Blind Spontaneous Anonymous Group Signatures for Ad Hoc Groups

Tony K. Chan, Karyin Fung, Joseph K. Liu, and Victor K. Wei

Department of Information Engineering,
The Chinese University of Hong Kong,
Shatin, Hong Kong
{klchan3, kyfung2, ksliu9, kwwei}@ie.cuhk.edu.hk

Abstract. Spontaneous anonymous group (SAG) cryptography is a fundamental alternative to achieve thresholding without group secret or setup. It has gained wide interests in applications to ad hoc groups. We present a general construction of blind SAG 1-out-of-n and t-out-of-n signature schemes from essentially any major blind signature. In the case when our scheme is built from blind Schnorr (resp. Okamoto-Schnorr) signature, the parallel one-more unforgeability is reduced to Schnorr's ROS Problem in the random oracle model plus the generic group model. In the process of our derivations, we obtain a generalization of Schnorr's result [17] from single public key to multiple public keys.

1 Introduction

The popular goals of group cryptography or threshold cryptography are usually:

> Any t members of a group of n members can jointly demonstrate a knowledge concerning the group that no combination of $t-1$ or fewer members can demonstrate.

There are threshold signature schemes that require no less than t members to jointly generate. There are threshold decryption schemes (cryptosystems) that require no less than t members to jointly decrypt. Besides unforgeability, other properties such as robustness, adaptive adversary models, blind signatures, culpability or exculpability, witness hiding, witness indistinguishability (anonymity) are also significant research topics.

Since its inception, group cryptography and threshold cryptography [11] have traditionally been achieved through the secret sharing technique [19, 4]. Also since its inception [9], anonymous (insider-indistinguishable) group cryptography has traditionally been achieved by the technique of blind signatures or other forms of transfer proof-of-knowledge (TPoK). For further details, see [9, 7]

Recently a fundamental alternative has gained wide interests. In the *spontaneity* paradigm to group cryptography, there is no group secret. There is also no setup. Any single entity can arbitrarily and spontaneously conscript $n-1$

C. Castelluccia et al. (Eds.): ESAS 2004, LNCS 3313, pp. 82–94, 2005.

diversion members to form a group, and complete a signature without the participation, or even knowledge, of the diversion members. The resulting signature can be proven to be from one of the n group members. Yet the actual signer remains anonymous (signer-indistinguishable), with irrevocable, exculpable anonymity. The only requirement is that each group member has a published public key, for the purpose of signature verification. There are also t-out-of-n threshold versions where t entities joint to spontaneously conscript $n - t$ diversion members.

Compared with traditional threshold signature schemes, spontaneous group signatures achieved the definition goal quoted at the beginning of this section. Yet there is no group secret. There is no group setup which requires the participation of non-insider members.

Due to its flexibility and the ease (or lack) of setup, SAG cryptography has been deemed perfectly suitable for applications in ad hoc groups [16, 6, 5].

Compared with traditional privacy (anonymity) protection schemes, spontaneous group cryptography is naturally anonymous. It achieves anonymity without using blinding techniques. Furthermore, the anonymity in spontaneous anonymous group (SAG) cryptography is very strong: in its basic version, the anonymity is unconditional (information-theoretic), irrevocable, and exculpable. The last property means that even if all communication sessions and all secret keys are subpoenaed, the anonymity cannot be revealed. Variants of SAG cryptography achieved different tradeoffs in anonymity based on candidate hard problems and optional revocability and optional culpability.

Our Contributions: In this paper, we present the first blind [8] spontaneous anonymous group (SAG) signature schemes. Based on essentially any major blind signature, we construct ring-type [16, 1] 1-out-of-n blind SAG signatures and CDS-type [10] t-out-of-n blind SAG signatures. The blindness of our SAG blind signature depends on that of its underlying component blind signature. The parallel one-more unforgeability of our SAG signature, when the underlying component is the Schnorr (resp. Okamoto-Schnorr) blind signature, is reduced to Schnorr's ROS Problem [17], in the random oracle model [3] plus the generic group model [14]. In the process, we extend Schnorr's result [17] on single-key parallel one-more unforgeability (p1m-uf) to obtain a reduction of multiple-key parallel unforgeability (mk-p1m-uf) of Schnorr (resp. Okamoto-Schnorr) blind signature to the ROS Problem, in the random oracle model plus the generic group model.

Paper Organization: Background materials in Section 2. Security models and definition of security notions in Section 3. Constructions of blind SAG signatures in Section 4. Security analyses in Section 5. Conclusions in Section 6.

2 Background Materials

We review background results needed subsequently.

2.1 General Background

A **PoK (Proof-of-Knowledge)** is a three-move interactive protocol consisting of (Prover, Verifier). Common input consists of a public key, PK. Prover has the additional input SK. The three moves are $\mathcal{K}$=($\mathcal{T}$,$\mathcal{C}$,$\mathcal{S}$)=(*commit*, *challenge*, *response*). *Completness* means, with all sides honest, results are as they should be. *Soundness* means two random challenge-response pair to the same commitment result in witness extraction. *Special Soundness* means: any two challenge-response pair with the same commitment result in witness extraction.

A **blind signature** consists of the tuple (BlindSigner, Warden, Verifier) where the three components form an interactive protocol as follows:

1. Common input to all three parties: PK. Additional input to BlindSigner: SK.
2. BlindSigner sends t' (commitment) to Warden.
3. Warden sends t to Verifier.
4. Verifier sends message m to Warden.
5. Warden sends c' to BlindSigner.
6. BlindSigner sends s' to Warden.
7. Warden sends s to Verifier.
8. Verifier confirms that (t, s) is a valid signature on m w.r.t. PK.

Typically, Warden is instantiated as a tuple of mappings (f_t, f_c, f_s) and that in various moves do the following:

1. Warden randomly generates Δ_c and Δ_s, computes $t := f_t(PK, t', \Delta_c, \Delta_s)$, and sends t to Verifier.
2. Verifier sends m to Warden.
3. Warden computes $c := H(t, m)$ $c' := f_c(PK, t', \Delta_c, \Delta_s, c)$ and sends c' to BlindSigner.
4. BlindSigner computes s' and sends it to Warden.
5. Warden computes $s = f_s(PK, t', \Delta_c, \Delta_s, t, c, c', s')$ and sends it to Verifier.

If (t', c', s') is a valid PoK, then so is (t, c, s). Some examples below.
Schnorr blind signature[18]: Relation $R = \{(y = g^x, x) | x \in \{1, \cdots, q\}\}$ with $(\mathcal{T}, \mathcal{C}, \mathcal{S} : \mathcal{T} = g^{\mathcal{S}} y^{\mathcal{C}})$ and $\mathcal{T} = \mathcal{T}' g^{\Delta_S} y^{\Delta_C}$,
Okamoto-Schnorr blind signature [15]: Relation $R = \{(g^{x_1} h^{x_2}, (x_1, x_2)) | x_1, x_2 \in \{1, \cdots, q\}\}$. with $(\mathcal{T}, \mathcal{C}, \mathcal{S}) = (\ g^{r_1} h^{r_2},\ c,\ (s_1, s_2) = (x_1 + r_1 c, x_2 + r_2 c)\ )$

Blindness: The signer of a blind signature has no information about the message during and after a blind signature/TPoK protocol. Given any message-signature pair, the signer cannot find out when and for whom it was signed.

2.2 Schnorr's ROS Assumption

Schnorr [17] presented a then-new algorithm to compute the parallel one-more forgery of Schnorr (resp. Okamoto-Schnorr) blind signatures. He showed the equivalence of the parallel one-more unforgery of those two blind signatures and the ROS Problem, in the random oracle model plus the generic group model.

His techique also applied to many other blind signatures. In this paper, we will use the following form of Schnorr's ROS Problem:

The ROS Problem: Given $1 \leq q_B \leq q_H$, typically $q_B << q_H$, and all computations are in Z_q. Compute a $q_H \times q_B$ matrix $\mathbf{A}$, such that the probability of computing the following problem is non-negligible:

> Given random $\hat{c} = [\hat{c}_1, \cdots, \hat{c}_{q_H}]$, compute $J \subset \{1, \cdots, q_H\}$ with $|J| = q_B+1$, $j_0 \in J$, $\{\alpha_j : j \in J\}$ with $\alpha_{j_0} \neq 0$, and β such that $\sum_{j \in J} \alpha_j [\mathbf{A}_j, \hat{c}_j] = \beta$ and $\{\mathbf{A}_j : j \in J \setminus \{j_0\}\}$ are linearly independent.

Note $\mathbf{A}_j$ denote the j-th row vectors of $\mathbf{A}$, and $[\mathbf{A}_j, \hat{c}_j]$ denotes the lengthened vector by one more entry $\hat{c}_j$.

2.3 Background About SAG Signature

First, the definition of SAG signatures.

Definition 1. *Let $L = \{PK_1, \cdots, PK_n\}$ be a list of n public keys, θ be an integer, $1 \leq \theta \leq n$, m be a message, and $\sigma = (t_1, \cdots, t_n, c_1, \cdots, c_n, s_1, \cdots, s_n)$ be a tuple. Let H, H_1, $\cdots$, H_n be full-domain collision-free secure hashing functions. The tuple $(L, n, \theta, m, \sigma)$ is a* ring-type SAG signature [16, 1] *if the following all hold:*

1. $\theta = 1$
2. *For each i, $1 \leq i \leq n$, we have $c_i = H_i(L, n, m, t_{i-1})$ and (t_i, c_i, s_i) is a valid PoK conversation w.r.t. PK_i. (t_0 is interpreted as t_n.)*

The tuple is a CDS1-type SAG signature [10] *if the following all hold*

1. *Each tuple (t_i, c_i, s_i) is a valid PoK conversation w.r.t. PK_i, $1 \leq i \leq n$.*
2. *The polynomial f interpolated from $f(i) = c_i$, $0 \leq i \leq n$, has degree at most $n - \theta$, where $c_0 = H(L, n, \theta, m, t_1, \cdots, t_n)$.*

The tuple is a CDS2-type SAG signature *if the following all hold*

1. *Each tuple (t_i, c_i, s_i) is a valid PoK conversation w.r.t. PK_i, $1 \leq i \leq n$.*
2. *For each $1 \leq j \leq \theta$, $\sum_{1 \leq i \leq n} i^j c_i = H_j(L, n, \theta, m, t_1, \cdots, t_n)$.*

Remark: To conserve bandwidth, the representation of an SAG signature can be shortened. For example, if $(t_1, \cdots, t_n)$ can be efficiently constructed from $(c_1, \cdots, c_n, s_0, \cdots, s_n)$, then it can be omitted from the representation. In ring-type SAG signatures, $(c_2, \cdots, c_n)$ can be further omitted since they can be constructed from $(c_1, s_1, \cdots, s_n)$.

A Construction of Ring-Type SAG Signatures [16]: Given a list of public keys $L = \{PK_1, \cdots, PK_n\}$, a message m, a suitable hash function H, a secret key SK_π corresponding to PK_π, a ring-type SAG signature can be constructed as follows:

1. Randomly generate a commitment $\mathcal{T}_\pi$.
2. For each $i = \pi + 1, \cdots, n, 1, \cdots, \pi - 1$, compute $\mathcal{C}_i = H(L, m, \mathcal{T}_{i-1})$ and then simualate a PoK conversation $(\mathcal{T}_i, \mathcal{C}_i, \mathcal{S}_i)$ w.r.t. PK_i.
3. For $i = \pi$, compute $\mathcal{C}_i = H(L, m, \mathcal{T}_{i-1})$ and then compute a PoK conversation $(\mathcal{T}_i, \mathcal{C}_i, \mathcal{S}_i)$ using the secret key SK_i.
4. Output SAG signature $(L, n, \theta = 1, m, \sigma)$ where $\sigma = (\mathcal{C}_1, \mathcal{S}_1, \cdots, \mathcal{S}_n)$ (thus achieving bandwidth conservation).

A Construction of CDS1-Type (Resp. CDS2-Type) SAG Signature [10]: Given list of public keys $L = \{PK_1, \cdots, PK_n\}$, message m, suitable hash function H. Let $I \subset \{1, \cdots, n\}$, $|I| = t$. Given secret keys $\{SK_\pi : \pi \in I\}$, generate SAG signature as follows:

1. For each $i \notin I$, simulate a PoK conversation $(\mathcal{T}_i, \mathcal{C}_i, \mathcal{S}_i)$.
2. For each $\pi \in I$, randomly pick $\mathcal{T}_\pi$.
3. Compute $\mathcal{C}_0 = H(L, n, \theta, m, \mathcal{T}_1, \cdots, \mathcal{T}_n)$, and solve for $\mathcal{C}_\pi$'s, $\pi \in I$, such that the polynomial f interpolated from $f(i) = \mathcal{C}_i$, $0 \le i \le n$, has degree no more than $n - \theta$. (resp. for CDS2-type, solve for $\mathcal{C}_\pi$'s, $\pi \in I$, such that $\sum_{1 \le i \le n} i^j \mathcal{C}_i = H_j(L, n, \theta, m, t_1, \cdots, t_n)$, $1 \le j \le \theta$.)
4. For each $\pi \in I$, compute a PoK conversation $(\mathcal{T}_\pi, \mathcal{C}_\pi, \mathcal{S}_\pi)$ using SK_π.
5. Output an SAG signature $(L, n, \theta, m, \sigma)$ where $\sigma = (f, \mathcal{S}_1, \cdots, \mathcal{S}_n)$ (achieving bandwidth conservation).

Properties of SAG Signatures: The SAG signature has statistical ZK (zero-knowledge) about its actual signers. Therefore, the signer anonymity is unconditional and exculpable. Furthermore, the SAG signature is a group signature which requires essentially no setup, especially in terms of group key setup or secret sharing of the group key. Any one user can conscript the public keys of another $n-1$ users to form an SAG signature without the participation or even knowledge of the conscripted diversion signers. Such properties make SAG signatures useful in diverse applications including whistle blowing[16], e-voting [13], and ad hoc group cryptography [6].

2.4 Generic Group Model (GGM)

We will use the generic group model of [14, 20, 17]. Some highlights below.

Only a restricted set of operations are allowed. They include random generation of integers and group elements, group computations, exponentiations, equality tests. There are only two data types: *group elements* and *non-group data*.

It is assumed the the discrete logarithm problem is uncomputable in the GGM[14].

We restrict ourselves to a polynomial number of steps. Therefore, there are only a polynomial number of unassociated group elements base g, public keys $y_1, \cdots, y_n$, commitments $t_1, \cdots, t_{q_B}$, randomly generated group elements $u_1, \cdots, u_{q_G}$. The computation transcript at each step τ consists of

$$f_\tau = g^{a_{\tau,-1}} \prod_i y_i^{a_{\tau,i}} \prod_{i'} t_{i'}^{b_{\tau,i'}} \prod_{i''} u_{i''}^{c_{\tau,i''}} \tag{1}$$

Each computation can only depend on parameters in existence before that step, resulting in zero exponent for parameters that come into existence after that step.

Probabilities of hash collisions, discrete logarithm collisions, integer computation collisions are all assumed negligible. (Except those resulting from BlindSign Oracle queries.)

3 Blind SAG Signature and Security Model

3.1 The Real World

A blind SAG signature scheme is a tuple (KeyGen, SAGWarden, $\mathsf{BlindSigner}_{PK_1}$, $\cdots$, $\mathsf{BlindSigner}_{PK_n}$, SAGVerifier) where

KeyGen: Upon input a security parameter 1^λ and generates public-private key pair (PK, SK).

SAGVerifier: Upon input a tuple $(L, n, \theta, m, \sigma)$, outputs ACCEPT or REJECT.

$\mathsf{BlindSigner}_{PK_1}$, $\cdots$, $\mathsf{BlindSigner}_{PK_n}$ are (ordinary) $\mathsf{BlindSigner}_{PK}$ protocols defined in the last Section.

SAGWarden: Upon input L', n', θ', m', it picks $I \subset \{1, \cdots, n'\}$, $|I| = \theta'$, and by invoking $\mathsf{BlindSigner}_{PK_i}$, $i \in I$, produces an SAG signature $(L', n', \theta', m', \sigma')$.

3.2 The Ideal World

1. $\mathcal{SO}$ (Signing Oracle): Upon input a public key PK' and any message m', it outputs a valid signature σ'.
2. SAGSign (SAG Signing Oracle): Upon input public key list L', length n', threshold θ', message m', it outputs a valid SAG signature $(L', n', \theta', m, \sigma')$.
3. BlindSign (Blind Signing Oracle): Upon query, it conducts a 4-move interactive protocol with the querier $\mathcal{Q}$ as follows:
   (a) Move-0: $\mathcal{Q}$ sends PK'.
   (b) Move-1: BlindSign sends a *commitment* t to $\mathcal{Q}$.
   (c) Move-2: $\mathcal{Q}$ sends a challenge c to BlindSign.
   (d) Move-3: BlindSign returns s such that (t, s) forms a valid PoK w.r.t. PK'.
4. Random Oracle: Upon receiving a query, it outputs a random number. All query-reply pairs are kept in record and no same reply for different queries.

3.3 Definitions of Security Notions

Definition 2. *(Completeness) If all parties are honest in following the protocols, then the output of the interactions with various oracles will produce valid signatures.*

Game UF

1. (Setup) Upon input a security parameter 1^λ, generate parameters n, θ, and invoke KeyGen n times to generate key pairs (SK_i, PK_i), $1 \leq i \leq n$. The above, except the secret keys, are published.

2. A forger, $\mathcal{F}$ makes q_B (resp. q_S, q_H, q_A) queries to the BlindSigner (resp. $\mathcal{SO}$, random oracle, SAGSign).
3. $\mathcal{F}$ delivers $> q_B/\theta$ valid SAG signatures $(L_i, n_i, \theta, m_i, \sigma_i)$, $1 \leq i \leq q_B + 1$, none of which coincides with any SAGSign query output.

Remark: For simplicity, we require $\mathcal{F}$ to deliver SAG signatures with the same threshold θ, and each public key used in SAG signatures delivered by $\mathcal{F}$ must have been generated in the Setup Phase of Game UF. In this paper, we restrict ourselves to at most a polynomially many queries in terms of the security parameter.

Definition 3. *(Parallel One-more Unforgeability (p1m-uf)) A blind SAG signature scheme is parallel one-more unforgeable (against adaptive chosen-message, chosen-public-key active attackers) if no PPT adversary can successfully complete Game UF with non-negligible probability.*

Remark: Specializing to $n = \theta = 1$, the above definition is defining p1m-uf of classic blind signatures.

Definition 4. *(Blindness) A blind SAG signature scheme has* blindness *if the probability distribution of the signature produced by Warden is indistinguishable from the probability distribution of the signatures produced by Warden conditioned on the blindsign conversation that produced it.*

Roughly speaking,

$$\Pr\left\{\begin{array}{l|l}\text{SAG signature} & \text{BlindSign Oracle} \\ \text{by Warden} & \text{conversation}\end{array}\right\} = \Pr\left\{\begin{array}{l}\text{SAG signature} \\ \text{by Warden}\end{array}\right\}$$

4 Constructing Blind SAG Signatures

We present the constructions of our blind SAG signatures.

4.1 Blind SAG Signature: CDS-Type [10]

Given a list of n public keys, $L = \{PK_1, \cdots, PK_n\}$, message m, threshold θ, and θ accesses to blind signer w.r.t. public keys from L, the following protocol generates a CDS1-type SAG signature (resp. CDS1-type, CDS2-type) $(L, n, \theta, m, \sigma)$:

1. Select $I \subset \{1, \cdots, n\}$, $|I| = \theta$.
2. For each $i \in \{1, \cdots, n\} \setminus I$, generate PoK triple (t_i, c_i, s_i) w.r.t. PK_i.
3. In θ sessions of the TPoK protocol, one for each $i \in I$, act as Warden equipped with $\mathsf{BlindSigner}_{PK_i}$ w.r.t. PK_i, as follows:
   (a) Obtain commitment t'_i from $\mathsf{BlindSigner}_{PK_i}$, for each $i \in I$.
   (b) For each $i \in I$, compute $\Delta_{s,i}$, $\Delta_{c,i}$, and $t_i = f_t(PK_i, t'_i, \Delta_{c,i}, \Delta_{s,i})$.
   (c) Compute $c_0 = H(L, n, \theta, m, t_1, \cdots, t_n)$.

(d) Compute c_i for all $i \in I$ such that the polynomial f interpolated from $f(i) = c_i$, $0 \le i \le n$, has degree at most $n - \theta$. (resp. for CDS2-type, solve for $\mathcal{C}_i$'s, $i \in I$, such that $\sum_{1 \le i \le n} i^j \mathcal{C}_i = H_j(L, n, \theta, m, t_1, \cdots, t_n)$, $1 \le j \le \theta$.)
(e) For each $i \in I$, compute $c'_i = f_c(PK_i, t'_i, \Delta_{c,i}, \Delta_{s,i}, c_i)$, and send c'_i to $\mathsf{BlindSigner}_{PK_i}$.
(f) For each $i \in I$, receive s'_i from $\mathsf{BlindSigner}$ i, and compute $s_i = f_s(PK_i, t'_i, \Delta_{c,i}, \Delta_{s,i}, c_i, s'_i)$.

4. Output $\sigma = (f, s_1, \cdots, s_n)$.

The blind signature for individual index i is referred to as the *underlying blind signature scheme* of the blind SAG signature scheme.

4.2 Blind SAG Signature: Ring-Type [16, 1]

Given a list of n public keys $L = \{PK_1, \cdots, PK_n\}$, message m and accesses once to $\mathsf{BlindSigner}_{PK_i}$ w.r.t. $PK_i \in L$, the following protocol generates a ring-type SAG signature (L, n, m, σ):

1. Select $\pi \in \{1, \cdots, n\}$.
2. Interact as Warden with $\mathsf{BlindSigner}_{PK_\pi}$ to obtain a commitment t'_π, and compute $t_\pi = f_t(PK_\pi, t'_\pi, \Delta_{c,\pi}, \Delta_{s,\pi})$ with randomly generated $\Delta_{c,\pi}$ and $\Delta_{s,\pi}$).
3. Sequentially for each $i = \pi+1, \cdots, n, 1, \pi-1$, compute $c_i = H(L, m, n, t_{i-1})$, and then simulate a PoK triple (t_i, c_i, s_i) w.r.t. PK_i.
4. Finish the interaction with $\mathsf{BlindSigner}_{PK_\pi}$ by
   (a) Compute and send $c'_\pi = f_c(PK_\pi, t'_\pi, \Delta_{c,\pi}, \Delta_{s,\pi}, c_\pi)$.
   (b) Receive s'_π nd compute $s_\pi = f_s(PK_\pi, t'_\pi, \Delta_{c,\pi}, \Delta_{s,\pi}, c_\pi, s'_\pi)$.
5. Output $\sigma = (c_1, \cdots, c_n, s_1, \cdots, s_n)$.

5 Security Analysis

We prove the completeness, the blindness, and the parallel one-more unforgeability of our blind SAG signature schemes. In the process, we also prove an extension of Schnorr's [17] ROS result from single public key to multiple public keys.

5.1 Multi-Key Parallel One-More Unforgeability of Blind Signature

The following results are well-known.

Theorem 1. *[17] The parallel one-more unforgeability (p1m-uf) of Schnorr (resp. Okamoto-Schnorr) blind signature is equivalent to the ROS Problem in the random oracle model plus the generic group model.*

In Schnorr's security model [17], all queries to blindsign are w.r.t. a single public key PK. We generalize it to *multiple-key parallel one-more unforgeability* (mk-p1m-uf) by allowing the Adversary to query blindsign with K different (PK_i) , $1 \leq i \leq n$, a total of q_B times in order to produce a total of $q_B + 1$ signatures each of which is verifiable against some members of the set of public keys $\{PK_1, \cdots, PK_K\}$. We will need this result.

Theorem 2. *The multiple-key parallel one-more unforgeability (mk-p1m-uf) of Schnorr (resp. Okamoto-Schnorr) blind signature is equivalent to the ROS Problem in the random oracle model plus the generic group model.*

Proof in the Appendix.

5.2 Security of Our Blind SAG Signatures

Theorem 3. (Completeness*) Our blind SAG signature has completeness.*

Proof: Trivial.

Theorem 4. (Blindness) *Assume L, n, θ are fixed. Our ring-type (resp. CDS1-type, CDS2-type) blind SAG signature has blindness provided the underlying blind signature also has it.*

Proof Sketch: Denote the SAGBlindSign session communication transcripts by $\mathcal{K}_i = (\mathcal{T}_i, \mathcal{C}_i, \mathcal{S}_i)$, $1 \leq i \leq \theta$, and the SAG signature in question by $(L, n, \theta, m, \sigma)$ where σ=$(t_1, \cdots, t_n, c_1, \cdots, c_n, s_1, \cdots, s_n)$. By the ZK of the underlying blind signatures, (t_i, c_i, s_i) is ZK w.r.t. $(\mathcal{T}_i, \mathcal{C}_i, \mathcal{S}_i)$. Furthermore, (non-blind) SAG signatures are ZK about which secret key actually generated it. Therefore σ is ZK. □

Theorem 5. *(*Unforgeability*) Our ring-type (resp. CDS1-type with $\theta = 1$, CDS2-type with $\theta \geq 1$) SAG blind signature based on Schnorr or Okamoto-Schnorr blind signature is parallel one-more unforgeable (p1m-uf) provided Schnorr's ROS Problem is hard, in the generic group model (GGM) plus the random oracle model (ROM).*

Proof in the Appendix.
Remark: The reduction in Theorem 1 is actually to the ROS Problem or the Discrete Logarithm Problem (DLP). The reduction in Theorem 2 (resp. Theorem 5) is actually to the ROS Problem or the *one-more discrete log (1mDL)* problem. (The 1mDL Problem: compute all discrete logarithms $\log_g y_i$ for $1 \leq y \leq q_{DL}+1$, given g and y_1, $\cdots$, $y_{q_{DL}+1}$ and a total of q_{DL} queries to a Corruption Oracle, which returns the discrete logarithms of qualified query values.) In the GGM, it can be deducted that the probability of computing dicrete log collisions, which include DLP and 1mDL, is negligible for PPT algorithms.

6 Concluding Remarks

We have constructed blind SAG signatures, both ring-type and CDS-type. We have reduced their parallel one-more unforgeability against adaptive chosen-plaintext, adaptive chosen-public-key attackers, to the parallel one-more unforgeability of the component blind signature, and a candidate hard problem, in two cases: where the underlying blind signature is the Schnorr (resp. Okamoto-Schnorr) blind signature.

The security and privacy (anonymity) of the blind SAG signature based on Schnorr blind signature is an interesting topic. The result of Schnorr[17] reduced the security of the Schnorr blind signature to the ROS (Randomized Oversampled Solvable system) Assumption. Recently, Wagner [21] gave a sub-exponential time algorithm to solve the ROS problem. If the array entries are all elements of a binary field, then the ROS Problem can be solved in polynomial time by a method from [2] or [12].

It will be interesting to generalize Schnorr ROS reduction to the Schnorr-based blind SAG signature. Since Schnorr identification scheme does not have zero-knowledge, it will also be interesting to explore the exact zero-knowledge properties of that blind SAG signature.

Acknowledgements. Helpful discussions with Duncan S. Wong are acknowledged.

References

1. M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n signatures from a variety of keys. In *Proc. ASIACRYPT 2002*, pages 415–432. Springer-Verlag, 2002. Lecture Notes in Computer Science No. 2501.
2. M. Bellare and D. Micciancio. a new paradigm for collision-free hasing: incrementality at reduced cost. In *Proc. EUROCRYPT 97*. Springer-Verlag, 1997. Lecture Notes in Computer Science No. 1233.
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. 1st ACM Conference on Computer and Communications Security*, pages 62–73. ACM Press, 1993.
4. G. R. Blakley. Safeguarding cryptographic keys. In *Proc. AFIPS National Computer Conference*, volume 48, pages 313–317, 1979.
5. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Proc. EUROCRYPT 2003*, pages 416–432. Springer-Verlag, 2003. Lecture Notes in Computer Science No. 2656.
6. E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *Proc. CRYPTO 2002*, pages 465–480. Springer-Verlag, 2002. Lecture Notes in Computer Science No. 2442.
7. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Proc. EUROCRYPT 2001*, pages 93–118. Springer-Verlag, 2001. Lecture Notes in Computer Science No. 1294.
8. D. Chaum. Blind signatures for untraceable payments. In *Proc. CRYPTO 82*, pages 199–203. Plenum Press, 1982.

9. D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *CACM*, 29(10):1030–1044, 1985.
10. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Proc. CRYPTO 94*, pages 174–187. Springer-Verlag, 1994. Lecture Notes in Computer Science No. 839.
11. Y. Desmedt. Some recent research aspects of threshold cryptography. In *Proc. First International Workshop on Information Security, ISW 97*, pages 158–173. Springer-Verlag, 1997. Lecture Notes in Computer Science No 1196.
12. J. K. Liu, V. K. Wei, and D. S. Wong. Cryptanalyzing Bresson, et al.'s spontaneous anonymous group threshold signature for ad hoc groups and patching via updating Cramer, et al.'s threshold proof-of-knowledge. *eprint*, 2004(042), 2004.
13. J. K. Liu, V. K. Wei, and D. S. Wong. Linkable and culpable ring signatures. *eprint*, 2004(027), 2004.
14. V.I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes 55*, pages 165–172, 1994.
15. T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *Proc. CRYPTO 92*, pages 31–53. Springer-Verlag, 1993. Lecture Notes in Computer Science No. 740.
16. R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Proc. ASIACRYPT 2001*, pages 552–565. Springer-Verlag, 2001. Lecture Notes in Computer Science No. 2248.
17. C. P. Schnorr. Security of blind discrete log signatures against interactive attacks. In *ICICS*. Springer, 2001. Lecture Notes in Computer Science No. 2229.
18. C.P. Schnorr. Efficient identication and signatures for smart cards. In *Proc. CRYPTO 89*, pages 239–252. Springer-Verlag, 1990. Lecture Notes in Computer Science No. 435.
19. A. Shamir. How to share a secret. In *Communications of the ACM*, volume 22(2), pages 612–613. ACM Press, 1979.
20. V. Shoup. Lower bounds for discrete logarithms and related problems. In *Proc. EUROCRYPT 97*, pages 256–266. Springer-Verlag, 1997. Lecture Notes in Computer Science No. 1233.
21. D. Wagner. A generalized birthday problem. In *Proc. CRYPTO 2002*, pages 288–303. Springer-Verlag, 2002. Lecture Notes in Computer Science No. 2442.

A Proof Sketch of Theorem 2

We mimick Schnorr's [17] proof. The generic mk-p1m attacker is as follows:

1. Obtain commitments: $t_{k,i}$, $1 \le k \le K$, $1 \le i \le q_{B,k}$; where $\sum_k q_{B,k} = q_B$.
2. Compute and then send challenges $c_{k,i}$, $1 \le k \le K$, $1 \le i \le q_{B,k}$.
3. Receive responses $s_{k,i}$. Output $q_B + 1$ signatures $(\hat{t}_{\ell,j}, \hat{s}_{\ell,j})$ on messages $\hat{m}_{\ell,j}$ where $\hat{t}_{\ell,j} = g^{\hat{s}_{\ell,j}} y_\ell^{\hat{c}_{\ell,j}}$, $\hat{c}_{\ell,j} = H(\hat{t}_{\ell,j}, \hat{m}_{\ell,j})$; and $1 \le \ell \le K$, $1 \le j \le \hat{q}_{B,\ell}$, $\sum_\ell \hat{q}_{B,\ell} = q_B + 1$

The oracle conversations can be arbitrarily interleaved. The hash query $\hat{c}_{\ell,j} = H(\hat{t}_{\ell,j}, \hat{m}_{\ell,j})$ must have been made.

Let $f_{\tau(\ell,j)} = \hat{c}_{\ell,j}$, for some index mapping τ.

In Eq(1), we can treat $u_i = y_{q_B+i}$. The u_i's can be used as public keys in querying the Signing Oracle. If they are not used as such, then set $q_{B,q_B+i} = 0$.

They cannot be used as public keys in the delivered signatures, if the conditions so require. Therefore, we can omit the u's w.l.o.g. Expanding the subscript of the t's from one to two according to the current convention, we obtain

$$\begin{aligned} f_{\tau(\ell,j)} &= g^{\hat{s}_{\ell,j}} y_\ell^{\hat{c}_{\ell,j}} \\ &= g^{a_{\tau(\ell,j),-1}} \prod_{k'} y_{k'}^{a_{\tau(\ell,j),k'}} \prod_k \prod_i t_{k,i}^{b_{\tau(\ell,j),k,i}} \\ &= g^{a_{\tau(\ell,j),-1}} \prod_{k'} y_{k'}^{a_{\tau(\ell,j),k'}} \prod_k \prod_i (g^{s_{k,i}} y_k^{c_{k,i}})^{b_{\tau(\ell,j),k,i}} \end{aligned}$$

and

$$1 = g^{\Delta_{s,\ell,j}} \prod_{k'} y_{k'}^{\Delta_{c,\ell,j,k'}}, \text{ for each } \ell, j,$$

where

$$\begin{aligned} \Delta_{s,\ell,j} &= -\hat{s}_{\ell,j} + a_{\tau(\ell,j),-1} + \sum_k \sum_i s_{k,i} b_{\tau(\ell,j),k,i} \\ \Delta_{c,\ell,j,k'} &= -\hat{c}_{\ell,j} \delta(\ell, k') + a_{\tau(\ell,j),k'} + \sum_i c_{k',i} b_{\tau(\ell,j),k',i}. \end{aligned}$$

where the Kronecker delta $\delta(u, v) = 1$ when $u = v$ and equals 0 otherwise. Note that the last two Δ-coefficients are computable by the generic adversary, but not by the Simulator. Therefore rewinding will not enable the Simulator to extract any secret key.

Case (1): $\Delta_{s,\ell,j} = \Delta_{c,\ell,j,k'} = 0$ for all ℓ, j, k'. Then the generic adversary has solved the ROS Problem:

$$\hat{c}_{\ell,j} = a_{\tau(\ell,j),\ell} + \sum_i c_{\ell,i} b_{\tau(\ell,j),\ell,i}, \text{ all } \ell, j.$$

where $\hat{c}_{\ell,j}$'s are $q_B + 1$ hash outputs.

Case (2): the opposite. Then the generic adversary has computed a nontrivial linear dependence among discrete logrithms of $y_{k'}$, i.e. the generic adversary has solved the one-more discrete logarithm problem.

Remark: In the generic group model (GGM), the above linear dependence is a form of *discrete logarithm collision*. It can be deducted in GGM that the probability of a PPT algorithm being able to compute a discrete logarithm collision, including the kind above, is negligible. □

B Proof of Theorem 5

We prove for **CDS1-type**, $\theta = 1$, first. The generic attacker in GGM of p1m-uf of blind Schnorr SAG signature is as follows:

1. Input: a list of public keys $L = \{y_1, \cdots, y_n\}$.
2. Receive commitments $t_{k,i}$, $1 \le i \le q_{B,k}$ from $\mathsf{BlindSigner}_{PK_i}$. Note $\sum_k q_{B,k} = q_B$.

3. Send challenges $c_{k,i}$, receive responses $s_{k,i}$.
4. Output SAG signatures $\sigma_j = (\hat{t}_{j,1}, \cdots, \hat{t}_{j,n}, \hat{c}_{j,1}, \cdots, \hat{c}_{j,n}, \hat{s}_{j,1}, \cdots, \hat{s}_{j,n})$ on message $\hat{m}_j$, $1 \leq j \leq q_B + 1$.

The queries $\hat{c}_{j,0} = H(L, n, \theta, \hat{m}_j, \hat{t}_{j,1}, \cdots, \hat{t}_{j,n})$, $1 \leq j \leq q_B + 1$, must have been made. Let the Lagrange interpolation be indicated $\sum_{0 \leq \ell' \leq n} \gamma_{\ell'} \hat{c}_{j,\ell'} = 0$. By GGM, there exists a index mapping τ such that, for $1 \leq j \leq q_B + 1$ and $1 \leq \ell \leq n$,

$$\begin{aligned}
\hat{t}_{j,\ell} &= g^{\hat{s}_{j,\ell}} y_\ell^{\hat{c}_{j,\ell}} \\
&= g^{a_{\tau(j,\ell),-1}} \prod_{\ell'} y_{\ell'}^{a_{\tau(j,\ell),\ell'}} \prod_{k,i} t_{k,i}^{b_{\tau(j,\ell),i}} \\
&= g^{a_{\tau(j,\ell),-1}} \prod_{\ell'} y_{\ell'}^{a_{\tau(j,\ell),\ell'}} \prod_{k,i} (g^{s_{k,i}} y_k^{c_{k,i}})^{b_{\tau(j,\ell),i}} \\
1 &= g^{\Delta_{s,j,\ell}} \prod_{\ell'} y_{\ell'}^{\Delta_{c,j,\ell,\ell'}} \text{ where} \\
\Delta_{s,j,\ell} &= -\hat{s}_{j,\ell} + a_{\tau(j,\ell),-1} + \sum_{k,i} s_{k,i} b_{\tau(j,\ell),k,i}, \text{ all } j, \ell \\
\Delta_{c,j,\ell,\ell'} &= -\hat{c}_{j,\ell} \delta(\ell, \ell') + \sum_{\ell'} a_{\tau(j,\ell),\ell'} + \sum_{k,i} c_{k,i} b_{\tau(j,\ell),k,i}, \text{ all } j, \ell, \ell'
\end{aligned}$$

The negligibility of discrete logarithm collision leads to

$$\begin{aligned}
0 &= -\hat{c}_{j,\ell'} + \sum_{\ell'} a_{\tau(j,\ell'),\ell'} + \sum_{k,i} c_{k,i} b_{\tau(j,\ell'),k,i}, \text{ all } j, \ell \\
-\gamma_0 \hat{c}_{j,0} &= \sum_{\ell'=1}^{n} \gamma_{\ell'} \hat{c}_{j,\ell'} \\
&= \sum_{1 \leq \ell' \leq n} \gamma_{\ell'} (\sum_{\ell'} a_{\tau(j,\ell'),\ell'} + \sum_{k,i} c_{k,i} b_{\tau(j,\ell'),k,i})
\end{aligned}$$

for $1 \leq j \leq q_B + 1$. The generic adversary has solved the above ROS Problem, where $\hat{c}_{j,0}$ are $q_B + 1$ hash outputs.

CDS2-type, $\theta \geq 1$. Similar to the above, with the following modificaitons:

The hash queries are $\hat{c}_{j,0}^{(\theta')} = H_{\theta'}(L, n, \theta, \hat{m}_j, \hat{t}_{j,1}, \cdots, \hat{t}_{j,n})$, $1 \leq j \leq q_B + 1$, $1 \leq \theta' \leq \theta$, have been made. The ROS Problem is

$$\begin{aligned}
-\gamma_0^{(\theta')} \hat{c}_{j,0}^{(\theta')} &= \sum_{\ell'=1}^{n} \gamma_{\ell'}^{(\theta')} \hat{c}_{j,\ell'} \\
&= \sum_{1 \leq \ell' \leq n} \gamma_{\ell'}^{(\theta')} (\sum_{\ell'} a_{\tau(j,\ell'),\ell'} + \sum_{k,i} c_{k,i} b_{\tau(j,\ell'),k,i})
\end{aligned}$$

where $\hat{c}_{j,0}^{(\theta')}$'s are $q_B + 1$ hash outputs expressed in terms of q_B commitments $c_{k,i}$'s.

ring-type: The proof is similar and omitted. □