A Group Signature Scheme with Improved Efficiency
(Extended Abstract)

Jan Camenisch[*]
BRICS[**]
Department of Computer Science
University of Aarhus
Ny Munkegade
DK – 8000 Århus C, Denmark
camenisch@daimi.aau.dk

Markus Michels [***]
r3 security engineering ag /
Entrust Technology
P. O. Box
CH – 8301 Glattzentrum/Zurich,
Switzerland
Markus.Michels@entrust.com

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 solution based on a variation of the RSA problem is more efficient than previous ones satisfying these requirements.

Keywords. Group signature scheme for large groups, digital signature schemes, 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 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

[*] Part of this work was done while this author was with ETH Zurich.

[**] Basic Research in Computer Science, Center of the Danish National Research Foundation.

[***] 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.

K. Ohta and D. Pei (Eds.): ASIACRYPT'98, LNCS 1514, pp. 160–174, 2000.

a single company's public key. A further application of group signature schemes is electronic cash as was pointed out in [32]. 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 [28].

Various group signature schemes have been proposed so far. However, in the schemes presented in [7,16,17,36] 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 families of efficient schemes presented in [9,10] (and the blind versions thereof [32]) 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 [28] 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 so-called *strong RSA-assumption* proposed in [1,25]. Compared to the solutions in [9,10], our scheme is based on a different number-theoretic assumption and is also more efficient.

2 Model and an Approach for Realization

2.1 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.

`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.

[2] The other schemes [29,35] with the same properties were shown to be flawed [31,33].

The following informally stated security requirements must hold:

Unforgeability of signatures: Only group members are able to sign messages.

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: The revocation manager can not accuse a signer falsely of having originated a given signature, e.g., by issuing an argument *arg* such that `vertracing` outputs 1.

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 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 a membership manager's public key of an ordinary digital signature scheme and a 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 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, finding integers u, x such that $u^e \equiv g^x + 1 \pmod{n}$ holds is hard, where g is an element of large order. The second one is that it is hard to find u and x with $|x| < |n|/2$ such that $u^3 \equiv x^5 + v \pmod{n}$ given v and n, where v is a suitably chosen integer and n is an RSA-modulus.

In the next section we will introduce an alternative assumption that allows the construction of a new group signature scheme.

[3] These are message dependent non-interactive arguments derived from 3-move honest-verifier zero-knowledge proofs of knowledge using the Fiat-Shamir heuristic [23,24].

3 Number Theoretic Assumptions

Recently, Barić and Pfitzmann [1] as well as Fujisaki and Okamoto [25] independently proposed a variation of the well-known RSA [39] assumption, the so-called *strong RSA assumption*. We will modify this assumption slightly. Let k, ℓ_g, ℓ_1, $\ell_2 < \ell_g$, and $\epsilon > 1$ be security parameters and, for simplicity, let denote $\tilde{\ell} := \epsilon(\ell_2 + k) + 1$. Furthermore, let $\mathcal{G}(\ell_g)$ denote the set of groups whose order has length ℓ_g and has two prime factors of length $(\ell_g - 2)/2$. Finally, let be $\mathcal{M}(G, z) = \{(u, e) \mid z = u^e, u \in G, e \in \{2^{\ell_1}, \ldots, 2^{\ell_1} + 2^{\ell_2}\}, e \in \mathsf{primes}\}$, where $G \in \mathcal{G}(\ell_g)$ and $z \in G$.

Assumption 1 (Modified strong RSA assumption). *For all probabilistic polynomial-time algorithms $\tilde{A}$, all polynomials $p(\cdot)$, all sufficiently large ℓ_g, and suitably chosen ℓ_1, ℓ_2, k, and ϵ*

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

Possible choices for G are discussed in Section 5. Let us remark that, given u, e, $\tilde{u}$, and $\tilde{e}$ with $z = u^e = \tilde{u}^{\tilde{e}}$, 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. According to a result in [22,41], and as all e's in M are prime, it is infeasible to compute (u, e') satisfying $u^{e'} = z$ for an e' that does not divide the product of all e's in M as long as the standard RSA assumption holds. Hence there is no further attack except the one mentioned above.

Our group signature scheme further relies on the so-called Decision Diffie-Hellman (DDH) assumption. Let $G \in \mathcal{G}(\ell_g)$, n' be the divisor of G's order of length $\ell_g - 2$, and define the two sets

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

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

Assumption 2 (Decision Diffie-Hellman assumption). *For all probabilistic polynomial-time algorithms $A : G^4 \to \{0, 1\}$, the two probability distributions*

$$Pr[a = 1 : T \in_R \mathcal{DH}, a := A(T)] \quad \textit{and} \quad Pr[a = 1 : T \in_R \mathcal{Q}, a := A(T)]$$

are computationally indistinguishable.

We remark that in the case $G = \mathbb{Z}_n^*$, where n is an RSA-modulus, the DDH assumption does not hold. The Jacobi-symbol, which can be computed efficiently without knowing the factorization 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 $\log_{g_1} y_1 \neq \log_{g_2} y_2$. If $G = \langle g \rangle$ is defined a subgroup of $\mathbb{Z}_n^*$ such that $(g|n) = 1$ this problem is overcome.

4 Building Blocks

In this section we introduce the building blocks for our scheme borrowing notation from [9]. These building blocks are signature schemes derived from statistical (honest-verifier) zero-knowledge proofs of knowledge using the Fiat-Shamir heuristic [23,24] and are therefore called "Signature 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 formalized as the random oracle model (e.g., see [2,37])[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 [40] that is derived from an honest-verifier 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 logarithm, and the fourth the knowledge of a discrete logarithm that lies in a certain interval. Of course, these building blocks can be combined in the usual 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

$$SKP\big\{(\alpha,\beta) :\ y = g^{\alpha} \ \wedge\ z = g^{\beta} h^{\alpha}\big\}\,(m),$$

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 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] 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 that yield an insecure signature scheme. However, it is believed that the approach is still valid for the kind of protocols considered here.

4.1 Showing the Knowledge of a Discrete Logarithm

This protocol is an adaption of the protocols for proving the knowledge of a discrete logarithm [14,40] to the setting with a group of unknown order due to Girault [26,27]. A consequence of this setting is that the usual knowledge extractor for showing that a protocol is a proof of knowledge does not work; since 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 [38] 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.

In the following we propose an alternative security proof using the model of Fujisaki and Okamoto [25]. In this model, the key setup is made a part of the protocol, i.e., the verifier chooses the group G and all other parameters and sends these as a first step to the prover. As a consequence, the knowledge extractor is allowed to choose the group and hence knows the group order. When turning this protocol into a signature scheme, the first steps, i.e., the key setup, are carried out interactively, and only the last three half-rounds are made non-interactive using the Fiat-Shamir heuristic.

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 $x = \log_g y$ of its public key y can compute such a signature $(s, c) = SPK\{(\alpha) : y = g^\alpha\}(m)$ 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 $\mathbb{Z}$).

Showing that the interactive protocol corresponding to this signature scheme and the key setup is a proof of knowledge of the integer $x := \log_g y$ is straight forward. The proof that it is honest-verifier statistical zero-knowledge for any $\epsilon > 1$ is immediate from the proofs found in [38,42] for similar protocols. In [10] it is analyzed how much information (t, c, s) gives about x depending on the choice of ϵ.

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 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 $\mathit{SPK}\{(\alpha) : y_1 = g^\alpha \wedge y_2 = h^\alpha\}(m)$ 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 proofs of this building block follow from the ones of the previous building block.

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 [20] to the setting with unknown order.

Definition 3. *Let $\epsilon > 1$ be a security parameter. A tuple $(c_1, c_2, s_1, s_2) \in \{0,1\}^k \times \{0,1\}^k \times \{-2^{\ell_g+k}, \ldots, 2^{\epsilon(\ell_g+k)}\} \times \{-2^{\ell_g+k}, \ldots, 2^{\epsilon(\ell_g+k)}\}$ 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*

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

Assume that the signer knows $x \in_R \{0,1\}^{\ell_g}$ such that $y_1 = g^x$ holds. Then a signature $\mathit{SPK}\{(\alpha, \beta) : y_1 = g^\alpha \vee y_2 = h^\beta\}(m)$ of a message $m \in \{0,1\}^*$ can be computed as follows.

- Choose $r_1 \in_R \{0,1\}^{\epsilon(\ell_g+k)}$, $r_2 \in_R \{0,1\}^{\epsilon(\ell_g+k)}$, $c_2 \in_R \{0,1\}^k$ and compute $t_1 := g^{r_1}$, $t_2 := h^{r_2}y_2^{c_2}$,
- $c_1 := c_2 \oplus \mathcal{H}(g\|h\|y_1\|y_2\|t_1\|t_2\|m)$,
- $s_1 := r_1 - c_1x$ (in $\mathbb{Z}$), and $s_2 := r_2$.

The security proofs of this building block follow from the ones of the previous building blocks and from [20].

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 a protocol presented by Chan et al. [13].

Definition 4. *Let $\epsilon > 1$ be a security parameter and let $\ell_1 < \ell_g$ and ℓ_2 denote lengths. A pair $(c, s) \in \{0,1\}^k \times \{-2^{\ell_2+k}, \ldots, 2^{\epsilon(\ell_2+k)}\}$ 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*

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

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 $x \in \{2^{\ell_1}, \ldots, 2^{\ell_1} + 2^{\ell_2} - 1\}$ 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 $\mathbb{Z}$).

Theorem 1. *The interactive protocol corresponding to the signature scheme of Definition 4 and the key setup is a statistical honest-verifier zero-knowledge proof of knowledge of an $x \in \{2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1}\}$ such that $y = g^x$ holds.*

Proof (Sketch). The proof that the protocol is statistical honest-verifier zero-knowledge is as before.

Let us consider the proof-of-knowledge part. Extracting the x such that $g^x = y$ is as usual. It remains to show that the extracted x lies indeed in the required interval. Let (t, c_i, s_i) be the accepting triples that the knowledge extractor got and used to compute x. Then we have $y^{c_1} g^{s_1 - c_1 2^{\ell_1}} = y^{c_2} g^{s_2 - c_2 2^{\ell_1}}$, where $c_1 \neq c_2$. Without loss of generality, we can assume that $c_2 > c_1$. Let denote $\Delta s := s_1 - s_2$ and $\Delta c := c_2 - c_1$. Then $(x - 2^{\ell_1})\Delta c \equiv \Delta s \pmod{\operatorname{ord}(g)}$ holds. As $\Delta c \in \{1, \ldots, 2^k - 1\}$ and $\Delta s \in \{-2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\epsilon(\ell_2+k)+1}\}$, we have $(x - 2^{\ell_1})\Delta c \in \{-2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\epsilon(\ell_2+k)+1}\} + j \cdot \operatorname{ord}(g)$ and thus also $(x - 2^{\ell_1}) \in \{-2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\epsilon(\ell_2+k)+1}\} + j \cdot \operatorname{ord}(g)$ for some integer j. From this it follows that $x \pmod{\operatorname{ord}(g)} \in \{2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1}\}$. Since it is assumed to be infeasible for the prover to compute the order of g, the integer x must in fact lie in $\{2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1}\}$ (cf. [25]). □

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

5 Proposed Scheme

In this section we propose a realization of a group signature scheme the security of which is based on Assumptions 1 and 2. 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 both assumptions hold. Furthermore, he chooses a second generators 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 is the only one who should know 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 certain 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 value. 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

- u is encrypted in (a, b) of under the revocation manager's public key (which is part of the group public key)
- d commits to e,
- e lies in a given 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 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 1 and 2 hold. He publishes z, g, h, G, ℓ_g, and a proof that z, g, and h have the same, large order of the order of magnitude 2^{ℓ_g}. Also, he proves that the order of g, h, and z is not prime and not smooth. The latter would enable the membership manager to compute discrete logarithms in G. The membership manager must also proof 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 is of the right form, there is no efficient proof system to the best of our knowledge. Thus one has to use general zero-knowledge proof techniques (e.g., [5,6,19]) and a circuit that takes as input integers p, q, p', and q' and outputs 1 if and only if the inputs are primes and if $n = pq$, $p = 2p' + 1$, and $p = 2p' + 1$ holds. The size of p and q can be checked by the number of input bits for them (they should have at most $\lceil 0.5 \log n \rceil$ bits). This is not very efficient but must be done only once. To verify that an element a has the (large) order $p'q'$ in $\mathbb{Z}_n^*$ and Jacobi symbol 1, one needs only to test whether $a \not\equiv 1 \pmod n$ and $\gcd(a - 1, n) = 1$ holds and provide a proof should that a is a quadratic residue modulo n. An alternative choice of G is a suitable elliptic curve (e.g., see [30]).

5.2 Registration

To become a group member Alice chooses a random prime $\hat{e} \in_R \{2^{\hat{\ell}-1}, \ldots, 2^{\hat{\ell}}-1\}$ such that $\hat{e} \not\equiv 1 \pmod 8$ and a random number $e_1 \in_R \{1, \ldots, 2^{\ell_2} - 1\}$. She computes $\tilde{z} := z^{\hat{e}} \pmod n$ and the commitment $\hat{c} = \tilde{z}^{e_1} h^{r_{e_1}}$ with $r_{e_1} \in_R \{0,1\}^{\ell_g}$. Then she sends $\tilde{z}$ and $\hat{c}$ to the membership manager. The membership manager chooses a random number $e_2 \in_R \{1, \ldots, 2^{\ell_2} - 1\}$ and sends it to Alice. Alice computes $e_3 := e_1 + e_2 \pmod{2^{\ell_2}}$ and $e := e_3 + 2^{\ell_1}$. If e is not a prime satisfying $e \not\equiv 1 \pmod 8$ and $e \not\equiv \hat{e} \pmod 8$ Alice reveals e and $\hat{e}$ to enable the membership manager checking that she hasn't cheated and they repeat the whole process. The success probability per round is roughly $1/(\ell_1 2 \ln 2)$.

If e is a prime, Alice computes $\tilde{e} := e\hat{e}$, 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

$$W := SPK\Big\{(\alpha, \beta, \gamma, \delta, \zeta) : \hat{c} = \tilde{z}^{\alpha} h^{\beta} \ \wedge \ (-2^{\epsilon(\ell_2+k)+1} < \alpha < 2^{\epsilon(\ell_2+k)+1}) \ \wedge \\ \tilde{z} = z^{\gamma} \ \wedge \ \Big((\hat{c}\tilde{z}^{e_2 - 2^{\ell_2} + 2^{\ell_1}})/z^{\tilde{e}} = h^{\delta} \ \vee \ (\hat{c}\tilde{z}^{e_2 + 2^{\ell_1}})/z^{\tilde{e}} = h^{\zeta}\Big)\Big\}(\tilde{z}) \ ,$$

with the membership manager (cf. previous section). Furthermore, Alice proves that $\tilde{e}$ is the product of two primes (e.g., using the methods described in [4,43]). 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 formed $\tilde{e}$ and $\tilde{z}$ correctly and that $\tilde{e}/\log_z \tilde{z} - 2^{\ell_1}$ equals the sum of e_2 and the e_1 committed to in $\hat{c}$ modulo 2^{ℓ_2}.

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 commitment 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). In particular $\ell_2 \gg \ell_1 - (\hat{\ell} + \ell_1)/4$ must hold [18].

5.3 Signature Generation

Let us first define a group signature and then show 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. 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 $w \in_R \{0,1\}^{\ell_g}$, 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 $P := SPK\{(\alpha) : y = g^\alpha \wedge b/u' = a^\alpha\}(\sigma\|m)$ (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 and the parameter setup.

Theorem 2. *The interactive protocol sequentially composed of the parameter setup and the protocol corresponding to the generation of a group signature is a zero-knowledge proof of knowledge of a membership key and certificate. Furthermore, the pair (a, b) encrypts the certificate under the revocation manager's public key y.*

Proof (Sketch). Using the standard techniques (cf. Section 4), this protocol can be shown to be a statistical zero-knowledge proof of knowledge of values x_1, x_2, and x_3 such that

$$x_1 \in \{2^{\ell_1} - 2^{\epsilon(\ell_2+k)+1}, \ldots, 2^{\ell_1} + 2^{\epsilon(\ell_2+k)+1}\}$$

$$z = \tfrac{b^{x_1}}{y^{x_2}} \ , \quad a^{x_1} = g^{x_2} \ , \quad a = g^{x_3} \ , \qquad \text{and} \ \ d = g^{x_1}h^{x_3}$$

holds. From the second and third equations we can conclude that $g^{x_2} = g^{x_3 x_1}$ and thus also $y^{x_2} = y^{x_3 x_1}$ holds. Therefore, we have

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

and hence $(x_1, \frac{b}{y^{x_3}})$ is a valid membership key-pair. The triple (a, b, d) is an unconditionally binding commitment to these two values and hence the group member/prover must have known[5] them when she computed a, b, and d. Since it is assumed that the group member cannot compute roots nor discrete logarithms (as otherwise Assumption 1 would not hold), she must have had other means to get such a pair, i.e., by having run the registration protocol with the membership manager.

Finally, the commitments can be opened by the entities knowing $\log_h y$ and $\log_g h$, respectively, i.e., the values are encrypted for these entities. We recall, that the first discrete log was chosen be the revocation manager, while the second is assumed to be unknown. □

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

Unforgeability of Signatures: This is due to Theorem 2.

Anonymity of Signatures: It can be shown that the values c, s_1, s_2, and s_3 do not reveal any useful 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 2.

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 2 this is infeasible and hence signatures are unlinkable.

No Framing: Given Theorem 2, signing in the name of a group member with certificate u and requires the computation of $\log_u z$ or to factor the value $\tilde{e}$ that the membership manager received from the group member during registration. Both is assumed to be infeasible.

Unforgeability of Tracing: The pair (a, b) that is part of a signature is an ElGamal encryption [21] of the signer's membership key under the revocation manager's public key y. Theorem 2 shows that $b/(y^{\log_g a}) = b/a^x$ is a valid membership public key. Due to Assumption 1 this must be the membership certificate of the group member who signed. Therefore, by decrypting (a, b) the revocation manager can reveal the originator of a signature at hand. In the tracing algorithm the revocation manager issues an SPK denoted *arg* which shows that he decrypted the membership public key correctly. Forging this SPK is infeasible under Assumption 1.

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

[5] This is important, since the knowledge-extractor knows the order, he can always find a random e and u such that $z = u^e$.

times more efficient and signatures are about three times shorter when choosing the same modulus for both schemes. 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 [8,34], which are much more efficient than the blind versions of [9,10] given in [32]. 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 [28]) 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 *LNCS*, 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 *LNCS*, 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. Boyar and R. Peralta. Short discreet proofs. In U. Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, volume 1070 of *LNCS*, pages 131–142. Springer Verlag, 1996.
6. G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, Oct. 1988.
7. J. Camenisch. Efficient and generalized group signatures. In W. Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *LNCS*, pages 465–479. Springer Verlag, 1997.
8. J. Camenisch, U. Maurer, and M. Stadler. Digital payment systems with passive anonymity-revoking trustees. In *Computer Security — ESORICS 96*, volume 1146 of *LNCS*, pages 33–43. Springer Verlag, 1996.
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 *LNCS*, 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, ISBN 3-89649-286-1, 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 *Advances in Cryptology — CRYPTO '98*, *LNCS*. Springer Verlag, 1998.
13. A. Chan, Y. Frankel, and Y. Tsiounis. Easy come – easy go divisible cash. In *Advances in Cryptology — EUROCRYPT '98*, volume 1403 of *LNCS*.
14. D. Chaum, J.-H. Evertse, and J. van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In *Advances in Cryptology — EUROCRYPT '87*, pages 127–141.
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 *LNCS*, 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 *LNCS*, pages 257–265.
17. L. Chen and T. P. Pedersen. New group signature schemes. In *Advances in Cryptology — EUROCRYPT '94*, volume 950 of *LNCS*, pages 171–181.
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 *LNCS*, pages 178–189. Springer Verlag, 1996.
19. R. Cramer and I. Damgård. Linear zero-knowledge: A note on efficient zero-knowledge proofs and arguments. In *Proc. 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 436–445. ACM press, 1997.
20. 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 *LNCS*, pages 174–187.
21. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology — CRYPTO '84*, volume 196 of *LNCS*, pages 10–18. Springer Verlag, 1985.
22. J.-H. Evertse and E. van Heyst. Which new RSA signatures can be computed from certain given RSA signatures? *Journal of Cryptology*, 5:41–52, 1992.
23. U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1:77–94, 1988.
24. 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 *LNCS*, pages 186–194. Springer Verlag, 1987.
25. E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In B. S. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, volume 1294 of *LNCS*, pages 16–30. Springer Verlag, 1997.
26. 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 *LNCS*, pages 481–486. Springer-Verlag, 1991.
27. M. Girault. Self-certified public keys. In *Advances in Cryptology — EUROCRYPT '91*, volume 547 of *LNCS*, pages 490–497. Springer-Verlag, 1992.
28. J. Kilian and E. Petrank. Identity escrow. In *Advances in Cryptology — CRYPTO '98*, *LNCS*. Springer Verlag, 1998.
29. S. J. Kim, S. J. Park, and D. H. Won. Convertible group signatures. In *Advances in Cryptology — ASIACRYPT '96*, volume 1163 of *LNCS*, pages 311–321.

30. K. Koyama, U. Maurer, T. Okamoto, and S. Vanstone New Public-key Schemes Based on Elliptic Curves over the Ring Z_n. In *Advances in Cryptology — CRYPTO '91*, volume 576 of *LNCS*,pages 252–266.
31. C. H. Lim and P. J. Lee. On the security of convertible group signatures. *Electronics Letters*, 1996.
32. A. Lysyanskaya and Z. Ramzan. Group blind digital signatures: A scalable solution to electronic cash. In *Proc. Second Int. Conf. on Financial Cryptography*, 1998.
33. M. Michels. Comments on some group signature schemes. TR-96-3-D, Departement of Computer Science, University of Technology, Chemnitz-Zwickau, Nov. 1996.
34. 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 *LNCS*, pages 31–53. Springer-Verlag, 1993.
35. S. J. Park, I. S. Lee, and D. H. Won. A practical group signature. In *Proc. of the 1995 Japan-Korea Workshop on Information Security and Cryptography.*
36. H. Petersen. How to convert any digital signature scheme into a group signature scheme. In *Security Protocols Workshop*, Paris, 1997.
37. D. Pointcheval and J. Stern. Security proofs for signature schemes. In U. Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, volume 1070 of *LNCS*, pages 387–398. Springer Verlag, 1996.
38. 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 *LNCS*, pages 422–436. Springer Verlag, 1998.
39. R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Comm. of the ACM*, 21(2):120–126, Feb. 1978.
40. C. P. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):239–252, 1991.
41. A. Shamir. On the generation of cryptographically strong pseudorandom sequences. In *ACM Trans. on Computer Systems*, volume 1, pages 38–44, 1983.
42. M. Stadler. *Cryptographic Protocols for Revocable Privacy.* PhD thesis, ETH Zürich, 1996. Diss. ETH No. 11651.
43. 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 *LNCS*, pages 128–134. Springer-Verlag, 1988.