Efficient Group Signatures without Trapdoors [*]

Giuseppe Ateniese Breno de Medeiros
ateniese@cs.jhu.edu breno.demedeiros@acm.org

Group signature schemes enable unlinkably anonymous authentication, in the same fashion that digital signatures provide the basis for strong authentication protocols. This paper introduces the first group signature scheme with constant-size parameters that does not require any group member, including group managers, to know trapdoor secrets. This novel type of group signature scheme allows public parameters to be shared among organizations, and are useful when several distinct groups must interact and exchange information about individuals while protecting their privacy.

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 [CvH91] in 1991. That was followed by several other works, but only relatively recent ones [ACJT00, CS98, CS97] have group public keys and group signatures with sizes that do not depend on the number of group members. (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.) The scheme in [ACJT00] is the most efficient and is the only one proven secure against an adaptive adversary. However, all the existing group signature schemes providing constant-size parameters require the group manager to know the factors of an RSA modulus. Sharing these factors among group managers of different organizations would compromise the security and/or the trust assumptions of the entire scheme. This paper describes 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. An informal proof of security for the scheme is provided (along the lines of the proof in [ACJT00]), in section §6.

[*]This is a revised version of the paper that appeared in ASIACRYPT 2003. In particular, it corrects the proof of security of the modified Nyberg-Rueppel signature and contains some changes in the writing and presentation of the results.

1.1 Motivation

Credential transfer systems (CTS) [Cha85, CE87, Dam88, Che95, LRSW99, CL01] are examples of environments where distinct groups or organizations are able to exchange information about individuals without compromising their privacy. CTS can be built via group signature schemes [CL01]. 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. 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 scheme with shareable parameters has a definite advantage since it eliminates the need for specialized techniques such as the ones employed in [CL01].

Recently in [BMW03], it has been shown that group signatures can be built based on the assumption that trapdoor functions exist. In this paper the same constructions are provided, based on the existence of one-way functions. The scheme is the first to be functionally trapdoor-free as no group member, nor even the group manager, needs to know the trapdoor information. Although a construction in the RSA ring is used that relies on the strong RSA assumption for security, the operation of the scheme exploits only the one-wayness of the RSA function, not its trapdoor properties.

2 Preliminaries

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 [FS87a], 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.

2.1 The Elgamal signature variants

In order to design the group certificate, it is helpful to start with a regular type of Elgamal signature for the group manager, and consider the necessary modifications to obtain a certificate form that lends itself to efficient verifiable encryption. (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 $h : \mathcal{M} \rightarrow \mathbf{Z}_q^*$ 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:

Table 1: Elgamal signature variants.

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	$s = k^{-1}(h(m) + xr) \mod q$	$r = (g^{s^{-1}h(m)} y^{s^{-1}r} \mod p) \mod q$

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* – notice that the value r is further reduced modulo q, resulting in a signature of smaller length. The group signature scheme described in this paper 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, only the application of the scheme to variants of the Elgamal signatures called Nyberg-Rueppel signatures is described. More concretely, the fact that Elgamal signatures involve hash function values as exponents makes it more difficult to prove (in zero-knowledge) the knowledge of a triple (m, r, s) satisfying any of the equations in table 2.1. The Nyberg-Rueppel presented in the next section have different verification equations where the redundancy function (the analogue of the hash function in the Elgamal case) appears as a base, not an exponent. After we replace the redundancy function with a number-theoretic function (exponentiation), the modified verification equations lead to more efficient knowledge proof constructions.

2.2 Modified Nyberg-Rueppel signatures

Nyberg-Rueppel signatures [NR94] are Elgamal-type signature variants originally 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, by definition, efficiently invertible. The NR signature, being probabilistic, 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. (Notation in the table implies the assumption that $\mathcal{M}_S = \mathbf{Z}_p^*$.)

Table 2: Nyberg-Rueppel signature variants.

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	$s = k^{-1}(x + r) \mod q$	$R(m) = ry^{s^{-1}}g^{s^{-1}r} \mod p$

If in the equations above, the redundancy function $R(\cdot)$ is replaced by an one-way function then the message-recovery property is lost. On the other hand, the requirement that the image of the function be sparse in the signing space may also be dropped. Moreover, the form of the modified verification equation – if the one-way function is suitably chosen – lends itself to the construction of proofs of knowledge of signatures that are more efficient. (When compared to similar proofs for unmodified Elgamal-type signature variants.)

Let $\mathcal{G}$ be a suitable group. (Not all groups $\mathcal{G}$ where Nyberg-Rueppel – or Elgamal – signatures make sense have the characteristics needed by our scheme. See section §4 for examples of appropriate groups.) The order of $\mathcal{G}$ may be a known prime or unknown composite number. 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 signer GM, with associated secret x. (In the group signature scheme, y corresponds to the certificate issuing key.) 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 let the one-way function $h(\cdot) : \mathcal{M} \to \mathcal{M}_S$ be defined by $h(m) = g_1^m$. Clearly, $h(\cdot)$ satisfies the requirements of a secure one-way 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.

The signing and verification algorithms of the modified Nyberg-Rueppel are as follows:

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

The term "mod q" is placed within parenthesis as that reduction is only computed when the order of $\mathcal{G}$ is a known prime. These signatures are issuable only by the signer GM, who is privy to the secret key x associated to y.

2.3 High level description of the scheme

In this section a general overview of the construction is provided, to serve as an intuitive introduction to the more technical details that follow. 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 value u and computing the pseudonym $I_U = g_1^u$. 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 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's 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. GM generates (r, s) as follows:

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

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:

$$I_U = r y^r g^s \quad (\text{in } \mathcal{G}). \tag{5}$$

The scheme must permit U to prove knowledge of the pseudonym I_U and certificate pair (r, s) without revealing any linkable function of r, s, or I_U. It must also allow GM to *open* the proof and show the identity of the group member. It would seem that both problems can be solved by employing a *verifiable encryption* of digital signature schemes. However, existing schemes [CD00, ASW98, KP98, Ate99] take the approach of revealing the value r of the signature pair (r, s) and applying the actual protocol only to the second value s, thus reducing the problem of verifiable encryption of a digital signature to the simpler problem of verifiably encrypting a discrete logarithm. In the current context this approach is insufficient, as revealing r has the result of linking different protocol executions, in contradiction with the unlinkability requirement.

To solve this issue, it is necessary to Elgamal encrypt the value r as well, and prove in zero-knowledge that a Nyberg-Rueppel signature is known on a secret value u. More concretely, every time the group member must use the certificate, she encrypts the inverse of the value r, to get the Elgamal pair $(R_1, R_2) = (r^{-1} y_2^\ell, g_2^\ell)$. This encryption is under the second public key $y_2 = g_2^z$ of the group manager, used for opening group member signatures, with associated secret z.

The group member also encrypts his pseudonym: $(Y_1, Y_2) = (I_U y_2^{\ell'}, g_2^{\ell'})$. Notice that the product cipher is:

$$(R_1 Y_1, R_2 Y_2) = (I_U r^{-1} y_2^{\ell+\ell'}, g_2^{\ell+\ell'}) = (y^r g^s y_2^{\ell+\ell'}, g_2^{\ell+\ell'}) \tag{6}$$

In order to prove knowledge of a membership certificate, the member U releases the above Elgamal encrypted pairs (R_1, R_2) and (Y_1, Y_2) and proves that the product cipher encrypts some information which the signer can write in two ways, i.e., as the product $I_U r^{-1}$ for pseudonym I_U (for which the signer knows the corresponding pseudonym secret) and value r, and also as $y^r g^s$, for the same value r and some s known to the signer. In other words, the signer shows that an equation like (6) holds for the product cipher.

To proceed, one must overcome a difficulty with equation (6): The value in the exponent is reduced modulo the order of the group $\mathcal{G}$, while the encrypted value r is an element of $\mathcal{G}$ itself. Note that the reduction function does not preserve group operations, and it is not multiplicative. Therefore, the method for proving equality between an Elgamal-encrypted value and a logarithm, due to Stadler [Sta96], cannot be directly applied. The solution is to employ a technique due to Boudot [Bou00] that permits efficient comparison between logarithms in different groups. For that, it is necessary to use an auxiliary group $\mathcal{F}$ of order compatible with the operations in $\mathcal{G}$. First, a commitment to the value r as an exponent of an element of $\mathcal{F}$ is computed. Next, it is shown that this equals the exponent of y in the product cipher (R_1Y_1, R_2Y_2). (That is, the exponent in the representation, with respect to the basis $\{y, g\}$, of the value encrypted in the product cipher.) Next, Stadler's technique is used to prove the equality of the encrypted value r (in the pair R_1, R_2 of $\mathcal{G}$), with the value committed as an exponent in $\mathcal{F}$.

To complete the sign protocol, the signer proves knowledge of the discrete logarithm to basis g of the value I_U, which is Elgamal encrypted in the pair (Y_1, Y_2). This shows that the group manager is able to open the signature with just an Elgamal decryption operation.

The construction of the group signature scheme requires several types of proofs of knowledge about various relations between secrets. All these proofs of knowledge have been presented elsewhere. In order to harmonize the notation, the definitions are presented in the following section.

3 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.

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. 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.

Assumption 3.1 (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 $\mathcal{G}$:*

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

and

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

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.

Assumption 3.2 (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 (3.6).

Defn 3.1 (Proof of knowledge of a discrete logarithm) *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 $\{-2^{\epsilon(\tau+\kappa)+1}$, …, $2^{\epsilon(\tau+\kappa)+1} - 1\}$ and c in $\{0, \ldots, 2^\tau - 1\}$, s.t. $c = \mathcal{H}(g||h||g^s h^c)$, where the symbol $||$ denotes string concatenation.*

To compute the pair (s, c), 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). Denote it by (notation introduced in [CS97]): $PK[x : h = g^x]$.

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, instead compute c as: $c = \mathcal{H}(g||h||g^s h^c||m)$. Denote this *signature of knowledge* ([CS97]) by: $SPK[x : h = g^x](m)$.

Returning to the notation in definition (3.1), 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 one 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 following definitions assume the group order q is unknown; as above, it is straightforward to adapt them to the case of known order.

Defn 3.2 (Proof of knowledge of a common discrete logarithm) *U can prove to a verifier V his knowledge of an x (with $0 \le x < 2^\kappa$) s.t. 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 ($-2^{\epsilon(\tau+\kappa)+1} \le s < 2^{\epsilon(\tau+\kappa)+1}$ and $0 \le c < 2^\tau$) s.t.*
$c = \mathcal{H}(g_1|| \ldots ||g_\ell||h_1|| \ldots ||h_\ell||(g_1 \ldots g_\ell)^s (h_1 \ldots h_\ell)^c)$.

U computes $c = \mathcal{H}(g_1|| \ldots ||g_\ell||h_1|| \ldots ||h_\ell||(g_1 \ldots g_\ell)^k)$ for a randomly chosen k ($-2^{\epsilon(\tau+\kappa)} \le k < 2^{\epsilon(\tau+\kappa)}$), and sets $s = k - cx$. Denote it by: $PK[x : h_1 = g_1^x \wedge \cdots \wedge h_\ell = g_\ell^x]$.

Defn 3.3 (Proof of knowledge of a representation) *U proves knowledge of values $x_1, \ldots, x_\ell$ (with $0 \leq x_i < 2^\kappa$) s.t. a given element A satisfies $A = g_1^{x_1} \cdots g_\ell^{x_\ell}$, by releasing s_i and c ($-2^{\epsilon(\tau+\kappa)+1} \leq s_i < 2^{\epsilon(\tau+\kappa)+1}$; $0 \leq c < 2^\tau$) s.t. $c = \mathcal{H}(g_1||\ldots||g_\ell||A||g_1^{s_1}\ldots g_\ell^{s_\ell}A^c)$.*

Again, U computes $c = \mathcal{H}(g_1||\ldots||g_\ell||A||g_1^{k_1}\ldots g_\ell^{k_\ell})$ for randomly chosen k_i ($-2^{\epsilon(\tau+\kappa)} \leq k_i < 2^{\epsilon(\tau+\kappa)}$), and sets $s_i = k_i - cx_i$. Denote it by: $PK[x_1, \ldots, x_\ell \,:\, A = g_1^{x_1} \cdots g_\ell^{x_\ell}]$.

The next two proofs of knowledge assert that a committed value lies in an interval. The first one was introduced in [CFT98b], and corrected in [CFT98a]. The second one, which uses the first as building block, was introduced in [Bou00], and is used in our scheme.

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.

Defn 3.4 (Commitment to a secret value) *Let x be a secret value held by U. Let g and h be two provably random generators of $\mathcal{G}$. $E = E(x,r) = g^x h^r$ 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 $\mathcal{E}$. Assume that the smallest integer λ s.t. $2^\lambda > n$ 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 and σ_2 be other security parameters. Assume further that $x < b$.

Defn 3.5 (Proof of knowledge of a committed value) *U can prove in ZK to a verifier V knowledge of a number x committed through $E = E(x,r) = g^x h^r$, by sending V a triple (c, D, D_1) satisfying: $c = \mathcal{H}(g||h||E||g^D h^{D_1} E^{-c} \mod n)$.*

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

Defn 3.6 (Proof of equality of two committed values) *U can prove in ZK 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:*

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

U generates the random values $t \in [1, 2^{\delta+\tau/2}b+1]$, $s_1 \in [1, 2^{\delta+\tau/2+\sigma}n-1]$, and $s_2 \in [1, 2^{\delta+\tau/2+\sigma_2}n-1]$. Next, U computes $W_1 = g_1^t h_1^{s_1} \mod n$, $W_2 = g_2^t h_2^{s_2} \mod n$; and sets $c = \mathcal{H}(g_1||h_1||g_2||h_2||E_1||W_1||W_2)$. Finally, U computes $D = t + cx$, $D_1 = s_1 + cr_1$, $D_2 = s_2 + cr_2$ (in **Z**). Denote this by $PK[x, r_1, r_2 : E_1 = E_1(x, r_1) \wedge E_2 = E_2(x, r_2)]$.

Defn 3.7 (Proof that a committed number is a square) *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*

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

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 $PK[x, r_2, r_3 : F = g^x h^{r_2} \wedge E = F^x h^{r_3}]$. Denote this by $PK[x, r_1 : E = E(x^2, r_1)]$.

Defn 3.8 (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 $D_1 \in [cb, 2^{\delta+\tau/2}b-1]$, and*

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

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 $C = \mathcal{H}(g||h||E||W)$, and $c = C \mod 2^{\tau/2}$; and sets $D_1 = s + cx$, $D_2 = t + cr$, repeating the procedure from the beginning if $D_1 \notin [cb, 2^{\delta+\tau/2}b-1]$. Denote the above by $PK_{CFT}[x, r : E = E(x, r) \wedge x \in [-2^{\delta+\tau/2}b, 2^{\delta+\tau/2}b]]$.

Defn 3.9 (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: $PK[x, r : E = E(x, r)]$, $PK[\tilde{x}_1, \tilde{r}_1 : \tilde{E}_1 = E(\tilde{x}_1^2, \tilde{r}_1)]$, $PK[\bar{x}_1, \bar{r}_1 : \bar{E}_1 = E(\bar{x}_1^2, \bar{r}_1)]$, $PK_{CFT}[\tilde{x}_2, \tilde{r}_2 : \tilde{E}_2 = E(\tilde{x}_2, \tilde{r}_2) \wedge \tilde{x}_2 \in [-\alpha, \alpha]]$, where $\tilde{E}_2 = \frac{E}{g^a \tilde{E}_1} \mod n$, $PK_{CFT}[\bar{x}_2, \bar{r}_2 : \bar{E}_2 = E(\bar{x}_2, \bar{r}_2) \wedge \bar{x}_2 \in [-\alpha, \alpha]]$, where $\bar{E}_2 = \frac{g^b}{E\bar{E}_1} \mod n$.*

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. 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 needed in the scheme is the verifiable Elgamal encryption of an exponent.

Defn 3.10 (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.*

Denote it by: $PK[r : \omega = \chi^r \wedge A = r^{-1}y^a \wedge B = g^a]$. The proof can be found in [Sta96], and it is repeated 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

$$c = \mathcal{H}(\chi \,||\, \omega \,||\, A \,||\, B \,||\, g_1 \,||\, \omega_1 \,||\, \cdots \,||\, g_\nu \,||\, \omega_\nu). \tag{7}$$

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 $\omega_i = (\omega^{c_i}\chi^{1-c_i})^{y_i'}$, and checks that (7) 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 $2^{-\nu}$.

4 Construction

In this section, the scheme is described more concretely, starting with $\mathcal{T}$, the set of shared public parameters. $\mathcal{T}$ specifies security parameters δ, ϵ, σ, 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. (Notation as in section §3.) $\mathcal{T}$ also specifies an arithmetic group $\mathcal{G}$ and three generators g, g_1 and g_2 of $\mathcal{G}$.

4.1 Choice of shared parameters

In this section 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 [CS00, JPV00] 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.

Table 3: Shared parameters

Shared parameters
Security parameters: δ, ϵ, σ, 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 $\mathcal{G}$.

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 defined as the quadratic residue subgroup of the multiplicative residues modulo n. The order of $\mathcal{E}$ is the universally unknown number $\phi(n)/4$. Group managers of competing organizations may all share the same modulus n, as the operation of the scheme does not require *anybody* to know the RSA trapdoor associated to n, and the trusted party may safely forget the factorization at its discretion.

The above public parameters can be further certified if so desired: Proofs of primality for p and q can be provided. As for g, g_1, and g_2, it is easy to test that each is not congruent to 0 or 1 modulo p, or to verify that each is a square, by computing the Legendre symbol[1] and checking if: $\left(\frac{g}{p}\right) = \left(\frac{g_1}{p}\right) = \left(\frac{g_2}{p}\right) = 1$.

4.2 Setup

Table 4: Group parameters:

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

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 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.

4.3 Join

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 PK of the logarithm to basis g_1 of this pseudonym (see definition 3.1), 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.

Table 5: The `JOIN` protocol

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

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

[1]The Legendre symbol of a non-zero residue a modulo p has value 1 if a is a quadratic residue and -1 if it is a non-quadratic. It can be computed efficiently [BS96].

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:

$$c = p - 2^{\sigma+\tau/2+2}\sqrt{p}, \tag{8}$$

and repeating the process of computing other random k and r until such an r is found. Note that $r < c$ with overwhelming probability in a single attempt, because since the quadratic residues are nearly uniformly distributed in the interval $[1, p-1]$, one has that $r < c$ with probability close 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. This very minor restriction on the possible values of r reflects requirements of the proof of a logarithm lying a specified interval (defn. 3.9), as shall be seen 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.

4.4 Sign and verify

Now the protocol SIGN is described. One goal of this protocol is that U convince a verifier V of its knowledge of a membership certificate (r, s) as above.

As in section §2, the signer chooses random ℓ, and ℓ', with $0 < \ell, \ell' < q$.

U releases the Elgamal encrypted pairs:

$$(Y_1, Y_2) = (I_U y_2^{\ell'}, g_2^{\ell'}); \quad (R_1, R_2) = (r^{-1} y_2^{\ell}, g_2^{\ell});$$

Next, U demonstrates that the pseudonym I_U is encrypted by the pair (Y_1, Y_2), and proves knowledge of the pseudonym secret u, by executing $PK[u, \ell' : Y_1 = g_1^u y_2^{\ell'} \ \wedge \ Y_2 = g_2^{\ell'}]$ (proofs 3.3 and 3.2). 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 $E_1 = E_1(r, 0) = \chi^r$ (definition 3.4). In the language of section §3, 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 (again definition 3.4) 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 $R_1 Y_1 = I_U r^{-1} y_2^{\ell+\ell'} = y^r g^s y_2^{\ell+\ell'}$ is also a commitment to the value r in the group $\mathcal{G}$, with generators y, g, and y_2. Denote it by $E_3 = R_1 Y_1$.

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. U executes two proofs of equality of two committed values (def. 3.6). In the first proof U sends V a triple (c', D', D'_1) satisfying:

$$c' = \mathcal{H}(\chi||\gamma||\beta||E_1||E_2||\chi^{D'} E_1^{-c'} \bmod \hat{p}||\gamma^{D'} \beta^{D'_1} E_2^{-c'} \bmod n).$$

In agreement with the notation in section §3, 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 quintuple (c, D, D_1, D_2, D_3) satisfying:

$$c = \mathcal{H}(\gamma||\beta||y||g||y_2||E_2||E_3||\gamma^D \beta^{D_1} E_2^{-c} \bmod n||y^D g^{D_2} y_2^{D_3} E_3^{-c} \bmod p||g_2^{D_3}(Y_2 R_2)^{-c} \bmod p).$$

Denote that by $PK[r, s, s_2, t : E_2 = E_2(r, s_2) \ \wedge \ E_3 = E_3(r, s, t) \ \wedge \ Y_2 R_2 = g_2^t]$.

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

$$r \equiv r_1 \mod p, \text{ and } r \equiv r_3 \mod q, \tag{9}$$

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. (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.) 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 (9). 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 (9) 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 9. 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. (3.9): $PK[r, s_2 : E_2 = E_2(r, s_2) \wedge r \in [-2^{\delta+\tau/2+1}\sqrt{c}, c+2^{\delta+\tau/2+1}\sqrt{c}]]$. 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 equation 8).

Finally, U must show that the exponent committed in E_1 equals the value encrypted in the pair (R_1, R_2), by executing (definition 3.10): $PK[r, t \ : \ E_1 = \chi^r \ \wedge \ R_1 = r^{-1} y_2^t \wedge W_2 = g_2^t]$. 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 $\mathcal{M}$ to be signed is used as an extra input of the hash function.

The protocol is summarized in table 6. Moreover, algorithm VERIFY can be derived immediately from the above formal description of SIGN as a proof of knowledge of a group certificate.

4.5 Open

As for OPEN, it is enough that the group manager decrypts the pair (Y_1, Y_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 (Y_1, Y_2) without revealing the group secret x, by proving that the discrete logarithm of the Y_1/I_U with respect to y_2 equals the logarithm of Y_2 with respect to g_2: $PK[x : Y_1 I_U^{-1} = Y_2^x \wedge \ y_2 = g_2^x]$. This constitutes the publicly verifiable *proof of authorship* of the signature.

Table 6: The `SIGN` protocol

<u>`Proof arguments:`</u>

Y_1, Y_2, R_1, R_2, χ, γ, β, E_1, and E_2.

<u>`Signature of knowledge:`</u>

$$SPK[u,\ell',\ell,r,s,s_2,t : Y_1 = g_1^u y_2^{\ell'} \ \wedge\ Y_2 = g_2^{\ell'}$$
$$\wedge\ E_1 = E_1(r,0) = \chi^r \ \wedge\ R_1 = r^{-1} y_2^{\ell} \wedge R_2 = g_2^{\ell}$$
$$\wedge\ E_2 = E_2(r,s_2) = \gamma^r \beta^{s_2} \ \wedge\ r \in [-2^{\delta+\tau/2+1}\sqrt{c}, c + 2^{\delta+\tau/2+1}\sqrt{c}]$$
$$\wedge\ E_3 = E_3(r,s,t) = Y_1 R_1 = y^r g^s y_2^t \ \wedge\ Y_2 R_2 = g_2^t\,](\mathcal{M})$$

5 An alternative construction in the RSA ring

In the previous construction, it was necessary to use an auxiliary group $\mathcal{E}$ of unknown order for the single purpose of utilizing the proof of knowledge of an exponent in a specified interval, definition 3.9. That further introduced the need for an extra proof of equality of committed logarithms in groups of different orders 3.6 (between $\mathcal{E}$ and $\mathcal{F}$).

On the other hand, if one chooses the original group $\mathcal{G}$ to be a group of unknown order, then it is not necessary to use the extra group $\mathcal{E}$. In fact, the proof of equality of logarithms committed in different groups (3.6) is sufficient to complete the "transfer" of the value r in the exponent of y to the value r in the exponent of χ. It is impossible for a malicious adversary to commit a different value as the exponent of χ and still pass the verification, as that would entail knowing another value congruent to r modulo the order of y, which divides the unknown group order.

Therefore it is possible to eliminate two steps in the `SIGN` protocol (with corresponding simplifications of the `VERIFY` protocol) if $\mathcal{G}$ is chosen to have unknown order, with considerable efficiency gains. In this section a brief description is given of an alternative realization of the scheme.

5.1 Alternative setup

Table 7: Shared parameters.

<u>Shared parameters</u>

`Security parameters` δ, ϵ, σ_1, σ_2, τ `(integers);`

`Secure hash function` $\mathcal{H}(\cdot) : \{0,1\}^* \longrightarrow \{0,1\}^\tau$`;`

n`, composite integer, the product of safe primes;`

$\hat{p}$`, 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`, (optional) proof that` n `is a product of safe primes;`

g, g_1`, and` g_2`, generators of` $\mathcal{G}$`;`

P'`, (optional) proof that` g, g_1, g_2 `are quadratic residues.`

Much of the notation and procedures are the same as in section 4. 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.

Table 8: Group-specific parameters.

Group-specific parameters
$\mathcal{S}$, a string including y and y_2;
CA's signature CERT$_{CA}(\mathcal{S})$.

5.2 Join

Table 9: The JOIN protocol.

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

The JOIN protocol is unchanged, except that there are no restrictions on the value of $r = I_U g^{-k} \mod n$, where k is chosen in the interval $[-2^{\tau+2\kappa}, 2^{\tau+2\kappa} - 1]$, for there is no necessity to use the proof 3.9. (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.

5.3 Sign, verify and open

As mentioned above, 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 mod n and r_2 mod $\phi(n)$ while satisfying the signature equation, because $\phi(n)$ is unknown.

Protocol OPEN is unchanged from the previous case – the encryption scheme is still Elgamal, and the verifiable decryption protocol is formally the same, with the difference that all the proof values are taken in the integers as opposed to reduced modulo the (now unknown) order.

Table 10: The `SIGN` protocol.

`Proof arguments:`

$Y_1,\ Y_2,\ R_1,\ R_2,\ \chi,\ E_1.$

`Signature of knowledge:`

$$SPK[u, \ell', \ell, r, s, t : Y_1 = g_1^u g^{\ell'} \ \wedge\ Y_2 = g_2^{\ell'}$$
$$\wedge\ E_1 = E_1(r, 0) = \chi^r \ \wedge\ R_1 = r^{-1} y_2^{\ell} \ \wedge\ R_2 = g_2^{\ell}$$
$$\wedge\ E_2 = Y_1 R_1 = E_2(r, s, t) = y^r g^s y_2^t \ \wedge\ Y_2 R_2 = g_2^t](\mathcal{M})$$

6 Security analysis

The security of the modified Nyberg-Rueppel signature under active attacks is equivalent to the statement that the group membership certificates are unforgeable under active attacks. Therefore the security analysis of the scheme depends on the following assumption:

Assumption 6.1 (Modified Nyberg-Rueppel) *The modified Nyberg-Rueppel signature scheme, as a signature scheme on short messages, is existentially unforgeable under chosen message attacks.*

In [AdM04] we provide a proof of security for the modified Nyberg-Rueppel, in the generic model of computation. The generic model is an idealized computational model (such as the random oracle model) wherein an attacker may only access the group operations as black box function calls, and may not operate meaningfully on encodings of group elements, except with negligible probability. While this model cannot capture the most efficient attacks on the discrete logarithm and other assumptions underlying the constructions in this paper, it does provide evidence that the underlying computational problems are hard.

Correctness: The `SIGN` protocol was defined as a combination of various proofs of knowledge, with the `VERIFY` algorithm simply performing the combined verification of each of the individual proofs. Correctness follows.

Unforgeability: As seen above, one may assume that the group membership certificates are unforgeable. This closes one avenue of attack, which would exploit a weakness of the `JOIN` protocol. Therefore the question of unforgeability reduces to whether it is possible to attack the `SIGN` protocol.

First note that the zero-knowledge proof of knowledge of a representation implies that the term (Y_1, Y_2) truly encrypts a public key: $(I_U y_2, g_2^{\ell'})$, for which the signer knows the corresponding secret. This follows from the proof that definition 3.3 is indeed a proof a knowledge. Similarly, the proof of knowledge of the a representation shows that the signer knows how to write the product cipher $(R_1 Y_1, R_2 Y_2)$ as a commitment to values r, s in the exponents of y and g : $(y^r g^s y_2^{\ell+\ell'}, g_2^{\ell+\ell'})$. The proof of equality of logarithms in distinct groups shows that the term $E_1 = \chi^r$ really commits to the same value r. Finally the proof of encryption of a committed logarithm reveals that the value committed in E_1 is also encrypted in the pair $(R_1, R_2) = (r^{-1} y_2^{\ell}, g_2^{\ell})$. Now returning to the product cipher, it follows that the signer knows how to represent the encrypted value either as a product $I_U r^{-1}$ or as $y^r g^s$, and consequently that the signer knows a certificate on a public key for which he knows the corresponding secret. This implies that the signer indeed owns a valid certificate.

Anonymity/Unlinkability: The SIGN protocol releases the signer's public key I_U in an Elgamal encryption, and the value r in three separate commitments. These commitments are either Elgamal encryptions, such as in the pair (R_1, R_2), or under randomized bases, such as $E_1 = \chi^r$. Within each group ($\mathcal{E}$, $\mathcal{F}$, $\mathcal{G}$), correlating these values across multiple instances of the protocol is equivalent to breaking the Decisional Diffie-Hellman assumption. As far as we could determine, there are no known methods that allow for correlating exponents in different groups, and it is reasonable to assume that this would be a harder task than Diffie-Hellman decision problem itself. Therefore, under the assumption that the various proofs of knowledge compose well – i.e., remain zero knowledge when executed in combination – it follows that different executions of the protocol are non-correlatable, implicating both the anonymity and unlinkability properties.

Exculpability: As seen in the discussion about the unforgeability property above, the signer proves, during the execution of the SIGN protocol, knowledge of the secret key associated to the public key bound to a group membership certificate. During the OPENing phase, the group manager can only recover the same public key committed in SIGN, as the the former protocol is a verifiable Elgamal decryption. Finally, the group manager cannot claim that the recovered public key belongs to a different user: The owner of the public key is the unique user who provided a publicly verifiable, signed request to JOIN the group under that public key.

Traceability/Coalition-resistance: As seen in the discussion of the unforgeability property, the SIGN protocol is only successful if the signer proves knowledge of the secret key associated to the public key encrypted in the pair (Y_1, Y_2). Therefore it is not possible for a coalition to incriminate a user (unless the user key is compromised). Moreover, since certificates are unforgeable and the SIGN protocol proves the signer's knowledge of a certificate on the same public key, it follows that the signer commits to a legitimate user identity and knows the corresponding signing key.

7 Conclusions

In this paper describes the first group signature scheme with constant-size parameters that does not require any group members, including group managers, to know trapdoor secrets. The scheme is not bound to a specific setting but it can work in various groups where the Decision Diffie-Hellman assumption holds.

This scheme is less efficient than the state-of-the-art scheme in [ACJT00]. However, the scheme in [ACJT00] requires the group manager to know trapdoor information which cannot be shared with other group managers, thus making it difficult to enable collaboration among distinct groups.

References

[ACJT00] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Advances in Cryptology: Proceedings of*

CRYPTO'00, volume 1880 of *Lecture Notes in Computer Science*, pages 255–270. Springer-Verlag, 2000.

[AdM04] Giuseppe Ateniese and Breno de Medeiros. Security of a nyberg-rueppel signature variant. Technical Report 093, International Association for Cryptologic Research (IACR), 2004. `http://eprint.iacr.org/2004/093/`.

[ASW98] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. In *Advances in Cryptology: Proceedings of EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 591–606. Springer-Verlag, 1998.

[Ate99] G. Ateniese. Efficient verifiable encryption (and fair exchange) of digital signatures. In *Proceedings of the 6th ACM CCS*, pages 138–146. ACM Press, 1999.

[BMW03] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *Advances in Cryptology: Proceedings of EUROCRYPT '03*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629. Springer-Verlag, 2003.

[Bou00] F. Boudot. Efficient proofs that a committed number lies in an interval. In *Advances in Cryptology: Proceedings of EUROCRYPT '00*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444. Springer-Verlag, 2000.

[BS96] Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory, Volume I: Efficient Algorithms*, chapter 5.8. MIT Press, first edition, 1996.

[CD00] J. Camenisch and I. Damgård. Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In *Advances in Cryptology: Proceedings of ASIACRYPT'00*, volume 1976 of *Lecture Notes in Computer Science*, pages 331–ff. Springer-Verlag, 2000.

[CE87] D. Chaum and J. Evertse. A secure and privacy-protecting protocol for transmitting personal information between organizations. In *Advances in Cryptology: Proceedings of CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 118–167. Springer-Verlag, 1987.

[CFT98a] A. Chan, Y. Frankel, and Y. Tsiounis. Easy come – easy go divisible cash. updated version with corrections. In *GTE Technical Report*, 1998. `http://www.ccs.neu.edu/home/yiannis`.

[CFT98b] A. Chan, Y. Frankel, and Y. Tsiounis. Easy come, easy go divisible cash. In *Advances of Cryptology: Proceedings of EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 561–ff. Springer-Verlag, 1998.

[Cha85] D. Chaum. Security without identification: Transactions systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[Che95] L. Chen. Access with pseudonyms. In *Cryptography: Policy and Algorithms, Proceedings of an International Conference*, volume 1029 of *Lecture Notes in Computer Science*, pages 232–243, 1995.

[CL01] J. Camenisch and A. Lysyanskaya. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In *Advances in Cryptology: Proceedings of EUROCRYPT'01*, volume 2045 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.

[CS97] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *Advances in Cryptology: Proceedings of CRYPTO'97*, volume 1296 of *Lecture Notes in Computer Science*, pages 410–424. Springer-Verlag, 1997.

[CS98] J. Camenisch and M. Stadler. A group signature scheme with improved efficiency. In *Advances in Cryptology: Proceedings of ASIACRYPT'98*, volume 1514 of *Lecture Notes in Computer Science*, pages 160–174. Springer-Verlag, 1998.

[CS00] R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. *ACM Transactions on Information and System Security (TISSEC)*, 3(3):161–185, 2000.

[CvH91] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology: Proceedings of EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer-Verlag, 1991.

[Dam88] I. Damgård. Payment systems and credential mechanisms with provable security against abuse by individuals. In *Advances in Cryptology: Proceedings of CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 328–335. Springer-Verlag, 1988.

[FS87a] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology: Proceedings of CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer-Verlag, 1987.

[FS87b] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology: Proceedings of CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer-Verlag, 1987.

[JPV00] M. Joye, P. Paillier, and S. Vaudenay. Efficient generation of prime numbers. In *Proceedings of Cryptographic Hardware and Embedded Systems – CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 340–354. Springer-Verlag, 2000.

[KP98] J. Killian and E. Petrank. Identity escrow. In *Advances in Cryptology: Proceedings of CRYPTO'98*, volume 1642 of *Lecture Notes in Computer Science*, pages 169–185. Springer-Verlag, 1998.

[LRSW99] A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. *Selected Areas in Cryptography*, 1999.

[NR94] K. Nyberg and R. A. Rueppel. Message recovery for signature schemes based on the discrete logarithm problem. In *Advances in Cryptology: Proceedings of EUROCRYPT'94*, 1994.

[Sta96] M. Stadler. Publicly verifiable secret sharing. In *Advances in Cryptology: Proceedings of EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 190–199. Springer-Verlag, 1996.