^* \text{ s.t. } x \equiv a^m \mod \hat{p}\}$;

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 including y and y_2;

CA's signature $\text{CERT}_{CA}(\mathcal{S})$.

Join protocol

$U \longrightarrow GM$:	$J_U = I^m \mod n$
$GM \longrightarrow U$:	$a,\ b \in [-2^{\tau/2+\kappa}, 2^{\tau/2+\kappa} - 1]$
$U \longrightarrow GM$:	$Sig_U(I_U = J_U^a g_1^b \mod n,\ PK[u : I_U = g_1^u])$
$GM \longrightarrow U$:	$r = I_U g^{-k} \mod n,$
	$s = -xr + k \in [-2^{2\kappa+\tau+1}, 2^{2\kappa+\tau+1} - 1]$

SIGN protocol

Proof arguments:

Y, R, X_1, X_2, Y_1, Y_2, W_1, W_2, Z_1, Z_2, χ, E_1.

Signature of knowledge:

$$SPK[v, u', \ell', s', r, t, t_2 : Y = y^v \ \wedge\ Y_1 = X_1^v \ \wedge\ Y_2 = X_2^v \ \wedge\ Z_1 = W_1^v \ \wedge\ Z_2 = W_2^v$$
$$\wedge\ R = g_1^{u'} g^{s'} \ \wedge\ Y_1 = g_1^{u'} y_2^{\ell'} \ \wedge\ Y_2 = g_2^{\ell'}$$
$$\wedge\ E_1 = E_1(r, 0) = \chi^r \ \wedge\ W_1 = r^{-1} y_2^t \ \wedge\ W_2 = g_2^t$$
$$\wedge\ E_2 = RZ_1 = E_2(r, t_2) = Y^r y_2^{t_2}](\mathcal{M})$$