Basic Research in Computer Science

BRICS RS-98-27 Camenisch & Michels: A Group Signature Scheme Based on an RSA-Variant

A Group Signature Scheme Based on an RSA-Variant

Jan Camenisch
Markus Michels

BRICS Report Series **RS-98-27**

ISSN 0909-0878 **November 1998**

See back inner page for a list of recent BRICS Report Series publications.
Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark

Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/98/27/`

A Group Signature Scheme Based on an RSA-Variant*

Jan Camenisch†
BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade
DK – 8000 Århus C, Denmark
camenisch@brics.dk

Markus Michels ‡
Entrust Technologies
r3 security engineering ag
Glatt Tower, 6th floor
CH – 8301 Glattzentrum,
Switzerland
Markus.Michels@entrust.com

November 3, 1998

Abstract

The concept of group signatures allows a group member to sign messages anonymously on behalf of the group. However, in the case of a dispute, the identity of a signature's originator can be revealed by a designated entity. In this paper we propose a new group signature scheme that is well suited for large groups, i.e., the length of the group's public key and of signatures do not depend on the size of the group. Our scheme is based on a variation of the RSA problem called strong RSA assumption. It is also more efficient than previous ones satisfying these requirements.

Keywords. Group signature scheme for large groups, digital signature scheme, revocable anonymity.

1 Introduction

In 1991 Chaum and van Heyst put forth the concept of a group signature scheme [16]. Participants are group members, a membership manager, and a revocation manager[1]. A group signature scheme allows a group member to sign messages anonymously on behalf of the group. More precisely, signatures can be verified with respect to a single public key of the group and do not reveal the identity of the signer. The membership manager is responsible for the system setup and for adding group

*A preliminary version of this paper appeared in [8].

†Part of this work was done while this author was with ETH Zurich.

‡Part of this work was done while this author was with Ubilab, UBS, Switzerland.

[1] In the original proposal, the membership manager and the revocation manager were a single entity called group manager.

members while the revocation manager has the ability to revoke the anonymity of signatures.

A group signature scheme could for instance be used by an employee of a large company to sign documents on behalf of the company. In this scenario, it is sufficient for a verifier to know that some representative of the company has signed. Moreover, in contrast to when an ordinary signature scheme would be used, the verifier does not need to check whether a particular employee is allowed to sign contracts on behalf of the company, i.e., he needs only to know a single company's public key. A further application of group signature schemes is electronic cash as was pointed out in [29]. In this scenario, several banks issue coins, but it is impossible for shops to find out which bank issued a coin that is obtained from a customer. Hence, the central bank plays the role of the membership and the revocation manager and all other banks issuing coins are group members. The identification as a group member is another application, e.g., in order to get access to a restricted area [25].

Various group signature schemes have been proposed so far. However, in the schemes presented in [5, 16, 17, 33] the length of signatures and/or the size of the group's public key depend on the size of the group and thus these schemes are not suitable for large groups. Only in the two schemes presented in [9, 10] (and the blind versions thereof [29]) are the length of signatures and the size of the group's public key independent of the number of group members[2]. The schemes presented in [25] satisfy the length requirement as well, but these are inefficient.

In this paper we propose a new group signature scheme for which the length of signatures and the size of the group's public key do not depend on the size of the group. The security of our scheme relies on a variant of the RSA assumption, called *strong RSA-assumption* and proposed in [1, 22], the discrete logarithm assumption the Diffie-Hellman decision assumption. Compared to the solutions in [9, 10], our scheme is based on a different number-theoretic assumption and is also more efficient.

2 The Model and an Approach for a Realization

2.1 The Model

A group signature scheme consists of the following algorithms:

`setup`: An interactive setup protocol between the membership manager, the group members, and the revocation manager. The public output is the group's public key Y. The private outputs are the individual secret keys x_G for each group member, the secret key x_M for the membership manager, and the secret key x_R for the revocation manager.

`sign`: A signature generation algorithm that on input a message m, an individual group member's secret key x_G, and the group's public key Y outputs a signature σ.

`verify`: A verification algorithm that on input a message m, a signature σ, and the group's public key Y returns 1 if and only if σ was generated by any group member using `sign` on input x_G, m, and Y.

[2] The other schemes [26, 32] with the same properties were shown to be flawed [28, 30].

`tracing`: A tracing algorithm that on input a signature σ, a message m, the revocation manager's secret key x_R, and the group's public key Y returns the identity *ID* of the group member who issued the signature σ together with an argument *arg* of this fact.

`vertracing`: A tracing-verification algorithm that on input a signature σ, a message m, the group's public key Y, the identity *ID* of a group member, and an argument *arg* outputs 1 if and only if *arg* was generated by `tracing` with respect to m, σ, Y, x_R.

The following informally stated security requirements must hold:

Unforgeability of signatures: Only group members are able to sign messages. Furthermore, they must only be able to sign in such a way that, when the signature is (later) presented to the revocation manager, he will be able to reveal the identity of the signer.

Anonymity of signatures: It is not feasible to find out the group member who signed a message without knowing the revocation manager's secret key.

Unlinkability of signatures: It is infeasible to decide whether two signatures have been issued by the same group member or not.

No framing: Even if the membership manager, the revocation manager, and some of the group members collude, they cannot sign on behalf of non-involved group members.

Unforgeability of tracing verification: The revocation manager cannot accuse a signer falsely of having originated a given signature, e.g., by issuing an argument *arg* such that `vertracing` outputs 1 if input another *ID* than the one of the signer.

The efficiency of a group signature scheme can be measured by the size of the public key Y, the length of signatures, and by the efficiency of the algorithms `sign`, `verify`, `setup`, `tracing`, and `vertracing`.

2.2 The Approach of Camenisch and Stadler

The core idea of the schemes proposed in [9, 10] is the following. A group's public key consists of the membership manager's public key of an ordinary digital signature scheme and the revocation manager's public key of a probabilistic encryption scheme. A user, say Alice, who wants to join the group chooses a random *secret key* x_G and computes her *membership key* $z := f(x_G)$, where f is a suitable one-way function. Alice commits to z (for instance by signing it) and sends z and her commitment to the membership manager M who returns her a *membership certificate* $u := \mathtt{sig}_M(z)$.

To sign a message m on behalf of the group, Alice encrypts z using the public key of the revocation manager (let c denote this ciphertext) and issues a *signature of knowledge*[3] [9] that she knows some values $\tilde{x}$ and $\tilde{u}$ such that $\tilde{u} = \mathtt{sig}_M(f(\tilde{x}))$ holds

[3]These are message dependent non-interactive arguments derived from 3-move honest-verifier zero-knowledge proofs of knowledge using the Fiat-Shamir heuristic [20, 21].

and that $f(\tilde{x})$ is encrypted in c. The verification of such a group-signature is done by checking this signature of knowledge. The revocation manager can easily revoke the anonymity of a group-signature by decrypting c and forwarding this value to the membership manager.

To realize a concrete scheme along these lines, one has to find a suitable one-way function f and a suitable signature scheme that yield an efficient signature of knowledge for the values $\tilde{x}$ and $\tilde{u}$. In [9, 10], two proposals based on different number theoretic assumption were put forth. The first assumption is that given e, g, and an RSA-modulus n, it is hard to find integers u and x such that $u^e \equiv g^x + 1 \pmod{n}$ holds, where g is an element of large order. The second one is that it is hard to find integers u and x with $|x| < |n|/2$ such that $u^3 \equiv x^5 + v \pmod{n}$ when given a suitably chosen integer v and an RSA-modulus n.

3 Number Theoretic Assumptions

In this section we describe the assumptions the security of our group signature schemes is based upon. In particular, we will introduce an alternative to the assumptions discussed in the previous section that allows the construction of a new group signature scheme and will show that this assumption can be reduced a variant of the RSA assumption.

Recently, Fujisaki and Okamoto [22] proposed a variation of the well-known RSA assumption [36]: the so-called *strong RSA assumption*. Let ℓ_g be a security parameter and let $\mathcal{G}(\ell_g)$ denote the set of groups whose order has length ℓ_g and consists of two prime factors of length $(\ell_g - 2)/2$.

Problem 1 (Strong RSA Problem). *Given* G *and* $z \in G/\{\pm 1\}$, *find a pair* $(u, e) \in G \times \mathbb{Z}$ *such that* $u^e = z$ *and* $e > 1$.

Let K denote a key-generator that on input 1^{ℓ_g} outputs a $G \in \mathcal{G}(\ell_g)$ and a $z \in G/\{\pm 1\}$.

Assumption 1 (Strong RSA Assumption). *There exists a probabilistic algorithm* K *such that for all probabilistic polynomial-time algorithms* A, *all polynomials* $p(\cdot)$, *all sufficiently large* ℓ_g

$$Pr[z = u^e \wedge e > 1 \,:\, (G, z) := K(1^{\ell_g}),\ (u, e) := A(G, z)] < \frac{1}{p(\ell_g)} \,.$$

In an implementation of the key-generator K, where G is chosen to be $\mathbb{Z}_n^*$ and n is an RSA modulus, the parameter z should not be chosen as a power in $\mathbb{Z}$.

Let us focus on a slight modification of this assumption. Let k, ℓ_1, $\ell_2 < \ell_g$, and $\epsilon > 1$ be further security parameters. For simplicity let denote $\tilde{\ell} := \epsilon(\ell_2 + k) + 1$. Let be $\mathcal{M}(G, z) = \{(u, e) \mid z = u^e, u \in G, e \in \{2^{\ell_1} - 2^{\ell_2}, \ldots, 2^{\ell_1} + 2^{\ell_2}\}, e \in \text{primes}\}$, where $G \in \mathcal{G}(\ell_g)$ and $z \in G$.

Problem 2 (Modified Strong RSA Problem). *Given* G, $z \in G$, *and* $M \subset \mathcal{M}(G, z)$ *with* $|M| = \mathcal{O}(\ell_g)$ *find a pair* $(u, e) \in G \times \mathbb{Z}$ *such that* $u^e = z$, $e \in \{2^{\ell_1} - 2^{\tilde{\ell}}, \ldots, 2^{\ell_1} + 2^{\tilde{\ell}}\}$, *and* $(u, e) \notin M$.

Assumption 2 (Modified Strong RSA Assumption). *There exists a probabilistic algorithm* K *such that for all probabilistic polynomial-time algorithms* A*, all polynomials* $p(\cdot)$*, all sufficiently large* ℓ_g*, all* $M \subset \mathcal{M}(G, z)$ *with* $|M| = \mathcal{O}(\ell_g)$*, and suitably chosen* ℓ_1*,* ℓ_2*,* k*, and* ϵ

$$Pr[z = u^e \ \wedge \ e \in \{2^{\ell_1} - 2^{\tilde{\ell}}, \ldots, 2^{\ell_1} + 2^{\tilde{\ell}}\} \ \wedge \ (u, e) \notin M \ : \ (G, z) := K_1(1^{\ell_g}), \\ (u, e) := A(G, z, M)] < \frac{1}{p(\ell_g)} \ .$$

A similar assumption was proposed by Barić and Pfitzmann [1], i.e., they require e to be a prime but do not have any restriction an the sizes of the exponents. Possible choices for G are discussed in Section 5. Let us remark that, given u, e, $\tilde{u}$, and $\tilde{e}$ with $z = u^e = \tilde{u}^{\tilde{e}}$ and $\gcd(e, \tilde{e}) = 1$, it is easy to find an element $\bar{u}$ satisfying $z = \bar{u}^{e\tilde{e}}$ using the extended Euclidean algorithm. However, as $e\tilde{e} \notin \{2^{\ell_1} - 2^{\tilde{\ell}}, \ldots, 2^{\ell_1} + 2^{\tilde{\ell}}\}$ for suitable chosen parameter ℓ_g, ℓ_1, ℓ_2, ϵ, and k the integer $e\tilde{e}$ does not satisfy the range constraint, i.e., $\epsilon(\ell_2 + k) + 1 < \ell_1$ must hold.

Adapting methods from [38], it can be shown that Assumption 1 implies Assumption 2. That is, given a probabilistic polynomial-time algorithm A_2 that solves Problem 2 we construct a probabilistic polynomial-time algorithm that solves Problem 1. Let G and $z \in G$ be the given instance of Problem 1. Then choose arbitrary primes $e_1, \ldots, e_t$ for $t = \mathcal{O}(\ell_g)$ satisfying the range conditions and set $\tilde{z} := z^{e_1 \cdots e_t}$, $\tilde{u}_i = \tilde{z}^{1/e_i} := z^{e_1 \cdots e_{i-1} e_{i+1} \cdots e_t}$ for $i = 1, \ldots, t$, and $M := \{(u_i, e_i) \mid i \in \{1, \ldots, t\}\}$. Run A_2 on input $G, \tilde{z}, M$ and get $(\tilde{u}, \tilde{e})$ such that $\tilde{u}^{\tilde{e}} = \tilde{z}$ and $\tilde{e} \in \{2^{\ell_1} - 2^{\ell_2}, \ldots, 2^{\ell_1} + 2^{\ell_2}\}$. Now we have $\tilde{u}^{\tilde{e}} = \tilde{z} = z^{e_1 \cdots e_t}$. Because of the range condition and since all e_i's are prime, $\gcd(\tilde{e}, e_1 \cdots e_t) = 1$ holds. Thus two integers a and b such that $a\tilde{e} + b(e_1 \cdots e_t) = 1$ can be found efficiently and we can compute the pair $(u := z^a \tilde{u}^b, \tilde{e})$ which is a solution of the given instance of Problem 1. Hence Problem 2 is at least as hard as Problem 1.

Besides the strong RSA assumption, our group signature scheme relies further on the discrete logarithm (DL) assumption and so-called Diffie-Hellman decision (DHD) assumption. Since the latter is not so well known, we state it explicitly. Let $G \in \mathcal{G}(\ell_g)$, n' be the divisor of G's order of length $\ell_g - 2$. Define the two sets

$$\mathcal{DH}(G) := \{(g_1, y_1, g_2, y_2) \in G^4 \mid \text{ord}(g_1) = \text{ord}(g_2) = n', \ \log_{g_1} y_1 = \log_{g_2} y_2\}$$

$$\mathcal{Q}(G) := \{(g_1, y_1, g_2, y_2) \in G^4 \mid \text{ord}(g_1) = \text{ord}(g_2) = \text{ord}(y_1) = \text{ord}(y_2) = n'\}$$

of Diffie-Hellman and arbitrary 4-tuples, respectively.

Assumption 3 (Diffie-Hellman Decision Assumption). *There exists a probabilistic algorithm* K *such that for all probabilistic polynomial-time algorithms* A *and all sufficiently large* ℓ_g*, the two probability distributions*

$$Pr\left[a = 1 \ : \ G := K(1^{\ell_g}), T \in_R \mathcal{DH}(G), a := A(T)\right]$$

and

$$Pr\left[a = 1 \ : \ G := K(1^{\ell_g}), T \in_R \mathcal{Q}(G), a := A(T)\right]$$

are computationally indistinguishable.

Note that in the case $G = \mathbb{Z}_n^*$, where n is an RSA-modulus, the DHD assumption does not hold. The Jacobi-symbol, which can be computed efficiently without knowing the factorisation of n, leaks information about $\log_{g_1} y_1$ and $\log_{g_2} y_2$. For instance, if $(g_1|n) = (g_2|n) = (y_2|n) = -1$ and $(y_1|n) = 1$, then $\log_{g_1} y_1 \neq \log_{g_2} y_2$. This problem is overcome if $G = \langle g \rangle$ is defined to be a subgroup of $\mathbb{Z}_n^*$ with $(g|n) = 1$.

4 Building Blocks

In this section we introduce the building blocks for our scheme borrowing notation from [9, 10]. These building blocks are signature schemes derived from statistical (honest-verifier) zero-knowledge proofs of knowledge using the Fiat-Shamir heuristic [20, 21] and are therefore called "signatures based on a proof of knowledge", SPK for short. Usually, the security of such building blocks is argued by showing that the underlying interactive protocols is secure and then by assuming that "nothing bad happens" when the verifier is replaced with a collision resistant hash-function. This approach has been formalised as the random oracle model (e.g., see [2, 34])[4]. For the signer/prover security means that the protocol should be zero-knowledge and for the verifier it means that the protocol should be a proof of knowledge. An example of this method is the Schnorr signature scheme [37] that is derived from an honest-verifier zero-knowledge proof of knowledge of the discrete logarithm of the signer's public key.

In the following we describe four building blocks. The first one shows the knowledge of a discrete logarithm, the second the equality of two discrete logarithms, the third the knowledge of one out of two discrete logarithms, and the fourth the knowledge of a discrete logarithm that lies in a certain interval. Of course, these building blocks can be combined in a natural way (e.g., see [10]). The building blocks have in common that the prover does not know the order of G, i.e., the verifier chooses a group $G = \langle g \rangle$ of large order such that only he can know the order. However, the order of magnitude 2^{ℓ_g} of the group's order shall be known to both. Furthermore, the verifier chooses a second generator h and proves that g and h have order $p'q'$, where p' and q' are two primes of length $(\ell_g - 2)/2$ and that he does not know $\log_g h$. How this can be done is discussed in the next section. Since the group order is not publicly known, we define the discrete logarithm of an $y \in G$ to the base g to be any integer x such that $y = g^x$ holds. Finally, we assume a collision resistant hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^k$ (e.g., $k \approx 160$).

Before we define the building blocks let us explain the notation with the following example [9]: a signature based on a proof of knowledge, denoted

$$SPK\left\{(\alpha, \beta) : \ y = g^{\alpha} \ \wedge \ z = g^{\beta} h^{\alpha}\right\}(m),$$

is used for 'proving' the knowledge of the discrete logarithm of y to the base g and of a representation of z to the bases g and h, and in addition, that the h-part of this representation equals the discrete logarithm of y to the base g. This is equivalent to the knowledge of a pair (α, β) satisfying the equations on the right side of the

[4]Recently, it has be shown that this approach does not work for general protocols [11], i.e., there exist protocols (although specially designed ones) which are secure in the random oracle model but yield an insecure signature scheme. However, it is believed that the approach is still valid for the kind of protocols considered here.

colon. In the sequel, we use the convention that Greek letters denote the elements whose knowledge is proven and all other letters denote elements that are known to the verifier.

4.1 Showing the Knowledge of a Discrete Logarithm

The building block presented in this subsection is an adaption of the protocols for proving the knowledge of a discrete logarithm [14, 37] to the setting with a group of unknown order due to Girault [23, 24]. A consequence of this setting is that the usual knowledge extractor for showing that a protocol is a proof of knowledge does not work: the knowledge extractor does not know the group's order either and hence cannot compute inverses modulo this group order and therefore not extract the witness. Poupard and Stern [35] give a security proof for this adaption in a weaker security model, i.e., they show that if an attacker was able to carry out the protocol for almost all public keys, then he could also compute the discrete logarithm of the prover's public key. Since the latter is assumed to be impossible the protocol is concluded to be secure.

An alternative way of proving the security was proposed by Fujisaki and Okamoto [22]. They show that under Assumption 1 the knowledge extractor is able to extract witnesses without knowing the group's order. We will stick to this method in this paper.

Definition 1. *Let $\epsilon > 1$ be a security parameter. A pair $(c, s) \in \{0,1\}^k \times \{-2^{\ell_g+k}, \ldots, 2^{\epsilon(\ell_g+k)}\}$ satisfying $c = \mathcal{H}(g\|y\|g^s y^c\|m)$ is a signature of a message $m \in \{0,1\}^*$ with respect to y and is denoted $SPK\{(\alpha) : y = g^\alpha\}(m)$.*

An entity knowing the secret key $x \in \{0,1\}^{\ell_g}$ such that $x = \log_g y$ can compute such a signature $(c, s) = SPK\{(\alpha) : y = g^\alpha\}(m)$ of a message $m \in \{0,1\}^*$ by

- choosing $r \in_R \{0,1\}^{\epsilon(\ell_g+k)}$ and computing $t := g^r$,
- $c := \mathcal{H}(g\|y\|t\|m)$, and
- $s := r - cx$ (in $\mathbb{Z}$).

In [10] it is analysed how much information (t, c, s) gives about x depending on the choice of ϵ.

Lemma 1. *If Assumption 1 holds, then the interactive protocol corresponding to $SPK\{(\alpha) : y = g^\alpha\}(m)$ is a honest-verifier statistical zero-knowledge proof of knowledge of the discrete logarithm of y.*

Proof. To prove that the protocol is statistical honest-verifier zero-knowledge for any $\epsilon > 1$, we have to show that an honest verifier, i.e., one who chooses the challenge c uniformly random from $\{0,1\}^k$, can simulate a protocol-conversation that is statistically indistinguishable from a protocol-conversation with the prover. The following constitutes a simulator the verifier could use to do so.

The simulator randomly chooses c' from $\{0,1\}^k$ and s' from $\{0,1\}^{\epsilon(\ell_g+k)}$ according to the uniform distribution. Using these values, the simulator computes $t' = g^{s'} y^{c'}$. To prove that these values are statistical indistinguishable from a view of a protocol run with the prover, it suffices to consider the probability distribution

$P_S(s)$ of the response s of the prover and the probability distribution $P_{S'}(s')$ according to which the simulator chooses s'. The latter is the uniform distribution over $\{0,1\}^{\epsilon(\ell_g+k)}$.

If the prover chooses r uniformly at random from $\{0,1\}^{\epsilon(\ell_g+k)}$ and the secret key randomly from $\{0,1\}^{\ell_g}$ according to any distribution, we have

$$P_S(s) \begin{cases} = 0 & \text{for } s < -(2^k-1)(2^{\ell_g}-1) \\ \leq 2^{-\epsilon(\ell_g+k)} & \text{for } -(2^k-1)(2^{\ell_g}-1) \leq s < 0 \\ = 2^{-\epsilon(\ell_g+k)} & \text{for } 0 \leq s \leq 2^{\epsilon(\ell_g+k)} - (2^k-1)(2^{\ell_g}-1) \\ \leq 2^{-\epsilon(\ell_g+k)} & \text{for } 2^{\epsilon(\ell_g+k)} - (2^k-1)(2^{\ell_g}-1) < s \leq (2^{\epsilon(\ell_g+k)}-1) \\ = 0 & \text{for } (2^{\epsilon(\ell_g+k)}-1) < s. \end{cases}$$

This holds for any distribution of c over $\{0,1\}^k$. Thus we have

$$\sum_{\alpha\in\mathbb{Z}} |P_S(\alpha) - P_{S'}(\alpha)| \leq \frac{2(2^k-1)(2^{\ell_g}-1)}{2^{\epsilon(\ell_g+k)}} \leq \frac{2^{k+\ell_g+1}}{2^{\epsilon(\ell_g+k)}} \leq \frac{2}{(2^{(\ell_g+k)})^{(\epsilon-1)}}$$

For ℓ_g and k as stated in the theorem, the last term can be expressed as one over a polynomial in the input length, and therefore the two distributions are statistical indistinguishable.

Let us show that it is a proof of knowledge. Given the fact that the equivalent protocol (e.g., [37]) for groups of known order is a proof of knowledge it is sufficient two show that the knowledge extractor can compute the witness once he has found two accepting triples. Let (t, c, s) and $(t, \tilde{c}, \tilde{s})$ be these two accepting triples. Since $t = g^s y^c = g^{\tilde{s}} y^{\tilde{c}}$ holds we have $y^{c-\tilde{c}} = g^{\tilde{s}-s}$. Let $d := \gcd(c-\tilde{c}, \tilde{s}-s)$. Using the extended Euclidean algorithm we obtain values u and v such that $u\frac{c-\tilde{c}}{d} + v\frac{\tilde{s}-s}{d} = 1$ and hence we have

$$g = g^{u\frac{c-\tilde{c}}{d} + v\frac{\tilde{s}-s}{d}} = (g^u y^v)^{\frac{c-\tilde{c}}{d}}.$$

If $d < c - \tilde{c}$ then $g^u y^v$ is a $\frac{c-\tilde{c}}{d}$-root of g. Since this contradicts Assumption 1 we must have $d = c - \tilde{c}$, hence $c - \tilde{c}$ divides $\tilde{s} - s$, and we can compute the integer

$$x := \frac{\tilde{s}-s}{c-\tilde{c}}$$

such that $g^x = y$. □

4.2 Showing the Equality of Two Discrete Logarithms

The next SPK is an adoption of a protocol for showing the equality of two discrete logarithms given in [15] to the setting in which the group's order is unknown.

Definition 2. *Let $\epsilon > 1$ be a security parameter. A pair $(c, s) \in \{0,1\}^k \times \{-2^{\ell_g+k}, \ldots, 2^{\epsilon(\ell_g+k)}\}$ satisfying $c = \mathcal{H}(g\|h\|y_1\|y_2\|y_1^c g^s\|y_2^c h^s\|m)$ is a signature of a message $m \in \{0,1\}^*$ with respect to y_1 and y_2 and is denoted*

$$SPK\{(\alpha) : y_1 = g^\alpha \wedge y_2 = h^\alpha\}(m).$$

Let $x \in \{0,1\}^{\ell_g}$ be the secret key of the signer such that $y_1 = g^x$ and $y_2 = h^x$ holds. Then a signature $(c, s) = SPK\{(\alpha) : y_1 = g^\alpha \wedge y_2 = h^\alpha\}(m)$ of a message $m \in \{0,1\}^*$ can be computed as follows.

- Choose $r \in_R \{0,1\}^{\epsilon(\ell_g+k)}$ and compute $t_1 := g^r$, $t_2 := h^r$,
- $c := \mathcal{H}(g\|h\|y_1\|y_2\|t_1\|t_2\|m)$, and
- $s := r - cx$ (in $\mathbb{Z}$).

The security properties and proofs of this building block follow from the ones of the previous building block and from [15].

4.3 Showing the Knowledge of One Out of Two Discrete Logarithms

The realization of the following SPK of one out of two discrete logarithms is an adoption of a protocol given in [19].

Definition 3. *Let $\epsilon > 1$ be a security parameter. A tuple $(c_1, c_2, s_1, s_2) \in \{0,1\}^k \times \{0,1\}^k \times \{-2^{\ell_g+k}, \dots, 2^{\epsilon(\ell_g+k)}\} \times \{-2^{\ell_g+k}, \dots, 2^{\epsilon(\ell_g+k)}\}$ satisfying $c_1 \oplus c_2 = \mathcal{H}(g\|h\|y_1\|y_2\|y_1^{c_1}g^{s_1}\|y_2^{c_2}h^{s_2}\|m)$ is a signature of a message $m \in \{0,1\}^*$ with respect to y_1 and y_2 and is denoted*

$$SPK\{(\alpha, \beta) : y_1 = g^\alpha \vee y_2 = h^\beta\}(m).$$

Without loss of generality, we assume that the signer knows $x \in_R \{0,1\}^{\ell_g}$ such that $y_1 = g^x$ holds. Then a signature $SPK\{(\alpha, \beta) : y_1 = g^\alpha \vee y_2 = h^\beta\}(m)$ of a message $m \in \{0,1\}^*$ can be computed as follows.

- Choose $r_1 \in_R \{0,1\}^{\epsilon(\ell_g+k)}$, $r_2 \in_R \{0,1\}^{\epsilon(\ell_g+k)}$, and $c_2 \in_R \{0,1\}^k$ and compute $t_1 := g^{r_1}$, $t_2 := h^{r_2}y_2^{c_2}$,
- $c_1 := c_2 \oplus \mathcal{H}(g\|h\|y_1\|y_2\|t_1\|t_2\|m)$,
- $s_1 := r_1 - c_1 x$ (in $\mathbb{Z}$), and $s_2 := r_2$.

The security properties and proofs of this building block follow from the ones of the previous building blocks and from [19].

4.4 Showing That a Discrete Logarithm Lies in an Interval

The last building block is based on a proof that the secret the prover knows lies in a given interval. It is related to protocols presented in [13, 22].

Definition 4. *Let $\epsilon > 1$ be a security parameter and let $\ell_1 < \ell_g$ and ℓ_2 denote lengths. A pair $(c, s) \in \{0,1\}^k \times \{-2^{\ell_2+k}, \dots, 2^{\epsilon(\ell_2+k)}\}$ satisfying $c = \mathcal{H}(g\|y\|g^{s-c2^{\ell_1}}y^c\|m)$ is a signature of a message $m \in \{0,1\}^*$ with respect to y and is denoted*

$$SPK\big\{(\alpha) :\ y = g^\alpha \ \wedge\ (2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1} < \alpha < 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1})\big\}(m).$$

Such a signature of a message $m \in \{0,1\}^*$ with respect to a public key $y \in G$ can be computed as follows if an integer $x \in \{2^{\ell_1}, \dots, 2^{\ell_1} + 2^{\ell_2}\}$ is known such that $y = g^x$ holds:

- choose $r \in_R \{0,1\}^{\epsilon(\ell_2+k)}$, and compute $t := g^r$,

- $c := \mathcal{H}(g\|y\|t\|m)$, and
- $s := r - c(x - 2^{\ell_1})$ (in $\mathbb{Z}$).

Lemma 2. *If Assumption 1 holds and $\epsilon > 1$ then the interactive protocol corresponding to $SPK\{(\alpha) : y = g^{\alpha} \wedge (2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1} < \alpha < 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1})\}(m)$ is a statistical honest-verifier zero-knowledge proof of knowledge of an integer x such that $x \in \{2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1}\}$ and $y = g^x$.*

Sketch. The proof that the protocol is statistical honest-verifier zero-knowledge is similar as for Lemma 1.

Let us consider the proof-of-knowledge part. As in the proof of Lemma 1 the knowledge-extractor gets two accepting triples (t, c, s) and $(t, \tilde{c}, \tilde{s})$ from which he can similarly compute the integer

$$x := 2^{\ell_1} + \frac{\tilde{s} - s}{c - \tilde{c}}$$

such that $g^x = y$ since $c - \tilde{c}$ must divide $\tilde{s} - s$ if Assumption 1 holds. It remains to show that this integer lies in the claimed bounds. Due to Definition 4 the integers s and $\tilde{s}$ must lie in $\{-2^{\epsilon(\ell_2+k)}, \ldots, 2^{\epsilon(\ell_2+k)}\}$ and since the smallest value that $c - \tilde{c}$ can have is 1 the computed x must lie in $\{2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1}\}$. $\square$

Note that $\epsilon(\ell_2 + k) + 2 < \log(\mathrm{ord}(g)) \approx \ell_g$ should hold in order to indeed restrict the size of $\log_g y$.

5 The Proposed Scheme

In this section we propose a realization of a group signature scheme the security of which is based on Assumptions 2 and 3. The basic idea of the scheme is the following. The membership manager chooses a group $G = \langle g \rangle$ and a group element z such that Assumptions 2 and 3 hold. Furthermore, he chooses a second generator h such that $\log_g h$ is unknown. Computing discrete logs in G to the bases g, h, or z must be infeasible. Finally, computing roots in G must be feasible only to the membership manager, i.e., he should the only one who knows the order of G. The revocation manager chooses his secret key x and publishes $y = g^x$.

Each group member chooses a prime e randomly in a determined range together with the membership manager. Only the group member learns e and stores it as a secret key. A membership certificate issued by the membership manager is an element $u \in G$ such that $u^e = z$ holds. Here we slightly deviate from the approach of Camenisch and Stadler, i.e., the membership certificate and the membership key are the same number. As a consequence, the issuing of certificates must be realized in a way that the membership manager is not able to learn the group member's secret key e.

A signature of a message m by a group member consists of a triple $(a, b, d) \in G^3$ and an SPK of integers u and e such that

- the pair (a, b) is an encryption of u under the revocation manager's public key (which is part of the group public key)

- d commits to e,
- e lies in the necessary range, and
- $u^e = z$ holds.

The membership manager can reveal the identity of a signer by asking the revocation manager to decrypt (a, b).

The following paragraphs describe the new scheme in detail and provide security and efficiency analyses.

5.1 The Setup of the Scheme

The setup procedure of our scheme consists of two phases. In the first phase the membership manager and the revocation manager construct the group's public key and choose their secret keys. This is described in this subsection. In the second phase of the setup, the group members choose their membership secret keys and get their membership certificates. This phase is described in the next subsection.

The membership manager chooses a group $G = \langle g \rangle$ and two random elements $z, h \in G$ with the same (large) order ($\approx 2^{\ell_g}$) such that Assumptions 2 and 3 hold. He publishes z, g, h, G, and ℓ_g and proves that g, h, and z have the same order which is non-prime, of the order of magnitude 2^{ℓ_g}, and non-smooth (how this can be done will be discussed later). The membership manager must further prove that z and h where chosen at random. The revocation manager chooses his secret key x randomly in $\{0, \ldots, 2^{\ell_g} - 1\}$ and publishes $y = g^x$ as his public key. Finally, a hash function $\mathcal{H} : \{0,1\}^* \longrightarrow \{0,1\}^k$ and security parameters $\hat{\ell}$, ℓ_1, ℓ_2, and ϵ are set. An example for choosing the parameters ϵ, $\hat{\ell}$, ℓ_g, ℓ_1, and ℓ_2 is given in Section 5.6.

A possible choice of $G = \langle g \rangle$ is a subgroup of $\mathbb{Z}_n^*$ such that $(g|n) = 1$. In this case the membership manager chooses two large random primes p and q ($\approx 2^{\ell_g/2}$) of form $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are primes as well, such that $p, q \not\equiv 1 \pmod 8$ and $p \not\equiv q \pmod 8$ holds. He keeps p and q secret and publishes $n := pq$. For proving that n if indeed the product of two safe primes the method described in [7] could be used. Verifying that an element a has (large) order at least $p'q'$ in $\mathbb{Z}_n^*$ and Jacobi symbol 1 can done by anyone: one needs only to test whether $a \not\equiv \pm 1 \pmod n$ and $\gcd(a-1, n) = 1$ holds. An alternative choice of G is a suitable elliptic curve (e.g., see [27]).

5.2 The Registration of a Group Member

To become a group member Alice chooses a random prime $\hat{e} \in_R \{2^{\hat{\ell}-1}, \ldots, 2^{\hat{\ell}} - 1\}$ and $e \in_R \{2^{\ell_1}, \ldots, 2^{\ell_1} + 2^{\ell_2} - 1\}$ such that $\hat{e}, e \not\equiv 1 \pmod 8$ and $\hat{e} \not\equiv e \pmod 8$, Alice computes $\tilde{e} := e\hat{e}$ and $\tilde{z} := z^{\hat{e}}$, commits to $\tilde{e}$ and $\tilde{z}$ (for instance by signing them), sends $\tilde{e}$, $\tilde{z}$, and their commitments to the membership manager, and carries out the interactive protocols corresponding to

$$W := SPK\Big\{(\alpha, \beta) : z^{\tilde{e}} = \tilde{z}^{\alpha} \ \wedge\ \tilde{z} = z^{\beta} \ \wedge \\ (2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}) < \alpha < (2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1})\Big\}(\tilde{z}) \, ,$$

with the membership manager (cf. previous section). Furthermore, Alice proves the the membership manager that $\tilde{e}$ is the product of two primes (e.g., using the methods described in [4, 39]). Using the same arguments as for the building blocks in the previous section, it can be seen that the protocol corresponding to W convinces the membership manager that Alice has chosen $\tilde{e}$ and $\tilde{z}$ correctly.

The membership manager computes $u := \tilde{z}^{1/\tilde{e}}$ and sends u to Alice, who checks that $\tilde{z} = u^{\tilde{e}}$ holds (which is equivalent to $z = u^e$). The membership manager stores $(u, \tilde{e}, \tilde{z})$ together with Alice's identity and her commitments to $\tilde{e}$ and $\tilde{z}$ in a group-member list. Finally, Alice stores the pair (u, e) as her membership key.

Of course, $\hat{\ell}$, ℓ_1, and ℓ_2 must be chosen such that $\tilde{e}$ cannot be factored (cf. Section 5.6) and that Assumption 2 holds. In particular $\ell_2 \gg \ell_1 - (\hat{\ell} + \ell_1)/4$ must hold (cf. [18]).

5.3 The Generation of a Group-Signature

Let us first define a group-signature and then consider how a group member can compute such a signature.

Definition 5. *Let ϵ, ℓ_1, and ℓ_2 be security parameters such that $\epsilon > 1$, $\ell_2 < \ell_1 < \ell_g$, and $\ell_2 < \frac{\ell_g - 2}{\epsilon} - k$ holds. A group-signature $\mathtt{sign}(x_G, (g, h, y, z), m)$ of a message $m \in \{0,1\}^*$ is a tuple $(c, s_1, s_2, s_3, a, b, d) \in \{0,1\}^k \times \{-2^{\ell_2 + k}, \ldots, 2^{\epsilon(\ell_2+k)}\} \times \{-2^{\ell_g + \ell_1 + k}, \ldots, 2^{\epsilon(\ell_g + \ell_1 + k)}\} \times \{-2^{\ell_g + k}, \ldots, 2^{\epsilon(\ell_g + k)}\} \times G^3$ satisfying*

$$c = \mathcal{H}(g \| h \| y \| z \| a \| b \| d \| z^c b^{s_1 - c2^{\ell_1}} / y^{s_2} \| a^{s_1 - c2^{\ell_1}} / g^{s_2} \| a^c g^{s_3} \| d^c g^{s_1 - c2^{\ell_1}} h^{s_3} \| m).$$

Remark 1. *Such a group-signature would be denoted*

$$SPK\{(\eta, \vartheta, \xi) : z = b^\eta / y^\vartheta \ \wedge\ 1 = a^\eta / g^\vartheta \ \wedge\ a = g^\xi \ \wedge\ d = g^\eta h^\xi \ \wedge \\ (2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1} < \eta < 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1})\}(m).$$

To sign a message $m \in \{0,1\}^*$ on the group's behalf, a group member Alice

- chooses an integer $w \in_R \{0,1\}^{\ell_g}$, computes $a := g^w$, $b := uy^w$, and $d := g^e h^w$,
- chooses $r_1 \in_R \{0,1\}^{\epsilon(\ell_2+k)}$, $r_2 \in_R \{0,1\}^{\epsilon(\ell_g+\ell_1+k)}$, and $r_3 \in_R \{0,1\}^{\epsilon(\ell_g+k)}$, and computes
- $t_1 := b^{r_1}(1/y)^{r_2}$, $t_2 := a^{r_1}(1/g)^{r_2}$, $t_3 := g^{r_3}$, $t_4 := g^{r_1}h^{r_3}$,
- $c := \mathcal{H}(g \| h \| y \| z \| a \| b \| d \| t_1 \| t_2 \| t_3 \| t_4 \| m)$,
- $s_1 := r_1 - c(e - 2^{\ell_1})$ (in $\mathbb{Z}$), $s_2 := r_2 - cew$ (in $\mathbb{Z}$), and $s_3 := r_3 - cw$ (in $\mathbb{Z}$).

The resulting signature of m is $(c, s_1, s_2, s_3, a, b, d)$. It can easily be verified that it satisfies the verification condition given in Definition 5.

5.4 Verifying Signatures, Tracing, and Verifying Tracing

A signature $(c, s_1, s_2, s_3, a, b, d)$ of a message m can be verified by checking the equation stated in Definition 5.

To reveal the originator of a given signature $\sigma := (c, s_1, s_2, s_3, a, b, d)$ of a message m, the revocation manager first checks its correctness. He aborts if the signature is not correct. Otherwise he computes $u' := b/a^x$, issues a signature

$$P := SPK\{(\alpha) : y = g^{\alpha} \ \wedge \ b/u' = a^{\alpha}\}(u'\|\sigma\|m)$$

(see Section 4.2), and reveals *arg* $:= u'\|P$. He then looks up u' in the group-member list and will find the corresponding u, the group member's identity and his/her commitment to $\tilde{e}$ and $\tilde{z}$.

Checking whether the revocation manager correctly revealed the originator of a signature $\sigma = (c, s_1, s_2, s_3, a, b, d)$ of a message m can simply be done by verifying σ and *arg*.

5.5 Security Analysis

Before discussing the security requirements described in Section 2.1 let us have a closer look at the interactive protocol corresponding to the generation of a group-signature.

Theorem 3. *The interactive protocol corresponding to the generation of a group signature is a honest-verifier statistical zero-knowledge proof of knowledge of a membership key and certificate provided that Assumption 1 holds. Furthermore, the pair (a, b) encrypts the certificate under the revocation manager's public key y.*

Sketch. Let $(t_1, t_2, t_3, t_4, c, s_1, s_2, s_3)$ and $(t_1, t_2, t_3, t_4, \tilde{c}, \tilde{s}_1, \tilde{s}_2, \tilde{s}_3)$ be two accepting tuples that the knowledge extractor obtained. Thus we get the four equations

$$\begin{aligned} z^{\tilde{c}-c} &= b^{s_1-\tilde{s}_1+(\tilde{c}-c)2^{\ell_1}}(1/y)^{s_2-\tilde{s}_2} &(1)\\ 1 &= a^{s_1-\tilde{s_1}+(\tilde{c}-c)2^{\ell_1}}(1/g)^{s_2-\tilde{s}_2} &(2)\\ a^{\tilde{c}-c} &= g^{s_3-\tilde{s}_3} &(3)\\ d^{\tilde{c}-c} &= g^{s_1-\tilde{s}_1+(\tilde{c}-c)2^{\ell_1}}h^{s_3-\tilde{s}_3} &(4) \end{aligned}$$

Under Assumption 1 we can compute $x_3 := (s_3 - \tilde{s}_3)/(\tilde{c} - c)$ (in $\mathbb{Z}$) such that $a = g^{x_3}$ holds (cf. Lemma 1). Using that we can rewrite Equation 4 as

$$(dh^{-x_3})^{\tilde{c}-c} = g^{s_1-\tilde{s}_1+(\tilde{c}-c)2^{\ell_1}}$$

and compute (since under Assumption 1 $\tilde{c} - c$ divides $s_1 - \tilde{s}_1$) the integer

$$x_1 = \frac{s_1 - \tilde{s}_1}{\tilde{c} - c} + 2^{\ell_1}$$

such that $d = g^{x_1}h^{x_3}$ and $x_1 \in \{2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1}\}$ holds (cf. Lemma 2). Similarly, from Equation 1 we can compute (under Assumption 1) the integer

$$x_2 = \frac{s_2 - \tilde{s}_2}{\tilde{c} - c}$$

such that $z = \frac{b^{x_1}}{y^{x_2}}$ and $a^{x_1} = g^{x_2}$ holds. Since $a = g^{x_3}$ we must have $g^{x_1 x_3} = g^{x_2}$ and hence $y^{x_1 x_3} = y^{x_2}$ and

$$z = \frac{b^{x_1}}{y^{x_2}} = \frac{b^{x_1}}{(y^{x_3})^{x_1}} = \left(\frac{b}{y^{x_3}}\right)^{x_1} .$$

Thus we can conclude that $(x_1, \frac{b}{y^{x_3}})$ is a valid membership key-pair. Furthermore, (a, d) is an unconditional binding commitment to x_1 whereas (a, b) is an unconditional binding commitment to $\frac{b}{y^{x_3}}$. Since the $\log_g h$ is supposed to be unknown, the value x_1 is computationally hidden. However, the revocation manager knows the integer $\log_g y$ and is therefore able to compute the second element of that pair as

$$\frac{b}{a^{\log_g y}} = \frac{b}{y^{x_3}} .$$

□

Let us now informally discuss the security properties of the proposed group signature scheme.

Unforgeability of signatures: Due to Theorem 3 the tuple (a, b, d) is an unconditionally binding commitment to a valid membership key-pair (e, u). Under Assumption 2 it is infeasible to compute such a pair without knowing the group's order (even if other pairs are already known; cf. Section 3). Therefore the membership key-pair must stem from an execution of the registration protocol with the membership manager and only group members can sign. Furthermore, Theorem 3 shows that the revocation manager will be able to reveal the membership key of the signer by decrypting (a, b) which is sufficient for the membership manager to identify the originator of a signature.

Anonymity of signatures: Assuming that the function $\mathcal{H}$ is a random function, the values c, s_1, s_2, and s_3 do statistically not reveal any knowledge. Hence, deciding whether a signature $(c, s_1, s_2, s_3, a, b, d)$ originates from a group member with public key u' requires to decide whether $\log_g a = \log_y \frac{b}{u'}$. If one was able to decide this efficiently, this would violate Assumption 3.

Unlinkability of signatures: Linking two signatures, i.e., deciding whether two signatures $(c, s_1, s_2, s_3, a, b, d)$ and $(c', s'_1, s'_2, s'_3, a', b', d')$ originate from the same group member requires to decide whether $\log_g \frac{a}{a'} = \log_y \frac{b}{b'} = \log_h \frac{d}{d'}$, as c, s_1, s_2, s_3 and c', s'_1, s'_2, s'_3 do not reveal useful knowledge. Under Assumption 3 this is infeasible and hence signatures are unlinkable.

No framing: Given Theorem 3, signing in the name of a group member with certificate u and requires the knowledge of $\log_u z$. This can only be obtained by either factor the value $\tilde{e}$ that the membership manager received from the group member during registration or by computing the discrete logarithm of z to the base u. Both is assumed to be infeasible.

Unforgeability of tracing verification: The revocation manager has to issue an SPK denoted P as evidence that he decrypted the pair (a, b) correctly. Since (a, b) is a unconditionally binding commitment, the revocation manager has no means to prove that a membership key different from the one of the originator is encrypted in (a, b).

5.6 Efficiency Analysis

With $\epsilon = 9/8, \ell_g = \hat{\ell} = 1200, \ell_1 = 860, \ell_2 = 600$, and $k = 160$, the signature generation and verification need little less than 13′000 modular multiplications modulo a 1200-bit modulus in average, and the signature is about 1 KBytes long. Compared to the most efficient scheme given in [9], our scheme is about three times more efficient and signatures are about three times shorter when choosing the same modulus for both schemes. However, the registration protocol is less efficient in our scheme. Signatures could made shorter without compromising the security of the scheme if the parameter w in the signing procedure is chosen from a smaller domain, e.g., $\{0,1\}^{\ell_2}$ instead of $\{0,1\}^{\ell_g}$.

6 Conclusion

It is worthwhile noting that it is possible to realize blind group signatures using the techniques given in [6, 31], which are much more efficient than the blind versions of [9, 10] given in [29]. Splitting the membership and/or the revocation manager can be done by applying the techniques of [3, 12], respectively (see also [10]). As the signature generation algorithm was derived from an interactive protocol, a group identification scheme (also called identity escrow [25]) is obtained by using this protocol for identification.

Acknowledgments

The second author was supported by the Swiss National Foundation (SNF/SPP project no. 5003-045293). We gratefully acknowledge the discussions with Ivan Damgård, Markus Stadler, and Bartosz Przydatek.

References

[1] N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In W. Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–494. Springer Verlag, 1997.

[2] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communication Security*, pages 62–73. Association for Computing Machinery, 1993.

[3] D. Boneh and M. Franklin. Efficient generation of shared RSA keys. In B. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, volume 1296 of *Lecture Notes in Computer Science*, pages 425–439. Springer Verlag, 1997.

[4] J. Boyar, K. Friedl, and C. Lund. Practical zero-knowledge proofs: Giving hints and using deficiencies. *Journal of Cryptology*, 4(3):185–206, 1991.

[5] J. Camenisch. Efficient and generalized group signatures. In W. Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 465–479. Springer Verlag, 1997.

[6] J. Camenisch, U. Maurer, and M. Stadler. Digital payment systems with passive anonymity-revoking trustees. In E. Bertino, H. Kurth, G. Martella, and E. Montolivo, editors, *Computer Security — ESORICS 96*, volume 1146 of *Lecture Notes in Computer Science*, pages 33–43. Springer Verlag, 1996.

[7] J. Camenisch and M. Michels. Proving in zero-knowledge that a number n is the product of two safe primes. Manuscript, submitted for publication.

[8] J. Camenisch and M. 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.

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

[10] J. L. Camenisch. *Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem*. PhD thesis, ETH Zürich, 1998. Diss. ETH No. 12520, Hartung Gorre Verlag, Konstanz.

[11] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *Proc. 30th Annual ACM Symposium on Theory of Computing (STOC)*, 1998.

[12] D. Catalano and R. Gennaro. New efficient and secure protocols for verifiable signature sharing and other applications. In H. Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, volume 1642 of *Lecture Notes in Computer Science*, pages 105–120, Berlin, 1998. Springer Verlag.

[13] A. Chan, Y. Frankel, and Y. Tsiounis. Easy come – easy go divisible cash. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 561–575. Springer Verlag, 1998.

[14] D. Chaum, J.-H. Evertse, and J. van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In D. Chaum and W. L. Price, editors, *Advances in Cryptology — EUROCRYPT '87*, volume 304 of *Lecture Notes in Computer Science*, pages 127–141. Springer-Verlag, 1988.

[15] D. Chaum and T. P. Pedersen. Transferred cash grows in size. In R. A. Rueppel, editor, *Advances in Cryptology — EUROCRYPT '92*, volume 658 of *Lecture Notes in Computer Science*, pages 390–407. Springer-Verlag, 1993.

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

[17] L. Chen and T. P. Pedersen. New group signature schemes. In A. De Santis, editor, *Advances in Cryptology — EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 171–181. Springer-Verlag, 1995.

[18] D. Coppersmith. Finding a small root of a bivariatre interger equation; factoring with high bits known. In U. Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 178–189. Springer Verlag, 1996.

[19] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y. G. Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer Verlag, 1994.

[20] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1:77–94, 1988.

[21] A. Fiat and A. Shamir. How to prove yourself: Practical solution to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology — CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer Verlag, 1987.

[22] E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In B. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer Verlag, 1997.

[23] M. Girault. An identity-based identification scheme based on discrete logarihtms modulo a composite number. In I. B. Damgård, editor, *Advances in Cryptology – EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 481–486. Springer-Verlag, 1991.

[24] M. Girault. Self-certified public keys. In D. W. Davies, editor, *Advances in Cryptology — EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer-Verlag, 1992.

[25] J. Kilian and E. Petrank. Identity escrow. In H. Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, volume 1642 of *Lecture Notes in Computer Science*, pages 169–185, Berlin, 1998. Springer Verlag.

[26] S. J. Kim, S. J. Park, and D. H. Won. Convertible group signatures. In K. Kim and T. Matsumoto, editors, *Advances in Cryptology — ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 311–321. Springer Verlag, 1996.

[27] K. Koyama, U. Maurer, T. Okamoto, and S. Vanstone. New public-key schemes based on elliptic curves over the ring Z_n. In J. Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 252–266. Springer-Verlag, 1992.

[28] C. H. Lim and P. J. Lee. On the security of convertible group signatures. *Electronics Letters*, 1996.

[29] A. Lysyanskaya and Z. Ramzan. Group blind digital signatures: A scalable solution to electronic cash. In *Proc. Second International Conference on Financial Cryptography*, 1998.

[30] M. Michels. Comments on some group signature schemes. Technical Report TR-96-3-D, Departement of Computer Science, University of Technology, Chemnitz-Zwickau, Nov. 1996.

[31] T. Okamoto. Provable secure and practical identification schemes and corresponding signature schemes. In E. F. Brickell, editor, *Advances in Cryptology — CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53. Springer-Verlag, 1993.

[32] S. J. Park, I. S. Lee, and D. H. Won. A practical group signature. In *Proceedings of the 1995 Japan-Korea Workshop on Information Security and Cryptography*, pages 127–133, Jan. 1995.

[33] H. Petersen. How to convert any digital signature scheme into a group signature scheme. In M. Lomas and S. Vaudenay, editors, *Security Protocols Workshop*, Paris, 1997.

[34] D. Pointcheval and J. Stern. Security proofs for signature schemes. In U. Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398. Springer Verlag, 1996.

[35] G. Poupard and J. Stern. Security analysis of a practical "on the fly" authentication and signature generation. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 422–436. Springer Verlag, 1998.

[36] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, Feb. 1978.

[37] C. P. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):239–252, 1991.

[38] A. Shamir. On the generation of cryptographically strong pseudorandom sequences. In *ACM Transaction on Computer Systems*, volume 1, pages 38–44, 1983.

[39] J. van de Graaf and R. Peralta. A simple and secure way to show the validity of your public key. In C. Pomerance, editor, *Advances in Cryptology — CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 128–134. Springer-Verlag, 1988.

Recent BRICS Report Series Publications

RS-98-27 **Jan Camenisch and Markus Michels.** ***A Group Signature Scheme Based on an RSA-Variant*****. November 1998. 18 pp. Preliminary version appeared in Ohta and Pei, editors,** ***Advances in Cryptology: 4th ASIACRYPT Conference on the Theory and Applications of Cryptologic Techniques*****, ASIACRYPT '98 Proceedings, LNCS 1514, 1998, pages 160–174.**

RS-98-26 **Paola Quaglia and David Walker.** ***On Encoding $p\pi$ in $m\pi$*****. October 1998. 27 pp. Full version of paper to appear in** ***Foundations of Software Technology and Theoretical Computer Science: 18th Conference*****, FCT&TCS '98 Proceedings, LNCS, 1998.**

RS-98-25 **Devdatt P. Dubhashi.** ***Talagrand's Inequality in Hereditary Settings*****. October 1998. 22 pp.**

RS-98-24 **Devdatt P. Dubhashi.** ***Talagrand's Inequality and Locality in Distributed Computing*****. October 1998. 14 pp.**

RS-98-23 **Devdatt P. Dubhashi.** ***Martingales and Locality in Distributed Computing*****. October 1998. 19 pp.**

RS-98-22 **Gian Luca Cattani, John Power, and Glynn Winskel.** ***A Categorical Axiomatics for Bisimulation*****. September 1998. ii+21 pp. Appears in Sangiorgi and de Simone, editors,** ***Concurrency Theory: 9th International Conference*****, CONCUR '98 Proceedings, LNCS 1466, 1998, pages 581–596.**

RS-98-21 **John Power, Gian Luca Cattani, and Glynn Winskel.** ***A Representation Result for Free Cocompletions*****. September 1998. 16 pp.**

RS-98-20 **Søren Riis and Meera Sitharam.** ***Uniformly Generated Submodules of Permutation Modules*****. September 1998. 35 pp.**

RS-98-19 **Søren Riis and Meera Sitharam.** ***Generating Hard Tautologies Using Predicate Logic and the Symmetric Group*****. September 1998. 13 pp.**

RS-98-18 **Ulrich Kohlenbach.** ***Things that can and things that can't be done in PRA*****. September 1998. 24 pp.**