Tracing-by-Linking Group Signatures

Victor K. Wei

Dept. of Information Engineering, Chinese University of Hong Kong, Hong Kong
kwwei@ie.cuhk.edu.hk

September 14, 2005

Abstract. In a group signature [19], any group member can sign on behalf of the group while remaining anonymous, but its identity can be traced in an future dispute investigation. Essentially all state-of-the-art group signatures implement the tracing mechnism by requiring the signer to escrow its identity to an Open Authority (OA) [2, 14, 4, 25, 5, 7, 24]. We call them *Tracing-by-Escrowing (TbE)* group signatures. One drawback is that the OA also has the unnecessary power to trace without proper cause. In this paper we introduce *Tracing-by-Linking (TbL)* group signatures. The signer's anonymity is irrevocable by any authority if the group member signs only once (per event). But if a member signs twice, its identity can be traced by a public algorithm without needing any trapdoor. We initiate the formal study of TbL group signatures by introducing its security model, constructing the first examples, and give several applications. Our core construction technique is the successful transplant of the TbL technique from single-term offline e-cash from the blind signature framework [10, 22, 21] to the group signature framework. Our signatures have size $O(1)$.

1 Introduction

In a group signature [19], any group member can sign on behalf of the group while remaining anonymous. However, to investigate a dispute, the signer's identity can be *traced*. Essentially all contemporary state-of-the-art group signatures implement the tracing mechanism by requiring the signer to escrow its identity to an Open Authority (OA) [2, 14, 4, 25, 5, 7]. We call them *Tracing-by-Escrowing (TbE)* group signatures. One drawback is that the OA's trapdoor has the unnecessary power to trace any signature without proper cause. For example, a change in government or administration can mandate the OA to trace some past signatures controversially.

In this paper, we initiate the formal study of *Tracing-by-Linking (TbL) group signatures*. In a TbL group signature, the signer's anonymity cannot be revoked by any combination of authorities. However, if a group member signs twice (per event), then its identity can be traced by any member of the public without needing any trapdoor.

Our main **contributions** are

- We initiate the formal study of tracing-by-linking (TbL) group signatures. We introduce its security model, and construct the first several TbL group signatures, and reduce their securities to standard intractability assumptions.
- We extending our constructions from *sign twice and anonymity revoked* to *sign k times and anonymity revoked*.
- We apply TbL group signatures to several applications, including Direct Anonymous Attestation (DAA), anonymous credentials, offline anonymous e-cash, and e-voting.

The paper is **organized** as follows: Section 2 contains the security model. Section 3 contains preliminaries. Section 4 contains constructions and security theorems. Section 5 contains discussions and applications.

Related Results: Essentially all state-of-the-art group signatures are TbE group signatures. The signer anonymity can be revoked by the OA's trapdoor even without cause. Partial key escrows and time-delayed key escrows [36, 30, 3] have been introduced to counteract abuses by the over-powered. The TbL group signature's anonymity is irrevocable by any combination of managers and authorities. There is no OA. In a

ring signature [20, 35, 1] the signer anonymity is also irrevocable. But signing any number of times does not result in anonymity revocation. In a linkable group (resp ring) signature scheme [32, 33, 15, 23, 12], signatures from the same signer can be linked, but its anonymity remains. These *link-but-not-trace* group (resp. ring) signatures typically *tag* the double signer in a way such that future signatures from the same signer can be linked more conveniently.

Our intuitions: The core of our construction technique is the successful transplant of the TbL technique from single-term offline e-cash scheme from the blind signature framework [9–11, 21, 22] to the group signature framework. Our TbL group signature has size $O(1)$. The essence of our TbL technique is to commit some randomness during group membership certification and then require the signer to use these randomness during a 3-move non-interactive zero-knowledge proof. Double spending implies answering challenges twice with the same *certified commitments* and it results in the extraction of the double signer's secret identity.

2 Security Model

We present a security model for the tracing-by-linking (TbL) group signature. In a nutshell, we replace the triplet of security notions, *anonymity*, *full traceability* and *non-frameability*, of TbE group signatures [4, 5] by a new triplet for TbL group signatures: *irrevocable anonymity*, *full linkability* and *non-slanderability*. Motivated by the DAA (Direct Anonymous Attestation) [12] application, our system consists of three types of entities in multitudes:

- Managers of Groups, or, equivalently, Certificate Authorities (CA's), with serial number cnt which stands for *counter value*.
- Users, or, equivalently, TPM (Trusted Platform Module), whose serial number is id.
- Verifiers with serial number bsn which stands for *basename*.

Having multiple CA's is equivalent to having multiple groups, and thus our model extends the single-group models of [4, 25, 5]. Having multiple verifiers allows multiple signatures, one set per verifier serial number bsn, and thus increases its usefulness of TbL group signatures.

Syntax. A TbL group signature is a tuple (Init, GKg, UKg, Join, Iss, GSig, GVf, Link, Indict) where:

- Init: $1^\lambda \rightarrow \mathsf{param}$. On input the security parameter 1^λ, Protocol init generates public system parameters param. Included: an efficiently samplable one-way NP-relation whose specification is $\langle \mathcal{R}_{user} \rangle$, an efficiently samplable family of trapdoor one-way NP-relations whose specifications constitute $\mathcal{F}$=$\{\langle \mathcal{R}_{CA,i} \rangle : i\}$ and whose trapdoors are denoted gsk_i's, and an initially-empty list of generated users denoted UL, and an initially-empty list of generated groups denoted GL.
- GKg:$\mathsf{cnt} \xrightarrow{\$} \langle \mathcal{R}_{CA,\mathsf{cnt}} \rangle$. On input cnt, Protocol GKg samples $\mathcal{F}$ to get a relation whose specification is $\langle \mathcal{R}_{CA,\mathsf{cnt}} \rangle$ and whose trapdoor is gsk_{cnt}, and adds an entry $(\mathsf{cnt}, \langle \mathcal{R}_{CA,cnt} \rangle)$ to the group list GL. By convention, $\langle \mathcal{R}_{CA,\mathsf{cnt}} \rangle$ includes gpk_{cnt}.
- UKg: $\mathsf{id} \xrightarrow{\$} (usk_{\mathsf{id}}, upk_{\mathsf{id}}) \in \mathcal{R}_{user}$. Protocol UKg accepts input id to sample a key pair from $\mathcal{R}_{user}$, adds an entry $(\mathsf{id}, upk_{\mathsf{id}})$ to the user list UL.
- Join,Iss is a pair of interactive protocols with common inputs $\mathsf{cnt} \in GL$ and $\mathsf{id} \in UL$, and Iss's addition input gsk_{cnt}, and Join's additional inputs usk_{id}. At the conclusion, join obtains extended secret key $xsk_{\mathsf{id},\mathsf{cnt}}$, extended public key $xpk_{\mathsf{id},\mathsf{cnt}}$ which includes a certificate $cert_{\mathsf{id},\mathsf{cnt}}$ satisfying $(xpk_{\mathsf{id},\mathsf{cnt}}, cert_{\mathsf{id},\mathsf{cnt}}) \in \mathcal{R}_{CA,\mathsf{cnt}}$ and $(xsk_{\mathsf{id},\mathsf{cnt}}, xpk_{\mathsf{id},\mathsf{cnt}}) \in \mathcal{R}_{user}$, such that Iss does not know $xsk_{\mathsf{id},\mathsf{cnt}}$, and an entry $(\mathsf{id}, \mathsf{cnt}, xpk_{\mathsf{id},\mathsf{cnt}})$ is added to the public system parameters param. Below, we may sometimes use the notations xpk (resp. xsk) and upk (resp. usk) interchangeably without ambiguity from context.
- GSig: (id, cnt, $xsk_{\mathsf{id},\mathsf{cnt}}$, bsn, M) $\rightarrow \sigma$. It takes inputs $\mathsf{id} \in UL$, $\mathsf{cnt} \in GL$, $xsk_{\mathsf{id},\mathsf{cnt}}$, bsn, and a message M, returns a signature σ. By convention, the *extended signature* σ includes cnt, bsn, and M. Optionally, an additional input $\mu_{\mathsf{id},\mathsf{cnt},\mathsf{bsn}}$ can be included to bookkeep the number of times signatures have been generated for each triple (id, cnt, bsn).
- GVf: $(\sigma, \mathsf{cnt}, \mathsf{bsn}) \rightarrow 0$ or 1. It takes input a signature σ, returns either 1 or 0 for valid or $\mathsf{invalid}$. If σ is an extended signature, then it includes cnt and bsn.

- Link: $(\sigma_1, \cdots, \sigma_{k+1}) \rightarrow 0$ or 1. It takes inputs $k+1$ valid signatures, σ_i, $1 \leq i \leq k+1$, returns either 1 or 0 for linked or unlinked.
- Indict: $(\sigma_1, \cdots, \sigma_{k+1}) \rightarrow \mathsf{id}$. It takes $k+1$ valid and linked signatures σ_i, $1 \leq i \leq k+1$, returns id.

Definition 1 (Correctness). *For integer i, $1 \leq i \leq k+1$, let σ_i=GSig (id_i, cnt_i, $xsk_{\mathsf{id}_i,\mathsf{cnt}_i}$, bsn_i, M_i). A TbL group signature has* verification correctness *if GVf (σ_1)=1 with probability one (or, equivalently, GVf (σ_i)=1 with probability one for each i, $1 \leq i \leq k+1$). It has* linking correctness *if Link $(\sigma_1, \cdots, \sigma_{k+1}) = 0$ with overwhelming probability when the $k+1$ triples (id_i, cnt_i, bsn_i), $1 \leq i \leq k+1$, are not all identical. It has* indictment correctness *if Link $(\sigma_1, \cdots, \sigma_{k+1}) = 1$ and Indict $(\sigma_1, \cdots, \sigma_{k+1}) = \mathsf{id}_1$ with overwhelming probability when the $k+1$ triples (id_i, cnt_i, bsn_i), $1 \leq i \leq k+1$, are all identical. It is* correct *if it has verification correctness, linking correctness, and indictment correctness.*

The following **oracles** are the attacker's tools.

- The *Random Oracle* $\mathcal{H}$: We use the Random Oracle normally.
- The *Corruption Oracle*: $\mathcal{CO} : (\mathsf{id}, \mathsf{cnt}) \rightarrow xsk_{\mathsf{id},\mathsf{cnt}}$. Upon input the $\mathsf{id} \in UL$, it outputs xsk_{id}.
- The *k-Signing Oracle* $\mathcal{SO}_k : (\mathsf{id}, \mathsf{cnt}, \mathsf{bsn}, M) \rightarrow \sigma$. Upon inputs a user $\mathsf{id} \in UL$, a CA $\mathsf{cnt} \in GL$, a verifier bsn, and a message M, it outputs a signature. For each tuple (id, cnt, bsn), at most k query with the same triple (id, cnt, bsn) are allowed regardless of the message M. This restriction reflects the reality that honest group members do not over-sign in TbL group signatures. We adopt the convention that $\mathcal{SO}_k$ will output $NULL$ upon query inputs that repeat a triple (id, cnt, bsn) more than k times.
- The Group Corruption Oracle: $\mathcal{GCO}(\mathsf{cnt})$ accepts input $\mathsf{cnt} \in GL$, and outputs the group trapdoor gsk_{cnt}. The group manager cnt continues to function honestly, but the attacker can observe its communications.
- The Add User Oracle: $\mathcal{AUO}(\mathsf{id})$ adds a user with identity id and sampled sk-pk pair $\mathsf{UKg}(\mathsf{id}) \xrightarrow{\$} (usk_{\mathsf{id}}, upk_{\mathsf{id}})$ to UL.
- The *Join Oracle*: $\mathcal{JO}(\mathsf{id}, usk_{\mathsf{id}}, upk_{\mathsf{id}})$ allows the attacker to interact, in the role of the Join Protocol, with the Iss Protocol.
- The *Issue Oracle*: $\mathcal{IO}(\mathsf{cnt}, gsk_{\mathsf{cnt}}, gpk_{\mathsf{cnt}})$ allows the attacker to interact with the Join Protocol, in the role of the Iss Protocol after the attacker corrupts with $\mathcal{GCO}(\mathsf{cnt})$.

We adopt the *static attacker model* where the attacker corrupts users at the beginning of the security experiments only, and the Simulator knows which users are corrupted. Issues with *adaptive attackers* [17], *reset attackers* [18], or UC (Universal Composability) attackers [16] are left to future research.

Irrevocable k-Anonymity. Irrevocable anonymity for TbL group signature is defined in the following experiment.

Experiment IA(k).

1. (*Initialization Phase*) Simulator $\mathcal{S}$ invokes Init, GKg, invokes UKg (resp. Join,Iss) $g_u \geq 2$ times to generate a set of joined users, with their extended user public keys xpk_{id}'s.
2. (*Probe-1 Phase*) $\mathcal{A}$ queries $\mathcal{GCO}$, $\mathcal{AUO}$, $\mathcal{CO}$, $\mathcal{JO}$, $\mathcal{H}$, $\mathcal{SO}_k$ in arbitrary interleaf.
3. (*Gauntlet Phase*) $\mathcal{A}$ selects a gauntlet group cnt_{ga}, two members who have joined this group, denoted *gauntlet users* id_0, id_1, a gauntlet verifier bsn_{ga}, and a message M for $\mathcal{S}$. Then $\mathcal{S}$ flips a fair coin $b \in \{0,1\}$ and returns the *gauntlet signature* $\sigma_{ga} = \mathsf{GSig}(\mathsf{id}_b, \mathsf{cnt}_{ga}, xsk_{\mathsf{id}_b,\mathsf{cnt}_{ga}}, \mathsf{bsn}_{ga}, M)$.
4. (*Probe-2 Phase*) $\mathcal{A}$ queries $\mathcal{GCO}$, $\mathcal{AUO}$, $\mathcal{CO}$, $\mathcal{JO}$, $\mathcal{H}$, $\mathcal{SO}_k$ in arbitrary interleaf.
5. (*End Game*) $\mathcal{A}$ delivers an estimate $\hat{b} \in \{0,1\}$ of b.

$\mathcal{A}$ *wins* Experiment IA(k) if $\hat{b} = b$,it has queried $\mathcal{SO}_k(\mathsf{id}, \mathsf{cnt}_{ga}, \mathsf{bsn}_{ga}, M')$ no more than $k-1$ times with $\mathsf{id} = \mathsf{id}_0$ (resp. $\mathsf{id} = \mathsf{id}_1$), and it has never queried $\mathcal{CO}(\mathsf{id}_0, cnt_{ga})$ or $\mathcal{CO}(\mathsf{id}_1, \mathsf{cnt}_{ga})$. The restriction on $\mathcal{SO}_k$ queries is trivially necessary because the TbL mechanism together with enough such $\mathcal{SO}_k$ queries wins Experiment IA(k) outright. $\mathcal{A}$'s *advantage* in Experiment IA(k) is its probability of winning, minus half. Oracle queries can be arbitrarily interleaved across Probe-1, Gauntlet, and Probe-2 Phases.

Definition 2 (Irrevocable k-Anonymity). *An TbL group signature is* irrevocably k-anonymous *if no PPT algorithm has a non-negligible advantage in Experiment IA(k). When $k = 1$, it is* irrevocably anonymous.

Remark: The irrevocable anonymity is a kind of computational zero-knowledge about the signer identity. In comparison, some stronger anonymity models in ring signatures allow $\mathcal{A}$ to corrupt the gauntlet users, and/or achieve statistical zero-knowledge. On the other hand, irrevocable anonymity allows $\mathcal{A}$ to corrupt all authorities while many TbE group signature models do not.

The k-linkability, the full k-linkability. Roughly speaking, full k-linkability means that any coalition of $q_e ll$ corrupted users, without the group manager, cannot produce $kq_\ell + 1$ valid signatures for the same $(\mathsf{cnt}, \mathsf{bsn})$ that are not linked to any colluder. The case $q_\ell = 0$ corresponds to unforgeability of the signature, and the case $q_\ell = 1$ corresponds to k-linkability. Formally, Full k-Linkability is defined in terms of the following experiment.

Experiment FL(k).
1. (*Initialization Phase*) $\mathcal{S}$ invokes Init, GKg, invokes UKg (resp. Join,Iss) a polynomially many times.
2. (*Probe-1 Phase*) $\mathcal{A}$ makes q_G (resp. q_A, q_C, q_J, q_H, q_S) queries to $\mathcal{GCO}$ (resp. $\mathcal{AUO}$, $\mathcal{CO}$, $\mathcal{JO}$, $\mathcal{H}$, $\mathcal{SO}_k$) in arbitrary interleaf.
3. $\mathcal{A}$ delivers cnt, bsn, and signatures σ_i for $1 \le i \le k(q_J + \hat{q}_C) + 1$, where cnt has never been queried to $\mathcal{GCO}$, each the signatures satisfies $\mathsf{GVf}(\sigma_i, \mathsf{cnt}, \mathsf{bsn}) = 1$ and is not the output of an $\mathcal{SO}_k$ query, and $\hat{q}_C$ is the total number of Join's by all users created in the Initialization Phase and corrupted by querying $\mathcal{CO}$.

$\mathcal{A}$ *wins* Experiment FL(k) if $\mathsf{Link}(\sigma_{i_1}, \cdots, \sigma_{i_{k+1}})$ does not output any of the corrupted users for arbitrary $1 \le i_1 < \cdots < i_{k+1} \le k(q_J + \hat{q}_C) + 1$. $\mathcal{A}$'s *advantage* is its probability of winning.

Definition 3 (Full k-Linkability). *A TbL group signature is* fully k-linkable *if no PPT algorithm has a non-negligible advantage in Experiment FL(k). It is* fully linkable *in the case $k = 1$.*

Non-slanderability In a nutshell, non-slanderability means a coalition of users together with the group manager cannot produce signatures that are linked and indicted to a group member outside the coalition. Formally,

Experiment NS(k)
1. $\mathcal{S}$ invokes Init, GKg, invokes UKg (resp. Join,Iss) a polynomially many times.
2. $\mathcal{A}$ queries $\mathcal{GCO}$, $\mathcal{AUO}$, $\mathcal{CO}$, $\mathcal{JO}$, $\mathcal{H}$, $\mathcal{SO}_k$ in arbitrary interleaf.
3. $\mathcal{A}$ delivers k+1 valid signatures, σ_i, $1 \le i \le k$+1, such that $\mathsf{Link}(\sigma_1, \cdots, \sigma_{k+1}) = 1$, and $\mathsf{Indict}(\sigma_1, \cdots, \sigma_{k+1}) = \mathsf{id}$ where id has never been queried to $\mathcal{CO}$.

$\mathcal{A}$ *wins* Experiment NS(k) if it completes. Its *advantage* is his probability of winning.

Definition 4 (k-Non-Slanderability). *A TbL group signature is* k-non-slanderable *if no PPT adversary has a non-negligible advantage in Experiment NS(k). It is* non-slanderable *in the case k=1.*

Summarizing, we have:

Definition 5 (Security). *A TbL group signature is* k-secure *if it is correct, irrevocably k-anonymous, fully k-linkable, and k-non-slanderable. It is* secure *in the case $k = 1$.*

Remark: There is a slightly weaker security model, the *SbV-secure TbL group signature* where SBV stands for *Slanderable-but-Vindicatable*. It is otherwise secure like secure TbL group signatures, except it allows the indictment of some non-guilty users who can subsequently *vindicate* themselves via an additional Protocol $\mathsf{Vindicate}$. Only those who are indicted but cannot vindicate themselves are truly guilty. We exhibit a SbV-secure TbL group signature in Appendix A of the full paper [40].

Related security notions. Exculpability (cf. misidentification attack [25]) of TbE group signature means that a coalition of users, together with the group manager (and the open authority) cannot produce a signature traced to an uncorrupted member. The TbL group signature does not have the "open signature" functionality of TbE group signatures, and the notion of exculpability is, in a sense, absorbed into into non-slanderability. Non-slanderability implies that the *indictment* is accurate: If user id_g is indicted, then non-slanderability implies no one except id_g could have generated the double signing.

3 Preliminaries

We say N is a *safe product* if $N = pq$, $p = 2p' + 1$, $q = 2q' + 1$, p, q, p', and q' are sufficiently large primes. The set of *quadratic residues* in Z_N, denoted QR_N, consists of all squares in Z_n.

Strong RSA Assumption There exists no PPT algorithm which, on input a random λ_s-bit safe product N and a random $z \in QR_N$, returns $u \in \mathbb{Z}_N^*$ and $e \in \mathbb{N}$ such that $e > 1$ and $u^e = z(\mathrm{mod}N)$, with non-negligible probability and in time polynomial in λ_s.

We will need the DDH (Decisional Diffie-Hellman) Assumption across two groups, with possibly different orders: Let G_a and G_b be two groups. The **DDH(G_a, G_b) Problem** is, given random $g, g^\alpha \in G_1$ and $h \in G_b$, distinguish h^α from random, where $0 < \alpha < \min\{\mathrm{order}(G_a), \mathrm{order}(G_b)\}$. The **DDH($g_a$, g_b) Assumption** is that no PPT algorithm can solve the DDH(G_z, G_b) Problem with non-negligible probability.

The q-Strong Diffie-Hellman (q-SDH) Assumptions Let $e : G_1 \times G_2 \to G_3$ be a pairing, with $q_i = \mathrm{order}(G_i)$, $1 \le i \le 3$. The *q-Strong Diffie-Hellman Problem (q-SDH)* is the problem of computing a pair $(g_1^{1/(\gamma+x)}, x)$ given $(g_1 \in G_1, g_2, g_2^\gamma, g_2^{\gamma^2}, \cdots, g_2^{\gamma^q} \in G_2$, and a homomorphism $\psi(g_2) = g_1$. The *q-SDH Assumption* is that no PPT algorithm has a non-negligible probability of solving a random sample of the q-SDH Problem. For further details, see [7].

Related is the **q-CAA (Coalition Attack Algorithm) Assumption** [31, 42], that no PPT algorithm can solve the q-CAA Problem: Given random $g_2, g_2^\gamma \in G_2$, and pairs (A_i, e_i) with distinct and nonzero e_i's satisfying $A_i^{\gamma+e_i} = v$, $1 \le i \le q$, compute a pair (A_{q+1}, e_{q+1}) with $e_{q+1} \neq e_i$ for any i, $1 \le i \le q$, and satisfying $a_{q+1}^{\gamma+e_{q+1}} = v$. Appendix B contains several equivalence reductions among the q-SDH Assumption, the q-CAA Assumption, and some other related assumptions.

4 Constructing TbL Group Signatures

We construct two TbL group signatures in pairings and in the strong RSA framework, respectively. The former signature is typically short, but involves expensive pairings computation by the Verifier. The latter signature is typically fast, but not as short as the former signature. The constructions in this Section has k=1. Constructions with $k > 1$ are discussed in the next Section. Also, for simplicity, we assume there is only one group cnt and thus omit the group index from the notations.

4.1 Generic construction: intuitions

We describe the intuition of our constructions in a loosely specific generic construction.

The user sk-pk pair is $((x_1, x_2, x_3, x_4), \mathsf{PedCom}(x_1, x_2, x_3; x_4)$, where PedCom is Pedersen's commitment [34], and $x_1 = U$ is the user identity. The group membership certificate is $\mathsf{cert} = \mathsf{Sign}_{gsk}(\mathsf{PedCom}(x_1, x_2, x_3; x_4))$, where Sign is a signature by the Group Manager, i.e. the group certificate issuer. The group signature is the following signature proof-of-knowledge (SPK):

$$\begin{aligned}\sigma = SPK\{(x_1, x_2, x_3, x_4, \mathsf{cert}) : \mathsf{cert} = \mathsf{Sign}_{gsk}(\mathsf{PedCom}(x_1, x_2, x_3; x_4)) \\ \wedge\ z = x_2 - cx_1 \text{ where } c \text{ is the challenge of this SPK } \wedge\ \mathsf{serial} = \mathbf{g}^{x_3}\}(\mathsf{param}, nonce, M)\end{aligned} \tag{1}$$

The group signature verifier, Protocol GVf, verifies the proof σ. Protocol Link outputs 1=linked if and only if two valid input signatures have the same serial. Protocol Indict outputs the signer identity $u = x_1$ computed from the following linear equations: $z = x_2 - cx_1$ and $z' = x_2 - c'x_1$, respectively, from two linked signatures.

The intuition in the security analysis is as follows: The anonymity comes the perfect statistical hiding property of Pedersen commitments. The full linkability and the non-slanderability are straightforward.

4.2 Instantiating in pairings: Protocol TbL-SDH

We construct a TbL group signature in pairings [31, 42, 6, 7], and reduce its security to intractability assumptions.

Init, GKg: Generate a pairing $\hat{e} : G_1 \times G_2 \rightarrow G_T$. Generate all discrete logarithm bases fairly, e.g. $g_i = \mathcal{H}('g', i) \in G_1$, $\mathbf{g} = \mathcal{H}('\mathbf{g}', i) \in G_T$, $h_i = \mathcal{H}('h', i) \in G_1$, $u_i = \mathcal{H}('u', i) \in G_2$. The group sk-pk pair is (γ, u^γ). Generates a user list UL which is initially empty. We adopt the flexible notation $\mathcal{H}$ that it is a full-domain cryptographically secure hash function mapping from a union of mixed domains to a union of mixed ranges. Ambiguity should not arise from the context.

UKg: On input id, sample the relation $\mathcal{R}_{user}$ to output $(usk_{\mathsf{id}}, upk_{\mathsf{id}})$.

Protocols Join,Iss: accepts common inputs id, Join's additional input usk_{id}, Iss's additional input γ and proceed as follows:

1. Protocol Join identitifes itself as User id with knowledge of usk_{id}, and then sets $x_1 = \mathsf{id}$ and randomly generates x'_2, x'_3, x'_4. Presents $upk_{\mathsf{id}} = h_1^{x_1} h_2^{x'_2} h_3^{x'_3} h_4^{x'_4}$; proves knowledge of x'_2, x'_3, and x'_4.
2. Protocol Iss verifies the proofs; randomly generates x''_2, x''_3, x''_4; gives them and a certificate (A, e) satisfying $A^{e+\gamma} h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4} = h_0 \in G_1$ to Protocol Join, where $x_i = x'_i + x''_i$, $2 \leq i \leq 4$ and. Then Protocol Iss inserts the entry $(\mathsf{id}, xpk = h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4}, cert = (A, e))$ into UL which is considered part of the public param.

Protocol GSig(id, $xsk_{\mathsf{id}} = (A, e, x_1, x_2, x_3, x_4)$, M): It outputs signature σ which is a signature proof of knowledge (non-interactive zero-knowledge proof-of-knowledge) of the following proof system

$$SPK\{\{(A, e, x_1, x_2, x_3, x_4) : A^{\gamma+e} h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4} = h_0 \in G_1 \\ \wedge\ x_2 = cx_1 + z \text{ where } c \text{ is the challenge of this proof } \wedge\ S = \mathbf{g}_{\mathsf{bsn}}^{x_3}\}(\mathsf{param}, nonce, M) \tag{2}$$

Further instantiation details of GSig are below: The commitments are

$$T_A = A g_A^{s_A}, \quad [\text{Note } \hat{e}(h_0, u)\hat{e}(T_A, u^\gamma)^{-1} \\ = \hat{e}(T_A, u)^e \hat{e}(h_1, u)^{x_1} \hat{e}(h_2, u)^{x_2} \hat{e}(h_3, u)^{x_3} \hat{e}(h_4, u)^{x_4} \hat{e}(g_A, u^\gamma)^{-s_A} \hat{e}(g_A, u)^{-s_0} \text{ where } s_0 = es_A] \\ D_0 = \hat{e}(T_A, u)^{r_e} \hat{e}(h_1, u)^{r_{x,1}} \hat{e}(h_2, u)^{r_{x,2}} \hat{e}(h_3, u)^{r_{x,3}} \hat{e}(h_4, u)^{r_{x,4}} \hat{e}(g_A, u^\gamma)^{-r_A} \hat{e}(g_A, u)^{-r_0} \\ T_1 = g_1^{x_1} g_2^{s_1}, \quad T_2 = g_1^{x_2} g_2^{s_2}, \quad S = \mathbf{g}_{\mathsf{bsn}}^{x_3}, \quad D_1 = g_1^{r_{x,1}} g_2^{r_1}, \quad D_2 = g_1^{r_{x,2}} g_2^{r_2}, \quad D_3 = \mathbf{g}_{\mathsf{bsn}}^{r_{x,3}} \tag{3}$$

where $r_{x,1} = x_2$, $r_1 = s_2$, $T_2 = D_1$. Observe the secrets and randomnesses are mixed, using the technique from Section 3. The challenge is:

$$c = \mathcal{H}(\mathsf{param}, nonce, M, T_A, T_1, T_2, S, D_0, D_1, D_2, D_3) \tag{4}$$

The responses are:

$$z_A = r_A - cs_A,\ z_e = r_e - ce,\ z_i = r_i - cs_i \text{ for } 0 \leq i \leq 2,\ z_{x,i} = r_{x,i} - cx_i \text{ for } 1 \leq i \leq 4.$$

The signature is:

$$\sigma = (\mathsf{param}, nonce, M, T_A, T_1, T_2, S, c, z_A, z_e, z_0, z_1, z_2, z_{x,1}, z_{x,2}, z_{x,3}, z_{x,4})$$

Protocol GVf(σ) parses the input, computes

$$D_0 = \hat{e}(T_A, u)^{z_e} \hat{e}(h_1, u)^{z_{x,1}} \hat{e}(h_2, u)^{z_{x,2}} \hat{e}(h_3, u)^{z_{x,3}} \hat{e}(h_4, u)^{z_{x,4}} \\ \cdot \hat{e}(g_A, u^\gamma)^{-z_A} \hat{e}(g_A, u)^{-z_0} [\hat{e}(h_0, u)\hat{e}(T_A, u^\gamma)^{-1}]^c, \\ D_1 = g_1^{z_{x,1}} g_2^{z_1} T_1^c, \quad D_2 = g_1^{z_{x,2}} g_2^{z_2}, \quad D_3 = \mathbf{g}_{\mathsf{bsn}}^{z_{x,3}} S^c \tag{5}$$

Verifies D_1 equals the parsed T_2 and the challenge c computed from Equation (4) equals to that parsed from the input.

Protocol Link(σ, σ'): Verifies the validity of both input signatures. Parses the inputs into $\sigma = (\cdots, S, \cdots)$ and $\sigma' = (\cdots, S', \cdots)$. Outputs 1 if $S = S'$, and outputs 0 otherwise.

Protocol Indict(σ, σ'): Verifies the validity of both input signatures. Parses both signatures. Solves x_1 from $x_2 = cx_1 + z_{x,1}$ and $x_2 = c'x_1 + z'_{x,1}$. Outputs $\mathsf{id} = x_1$.

The following Theorem analyzes the security of Protocol TbL-SDH.

Theorem 1. *Protocol TbL-SDH is a TbL group signature with $k = 1$ which, assuming the Random Oracle (RO) model,*

1. *is* correct*;*
2. *is* irrevocably anonymous *provided the DDH Assumption holds in G_T;*
3. *is* fully linkable *provided the q-SDH Assumption holds;*
4. *is* non-slanderable *provided Discrete Logarithm is hard.*

In summary, Protocol TbL-SDH is a secure TBL group signature *if the q-SDH Assumption holds and the DDH Assumption in G_T holds in the RO model.*

Proof Sketch: Correctness is straightforward. Noting the user public key $xpk = \mathsf{PedCom}(x_1, x_2, x_3; x_4)$ is a Pedersen commitment [34], our irrevocable anonymity follows from the *perfect statistical hiding* property of Pedersen commitments, except that the anonymity of the *serial*, S, reduced to the DDH Assumption in the group G_T.. Noting our group signature proof system in Equation (2) proves more relations than the group signature proof system of Boneh and Boyen [6], our full linkability follows from a proof similar to [6]'s proof of full traceability. Bu the same token, our proof of non-slanderability follows a similar path to [6]'s proof of non-frameability.

One extra point worth additional attention is our simulation of the Signing Oracle $\mathcal{SO}(\mathsf{id}, \mathsf{cnt}, M)$. It is simulated as follows:

1. Retrieve user public key xpk_{id} from the user database UL.
2. Randomly generate the challenge c.
3. Randomly generate $z_{x,1}$, z_1, T_1, and compute D_1 according to Equation (5).
4. Set $T_2 = D_1$.
5. Randomly generate the remaining responses $z_{x,i}$ for $2 \leq i \leq 4$, z_i for $0 \leq i \leq 2$, S, and T_A. Compute D_0, D_2, D_3 from Equation (5). Backpatch the random oracle in Equation (4).

It can be shown that this simulation is indistinguishable from the real world Protocol GSig. □

4.3 Instantiating in strong RSA: Protocol TbL-SRSA

We also construct a TbL group signature in the strong RSA framework [2, 14].

Protocols Init, GKg: Given security parameter 1^{λ_s}, initializes additional security parameters $\epsilon > 1$, ℓ_p, λ_1, λ_2, γ_1, γ_2 satisfying $\lambda_1 > \epsilon(\lambda_2 + \lambda_s) + 3$, $\lambda_2 > 4\ell_p$, $\gamma_1 > \epsilon(\gamma_2 + \lambda_s) + 3$, and $\gamma_2 > \lambda_1 + 3$. Note ℓ_p sets the size of the modulus of the RSA framework, ϵ controls the tightness of the zero-knowledge [2]. Generate a product N of two ℓ_p-bit safe primes p and q, i.e. $p = 2p' + 1$, $q = 2q' + 1$, p' and q' are both primes. Generate a known-order group $G_S = \langle \mathbf{g} \rangle$, where $\text{order}(G_S) = p_s$ which is a $(\gamma_1 + 1)$-bit prime. The signature-signing group sk-pk pair is $((p, q), N)$. Generates a user list UL which is initially empty. Let all discrete logarithm bases be fairly generated, e.g. $g_i = \mathcal{H}('g', i) \in QR_N$, $h_i = \mathcal{H}('h', i) \in QR_N$, $\mathbf{g}_i = \mathcal{H}('\mathbf{g}', i) \in G_S$. We adopt the flexible notation $\mathcal{H}$ which is a full-domain collision-resistant hash functin mapping from a union of mixed domains to a union of mixed ranges. Ambiguity should not arise from the context. Denote the intervals $\Gamma =]2^{\lambda_1} - 2^{\lambda_2}, 2^{\lambda_1} + 2^{\lambda_2}[$ and $\Gamma =]2^{\gamma_1} - 2^{\gamma_2}, 2^{\gamma_1} + 2^{\gamma_2}[$.

Protocols Join,Iss: accepts common inputs id, Join's additional input usk_{id}, Iss's additional input γ and proceed as follows:

1. Protocol Join authenticates itself as User id. It is required that $\mathsf{id} \in \Lambda$. Then it sets $x_2 = \mathsf{id}$ and randomly generates x'_1, x'_3, $x'_4 \in]-2^{\lambda_2 - 1}, 2^{\lambda_2 - 1}[$. Presents $upk_{\mathsf{id}} = h_1^{x'_1} h_2^{x_2} h_3^{x'_3} h_4^{x'_4}$. Prove knowledge of x'_1, x'_3, $x'_4 \in]-2^{\lambda_2 - 1}, 2^{\lambda_2 - 1}[$, and $x_2 = \mathsf{id}$.
2. Protocol Iss verifies the proofs; randomly generates x''_1, x''_3, $x''_4 \in]-2^{\lambda_2 - 1}, 2^{\lambda_2 - 1}[$; creates a certificate (A, e) satisfying $A^e h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4} = h_0$ for Protocol Join, where $x_i = 2^{\lambda_1} + x'_i + x''_i \in \Lambda$, $i = 1, 3, 4$, and $e \in \Gamma$ is a prime. Then Protocol Iss inserts the entry $(\mathsf{id}, xpk = h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4}, cert = (A, e))$ into UL which is considered part of the public param.

For simplicity, assume each x'_i (resp. x''_i) is even so all user public keys are in QR_N.

Protocol $\mathsf{GSig}(\mathsf{id}, xsk_{\mathsf{id}} = (A, e, x_1, x_2, x_3, x_4), M)$: It outputs signature σ which is a signature proof of knowledge (non-interactive zero-knowledge proof-of-knowledge) of the following proof system: (Range checks are omitted for simplicity. For details see Boudot [8].)

$$SPK\{\{(A, e, x_1, x_2, x_3, x_4) : A^e h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4} = h_0 \in QR_N \\ \wedge\ x_3 = cx_2 + z \text{ where } c \text{ is the challenge of this proof } \wedge\ S = h_{\mathsf{bsn}}^{x_4}\}(\mathsf{param}, nonce, M)$$

Further instantiation details of GSig are below: The commitments are

$$\begin{gathered} T_0 = g_0^{s_0},\ \ T_A = Ag_A^{s_0},\ \ [\text{Note } h_0 = T_A^e h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4} g_A^{-s_1} \text{ where } s_1 = es_0, \text{ and } 1 = T_0^e g_0^{-s_1}] \\ T_e = g_{e,1}^e g_{e,2}^{s_0},\ \ T_2 = g_{2,1}^{x_2} g_{2,2}^{s_0},\ \ T_3 = g_{3,1}^{x_3} g_{3,2}^{s_0},\ \ T_4 = g_{4,1}^{x_4} g_{4,2}^{s_0},\ \ S = h_{\mathsf{bsn}}^{x_3}, \\ D_0 = g_0^{r_0},\ \ D_A = T_A^{r_e} h_1^{r_{x,1}} h_2^{r_{x,2}} h_3^{r_{x,3}} h_4^{r_{x,4}} g_A^{-r_1},\ \ D_1 = T_0^{r_e} g_0^{-r_1},\ \ D_e = g_{e,1}^{r_e} g_{e,2}^{r_0}, \\ D_2 = g_{2,1}^{r_{x,2}} g_{2,2}^{r_0},\ \ D_3 = g_{3,1}^{r_{x,3}} g_{3,2}^{r_0},\ \ D_4 = g_{4,1}^{r_{x,4}} g_{4,2}^{r_0},\ \ D_s = h_{\mathsf{bsn}}^{r_{x,3}}, \\ T_5 = \mathbf{g}_5^{x_2} \mathbf{g}_6^{s_3},\ \ T_6 = \mathbf{g}_5^{x_3} \mathbf{g}_6^{s_4},\ \ D_5 = \mathbf{g}_5^{r_{x,2}} \mathbf{g}_6^{r_3},\ \ D_6 = \mathbf{g}_5^{r_{x,3}} \mathbf{g}_6^{r_4} \end{gathered} \tag{6}$$

where $r_{x,2} = x_3$, $r_3 = s_4$ and therefore $D_5 = T_6$. The challenge is:

$$\begin{aligned} c = \mathcal{H}(&\mathsf{param}, nonce, M, T_0, T_A, T_e, \\ &T_2, T_3, T_4, T_5, T_6, S, D_0, D_A, D_e, D_1, D_2, D_3, D_4, D_s, D_5, D_6) \end{aligned} \tag{7}$$

The responses are:

$$z_e = r_e - ce,\ \ z_i = r_i - cs_i \text{ for } 0 \le i \le 4,\ \ z_{x,i} = r_{x,i} - cx_i \text{ for } 1 \le i \le 4.$$

The signature is:

$$\begin{aligned} \sigma = (&\mathsf{param}, nonce, M, T_0, T_A, T_e, S, \\ &T_2, T_3, T_4, T_5, c, z_e, z_0, z_1, z_2, z_3, z_4, z_{x,1}, z_{x,2}, z_{x,3}, z_{x,4}) \end{aligned}$$

Protocol $\mathsf{GVf}(\sigma)$ parses the input and computes

$$\begin{gathered} D_0 = g_0^{z_0} T_0^c,\ \ D_A = T_A^{z_e} h_1^{z_{x,1}} h_2^{z_{x,2}} h_3^{z_{x,3}} h_4^{z_{x,4}} g_A^{-z_1} T_A^c,\ \ D_1 = T_0^{z_e} g_0^{-z_1}, \\ D_e = g_{e,1}^{z_e} g_{e,2}^{z_0} T_e^c,\ \ D_2 = g_{2,1}^{z_{x,2}} g_{2,2}^{z_0} T_2^c,\ \ D_3 = g_{3,1}^{z_{x,3}} g_{3,2}^{z_0} T_3^c,\ \ D_4 = g_{4,1}^{z_{x,4}} g_{4,2}^{z_0} T_4^c, \\ D_s = \mathbf{g}_{\mathsf{bsn}}^{z_{x,3}} S^c,\ \ D_5 = \mathbf{g}_5^{z_{x,2}} \mathbf{g}_6^{z_3} T_5^c,\ \ T_6 = D_5,\ \ D_6 = \mathbf{g}_5^{z_{x,3}} \mathbf{g}_6^{z_4} T_6^c \end{gathered}$$

Verifies that the challenge c computed from Equation (7) equals to that parsed from the input.

Protocol $\mathsf{Link}(\sigma, \sigma')$: Verifies the validity of both input signatures. Parses the inputs into $\sigma = (\cdots, S, \cdots)$ and $\sigma' = (\cdots, S', \cdots)$. Outputs 1=linked if $S = S'$, and outputs 0 otherwise.

Protocol $\mathsf{Indict}(\sigma, \sigma')$: Verifies the validity of both input signatures. Parses both signatures. Solves x_2 and x_3 from $x_3 = cx_2 + z_{x,3}$ and $x_3 = c'x_2 + z'_{x,3}$ in $\pmod{p_s}$ where p_s is the (known) order of the group G_S. Confirm range $x_2, x_3 \in \Lambda$. Outputs $\mathsf{id} = x_2$.

The following Theorem analyzes the security of Protocol TbL-SRSA.

Theorem 2. *Protocol TbL-SRSA is a TBL group signature with $k = 1$ which, assuming the Random Oracle (RO) model,*

1. *is* correct*;*
2. *is* irrevocably anonymous *provided the DDH Assumption holds in QR_N;*
3. *is* fully linkable *under the strong RSA Assumption;*
4. *is* non-slanderable *provided Discrete Logarithm is hard in QR_N.*

In summary, Protocol TbL-SRSA is a secure TbL group signature *provided the strong-RSA Assumption and the DDH Assumption both hold in QR_N in the RO model.*

Proof Sketch: Correctness is straightforward. Noting the user public key $xpk = \mathsf{PedCom}(x_1, x_2, x_3; x_4)$ is a Pedersen commitment [34], our irrevocable anonymity follows from the *perfect statistical hiding* property of Pedersen commitments, except that the anonymity of the *serial*, S, reduces to the DDH Assumption in QR_N. Noting our group signature proof system in Equation (2) proves more relations than the group signature proof system of Ateniese, et al. [2], our full linkability follows from a proof similar to [2]'s proof of full traceability. By the same token, our non-slanderability follows the no-frameability of [2].

The simulation of the Signing Oracle $\mathcal{SO}(\mathsf{id}, \mathsf{cnt}, M)$ is simular to the simulation of $\mathcal{SO}$ in Theorem 1, and omitted. □

4.3.1 Protocol TbL-SRSA2: with higher indictment complexity. Another TbL group signature, which we call *Protocol TbL-SRSA2*, can be modified from Protocol TbL-SRSA by setting $G_S = QR_N$. Then all elements used in the signature are in the group QR_N whose order is known only to the Group Manager. All (sub-)protocols of TbL-SRSA remains the same in TbL-SRSA2, except Protocol Indict.

The Protocol $\mathsf{Indict}(\sigma, \tilde{\sigma})$ is modified as follows: Proceed as before in TbL-SRSA. The solution of x_2 and x_3 from $x_3 = cx_2 + z_{x,3}$ and $x_3 = c'x_2 + z'_{x,3}$ in $\pmod{\text{order}(QR_N)}$ results in $x_2 = \bar{x}_2/\Delta$ and $x_3 = \bar{x}_3/\Delta$ where Δ is the determinant of the solution. For each user in the user database, test if $\mathsf{id} \cdot \Delta \stackrel{?}{=} \bar{x}_2$. If match, Protocol Indict outputs id.

Now the computation complexity of this Indict protocol is proportional to the number of listed users, whereas the complexity of TbL-SRSA's Indict is $O(1)$. But indictment should be a rare event.

The reductionist security of Protocol TbL-SRSA2 is essentially the same as that of Protocol TbL-SRSA.

5 Discussions, Applications, Conclusions

Link-and-trace other than identity. Our TbL group signatures above link-and-trace the signer identity. However, it can be easily modify to link-and-trace the user secret key. In fact, Protocol TbL-SDH (resp. TbL-SRSA) already links-and-traces all four user secret keys x_1, x_2, x_3, x_4.

Hybridizing TbL and TbE. If we also require the TbL group signature to incorporate a verifiable escrow of the signer identity to an OA (Open Authority), then the doubly-signed signatures can be traced by the TbL mechanism and the singly-signed signatures can be traced by the OA. For example, Protocol TbL-SDH becomes

$$SPK\{\{(A, e, x_1, x_2, x_3, x_4, \rho) : A^{\gamma+e} h_1^{x_1} h_2^{x_2} h_3^{x_3} h_4^{x_4} = h_0 \ \wedge \ x_3^{-1} = cx_1^{-1} + z \\ \wedge x_4^{-1} = cx_2^{-1} + z' \ \wedge \ c \text{ is the challenge } \wedge \ \mathsf{ctxt} = \mathsf{Enc}(\mathsf{pk}_{OA}, h_1^{x_1}, \rho)\}(M)$$

Sign $k+1$ times and be traced. Our TbL group signature traces after a user signs twice, within the same group and for the same verifier. It can be extended to trace after signing k times with $k > 2$ as follows. Option One: Issue each certificate with k committed randomnesses. Then the certificate can be used k times within the same group and for the same verifier without revealing the identity. However, signing $k+1$ times necessarily uses a certain committed randomness twice and thus results in anonymity revocation. Option Two: Each verifier provides k sets of discrete-log bases for each group. Then a user with a TbL certificate can sign k times using the different bases without anonymity revocation. Signing $k+1$ times necessarily uses a certain Verifier-specific discrete logarithm bases set twice and thus results in identity extraction.

Applications Linkable group (resp. ring) signatures or their equivalents, in the link-but-not-trace paradigm, have been proposed for several applications, including e-cash [27, 15, 37, 29, 28, 39, 38], e-voting [32, 33, 26, 39, 38], DAA (Direct Anonymous Attestation) [13], and anonymous credentials [15]. In each application, an entitled user access its privileges by anonymously authenticate itself with a group (resp. ring) signature or an equivalent mechanism such as anonymous group authentication, anonymous credentials, or DAA. In order to regulate resource use, double or multiple signatures (or *sign-on*'s) by the same user entitlement are linked (but not raced) and countermeasures taken. Typically, a *blacklist* of offenders' *tags* [15, 13] is published to facilitate future detection of linked signatures.

When presented with a signature, the verifier checks its validity, and then confirms it is not on the blacklist before accepting. The blacklist can be available online, or locally cached, or both. Unavoidably, there are

synchronization and latency issues with the updating and the availability of the blacklist. An attacker can exploit such vulnerabilities by launching concurrent sign-on sessions with multiple verifiers during network congestion when the blacklist updating and inter-verifier communications are stressed. Then the *offline* verifier faces a Hobson's choice of either (1) probationally accepts valid signatures (sign-on's) without checking an updated blacklist and suffer the potential damage, or (2) summarily rejects all signatures to err on the safe side.

The TbL paradigm of link-and-trace is an effective deterrant against the above vulnerability exploitation. An offline verifier can probationally accept valid signatures without an updated blacklist. Double signers can be traced and penalized afterwards when the blacklist eventually returns online and is updated. The TbL paradigm does not *prevent* the exploitation, but it is an effective deterrant. Its deterrent effect can also alleviate the urgency of the availability and the updating of the blacklist. We observe that the TbL deterrant is relatively more desirable in a scalable offline anonymous e-cash application when the offline scenario is highly realistic; while it is relatively less significant in an anonymous e-voting scheme where the vote tallying is after the detection of double votes.

Conclusion We initiate the formal study of the TbL (Tracing-by-Linking) group signatures. We introduce its security model, and we present the first several instantiations with provable security. Our main construction technique is the mix-the-secrets-and-randomnesses (MSR) technique. It remains interesting to discover alternative mechanisms to achieve TbL other than MSR, to reduce complexity and bandwidth costs, and to construct more flexible instantiations to achieve more versatile features.

Acknowledgements to Hong Kong RGC Earmarked Grants 4232-03E for financial support, and to Shaoquan Jiang for proofreading my errors.

References

1. M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n signatures from a variety of keys. In *ASIACRYPT 2002*, pages 415–432. Springer-Verlag, 2002.
2. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO 2000*, pages 255–270. Springer-Verlag, 2000.
3. M. Bellare and S. Goldwasser. Verifiable partial key escrow. In *ACM-CCS 1997*, pages 78–91, 1997.
4. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: formal definitions, simplified requirements and a construction based on general assumptions. In *EUROCRYPT'03*, volume 2656 of *LNCS*. Springer-Verlag, 2003.
5. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer-Verlag, 2005. Also ePrint 2004/077.
6. D. Boneh and X. Boyen. Short signatures without random oracles. In *Eurocrypt 2004*, volume 3027 of *LNCS*, pages 56–73. Springer–Verglag, 2004.
7. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO 2004*, pages 41–55. Springer-Verlag, 2004. Lecture Notes in Computer Science No. 3152.
8. F. Boudot. Efficient proofs that a committed number lies in an interval. In *Eurocrypt'00*, pages 431–444, 2000.
9. S. Brands. An efficient off-line electronic cash system based on the representation problem. Technical Report CS-R9323, CWI, CWI, April 1993.
10. S. Brands. Untraceable off-line cash in wallet with observers. In *CRYPTO'93*, pages 302–318. Springer-Verlag, 1993.
11. S. Brands. Untraceable off-line cash in wallets with observers. manuscript, CWI, 1993.
12. E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. Cryptology ePrint Archive, Report 2004/205, 2004. http://eprint.iacr.org/.
13. E. F. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In *ACM-CCS 2004*, pages 132–145, 2004. Also ePrint 2004/205.
14. J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *SCN 2002*, volume 2576 of *LNCS*, pages 268–289. Springer-Verlag, 2003.
15. J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *SCN 2002*, volume 2576 of *LNCS*, pages 268–289. Springer-Verlag, 2003.
16. R. Canetti. Universal composable security: a new paradigm for cryptographic protocols. In *FOCS 2001*, pages 136–145. IEEE Computer Society, 2001.

17. R. Canetti, R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Adaptive security for threshold cryptosystems. In *CRYPTO 1999*, volume 1666 of *LNCS*, pages 98–115. Springer-Verlag, 1999.
18. Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Aresettable zero-knowledge. In *STOC 2000*, pages 235–244. ACM Press, 2000.
19. D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer-Verlag, 1991.
20. R. Cramer, I. Damgard, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO'94*, pages 174–187. Springer-Verlag, 1994.
21. N. Ferguson. Extensions of single-term coins. In *CRYPTO'93*, pages 292–301. Springer-Verlag, 1993.
22. N. Ferguson. Single term off-line coins. In *Eurocrypt'93*, pages 318–328. Springer-Verlag, 1993.
23. M. Fischlin. The Cramer-Shoup strong-RSA signature scheme revisited. In *PKC*, pages 116–129, 2003.
24. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable signatures. In *Eurocrypt 2004*, LNCS, pages 571–589. Springer-Verlag, 2004.
25. A. Kiayias and M. Yung. Group signatures: provable security, efficient constructions, and anonymity from trapdoor-holders. Cryptology ePrint Archive, Report 2004/076, 2004. http://eprint.iacr.org/.
26. J. K. Liu, V. K. Wei, and D. S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract). In *ACISP'04*, volume 3108 of *LNCS*, pages 325–335. Springer-Verlag, 2004.
27. A. Lysyanskaya and Z. Ramzan. Group blind digital signatures: A scalable solution to electronic cash. In *FC'98*, pages 184–197. Springer-Verlag, 1998.
28. G. Maitland and C. Boyd. Fair electronic cash based on a group signature scheme. In *ICICS'01*, volume 2229 of *LNCS*. Springer-Verlag, 2001.
29. G. Maitland, J. Reid, E. Foo, C. Boyd, and E. Dawson. Linkability in practical electronic cash design. In *ISW 2000*, volume 1975 of *LNCS*, pages 149–163. Springer-Verlag, 2000.
30. S. Micali. Gauranteed partial key escrow. memo 537, MIT, 1995.
31. S. Mitsunari, R. Sakai, and M. Kasahara. A new traitor tracing. IEICE Trans. Fundamentals, E85-A(2):481-4, 2002.
32. T. Nakanishi, T. Fujiwara, and H. Watanabe. A linkable group signature and its application to secret voting. In *4th Int'l Symp. on Communicatin Theory and Appl.*, 1997.
33. T. Nakanishi, T. Fujiwara, and H. Watanabe. A linkable group signature and its application to secret voting. *Trans. of Information Processing Society of Japan*, 40(7):3085–3096, 1999.
34. T. P. Pedersen. A threshold cryptosystem without a trusted third party. In *EUROCRYPT 1991*, volume 547 of *LNCS*, pages 522–526. Springer-Verlag, 1991.
35. R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT 2001*, pages 552–565. Springer-Verlag, 2001.
36. A. Shamir. Partial key escrow: a new approach to software key escrow. presentation at NIST key escrow standards meeting, Sept. 15, 1995.
37. J. Traoré. Group signatures and their relevance to privacy-protecting off-line electronic cash systems. In *ACISP'99*, pages 228–243. Springer-Verlag, 1999.
38. Patrick P. Tsang and Victor K. Wei. Short linkable ring signatures for e-voting, e-cash and attestation. In *ISPEC 2005*, volume 3439 of *LNCS*, pages 48–60. Springer-Verlag, 2005.
39. Patrick P. Tsang, Victor K. Wei, Man Ho Au, Tony K. Chan, Joseph K. Liu, and Duncan S. Wong. Separable linkable threshold ring signatures. In *Indocrypt 2004*, volume 3348 of *LNCS*, pages 384–298. Springer-Verlag, 2004.
40. Victor K. Wei. Tracing-by-linking group signatures. Cryptology ePrint Archive, Report 2004/370, 2004. http://eprint.iacr.org/.
41. Victor K. Wei. Tight reductions among strong Diffie-Hellman assumptions. Cryptology ePrint Archive, Report 2005/057, 2005. http://eprint.iacr.org/.
42. F. Zhang, R. Safavi-Naini, and W. Susilo. An efficient signature scheme from bilinear pairings and its applications. In *PKC 2004*, 2004.

A A slanderable-but-vindicatable secure TbL group signature

We present an SbV (Slanderable-but-Vindicatable) but otherwise secure TbL group signature scheme. It is an extension of the link-but-not-trace ring signature of [39].

For simplicity we consider a static group model where there is no (Join, Iss) within the duration of consideration and there are exactly n members of the group. The group sk-pk pair it ($\emptyset$, $(y_1 = g^{x_1}, \cdots, y_n = g^{x_n})$). (Yes, there is no group trapdoor.) The sk-pk pair of User i is $(x_i,\ y_i)$. The TbL group signature is

$$\sigma = SPK\{x : (\bigvee_{1\le i\le n} y_i = g^x) \wedge (\bigvee_{1\le i\le n} \tilde{y}_i = h^x)\}(M) \tag{8}$$

where $h = \mathcal{H}(nonce)$, i.e. the hash of a non-repeating value. Note that the signature σ necessarily contains $\tilde{y}_1, \cdots, \tilde{y}_n$, and therefore it grows in length proportional to the group size n.

Protocol $\mathsf{GVf}(\sigma)$ verifies the proof. Assume signatures σ and σ' are both valid and they parse to produce $\tilde{y}_i$ and $\tilde{y}'_i$, respectively. Protocol $\mathsf{Link}(\sigma, \sigma')$ outputs 1 if and only if $\tilde{y}_j = \tilde{y}'_j$ for some j, $1 \le j \le n$. And in that case, Protocol $\mathsf{Indict}(\sigma, \sigma')$ outputs j.

An attacker other than User j can produce two valid signatures with the same randomly generated $\tilde{y}_j$, and thus indict User j. But User j can then vindicate itself by proving the inequality of discrete logarithms: $\log_g(y_j) \neq \log_h(\tilde{y}_j)$. □

B Tight Reductions among Strong Diffie-Hellman Assumptions

We derive some tight equivalence reductions between several Strong Diffie-Hellman (SDH) assumptions. This Appendix also appeared as [41].

B.1 Results

Let $e : G_1 \times G_2 \to G_T$ be a bilinear mapping. The *k-Strong Diffie-Hellman Problem (k-SDH)* is the problem of computing a pair $(g_1^{1/(\gamma+x)}, x)$ given $g_1 \in G_1$, and g_2, g_2^{γ}, $g_2^{\gamma^2}$, $\cdots$, $g_2^{\gamma^k} \in G_2$. The *k-Strong Diffie-Hellman Assumption* is that no PPT algorithm has a non-negligible probability of solving a random instance of the k-Strong Diffie Hellman Problem. For details, see [6, 7].

The k-SDH Assumption is closely related to the coalition-resistance of pairing-based signature schemes and group signature schemes [31, 42, 6, 7, 40]. Typically, k colluders cannot jointly forge an additional signature when the k-SDH Assumption holds. The following variants are also related to the coalition-resistance of pairing based signatures and group signatures:

- The *k-SDH' Problem* is the problem of computing a pair $(g_1^{1/(\gamma+x)}, x)$ given g_1, g_1^{γ}, $g_1^{\gamma^2}$, $\cdots$, $g_1^{\gamma^k} \in G_1$ and $g_2, g_2^{\gamma} \in G_2$.
- The *k-CAA Problem* is, given g_2, $g_2^{\gamma} \in G_2$, $v \in G_1$, and and pairs (A_i, e_i) with distinct and nonzero e_i's satisfying $A_i^{\gamma+e_i} = v$, $1 \le i \le k$, compute a pair (A_{k+1}, e_{k+1}) with $e_{k+1} \neq e_i$ for any i, $1 \le i \le k$, and satisfying $A_{k+1}^{\gamma+e_{k+1}} = v$.
- The *k-CAA2 Problem* is, given g_2, $g_2^{\gamma} \in G_2$, u, $v \in G_1$, (A_i, e_i, x_i) satisfying $A_i^{\gamma+e_i} u^{x_i} = v$ for $1 \le i \le k$ and all e_i's are distinct and nonzero, compute another triple $(A_{k+1}, e_{k+1}, x_{k+1})$ satisfying $A_{k+1}^{\gamma+e_{k+1}} u^{x_{k+1}} = v$ and $e_{k+1} \neq n_i$ for any i, $1 \le i \le k$.

The *k-SDH' (resp. k-CAA, k-CAA2) Assumption* is that no PPT algorithm has a non-negligible probability of solving a random instance of the k-SDH' (resp. k-CAA, k-CAA2) Problem. The k-CAA Assumption is from Zhang, et al.[42], where CAA stands for *Collusion Attack Algorithm*. They showed the k-CAA Assumption holds if and only if their group signature scheme is k-coalition resistant. [6, 7] showed the k-CAA Assumption is at least as strong as the k-SDH Assumption. However, no reduction in the opposite direction was given. The *full traceability* of the *exculpable* version of [7]'s group signature in their Section 7 can be easily shown equivalent to the k-CAA2 Assumption. [40] showed the k-CCA2 Assumption is at least as strong as the k-SDH Assumption.

The k-CAA2 Assumption is easily at least as strong as the k-CAA Assumption. Typically, there exists an efficiently computable homomorphism ψ such that $\psi(g_2) = g_1$. Then the k-SDH Assumption is at least as strong as the k-SDH' Assumption. In Section B.2, we prove the following two Theorems:

Theorem 3. *The k-SDH' Assumption and the k-CAA Assumption are equivalent.*

Theorem 4. *Assume the discrete log value* $\log_v(u)$ *is known. Then the k-SDH' Assumption and the k-CAA2 Assumption are equivalent.*

In proving results concerning SDH-related protocols, u is often the output of a fair hashing function. Then the value of $\log_v(u)$ is known under the random oracle model. In such cases, Theorem 4 can be used to establish equivalence between certain SDH-based signature and group signature schemes and the k-SDH' Assumption. It remains intereating to explore other equivalence reductions between these and other SDH-related assumptions, and their applications to pairing-based signatures and group signatures.

We also note that the above equivalence reductions are *tight*, meaning that one solution algorithm's time complexity (resp. success probability) is within a reasonable additive term of the solution algorithm of the other problem. Such tightness will be established by our proofs below.

B.2 Proofs

B.2.1 Proof Sketch of Theorem 3 (1) *Solving k-CAA Problem implies solving k-SDH' Problem.* Assume PPT algorithm $\mathcal{A}$ solves k-CAA. Given a k-SDH' problem instance, randomly generate distinct nonzero e_i, $1 \le i \le k$. Let $f(\gamma) = \prod_{i=1}^{k}(\gamma + e_i)$. Denote $f(\gamma) = \sum_{i=0}^{k} f_i\gamma^i$. Let $v = g_1^{f(\gamma)}$. For $1 \le i \le k$ let $f^{[j]} = f(\gamma)/(\gamma + e_j) = \sum_{i=0}^{k-1} f_i^{[j]}\gamma^i$. Then

$$A_j = v^{1/(\gamma+e_j)} = g_1^{f^{[j]}(\gamma)} = g_1^{\sum_{i=0}^{k-1} f_i^{[j]}\gamma^j} = \prod_{i=0}^{k-1}(g_1^{\gamma^j})^{f_i^{[j]}}$$

Note that for each j, $1 \le j \le k$, we have $A_j^{\gamma+e_j} = v$. Invoking $\mathcal{A}$ to solve this k-CAA Problem, we obtain (A_{k+1}, e_{k+1}) satisfying $A_{k+1}^{\gamma+e_{k+1}} = v$. Denote $B = v^{\hat{f}(\gamma)^{-1}}$ where $\hat{f}(\gamma) = f(\gamma)(\gamma + e_{k+1})$. Next, we describe how to compute B. Denote $\hat{f}(\gamma) = \sum_{i=0}^{k+1} \hat{f}_i\gamma^i$ and

$$\hat{f}^{[j]}(\gamma) = \hat{f}(\gamma)(\gamma + e_j)^{-1} = \prod_{1\le i\le k+1, i\ne j}(\gamma + e_i) = \sum_{i=1}^{k} \hat{f}_i^{[j]}\gamma^i$$

for $1 \le j \le k+1$. Denote $\tilde{e} = e_{k+1}$, we have

$$B^{\gamma^{j+1}+\gamma_j\tilde{e}} = B^{(\gamma^j+\gamma^{j-1}\tilde{e})\gamma} = g_1^j, \text{ for } 0 \le j \le k$$
$$B^{\hat{f}(\gamma)} = v$$

The above system of $k+2$ equations can be solved for the $k+2$ unknowns B^{γ^ℓ}, $0 \le \ell \le k+1$, including B where $(B, \tilde{e})$ solves the k-SDH' Problem.

(2) *Solving k-SDH' Problem implies solving k-CAA Problem.* Assume $\mathcal{A}$ is a PPT solver of the k-SDH' Problem. Given $A_i^{\gamma+e_i}$, $1 \le i \le k$, let $f(\gamma) = \prod_{I=1}^{k}(\gamma + e_i)$. Let $g_1 = v^{1/f(\gamma)}$. Next, we describe how to compute g_1.

Denote $f(\gamma) = \sum_{i=0}^{k} f_i\gamma^i$ and $f^{[j]}(\gamma) = f(\gamma)/(\gamma + e_j) = \sum_{i=0}^{k-1} f_i^{[j]}\gamma^i$, for $1 \le j \le k$. We have $v = g_1^{f(\gamma)} = \prod_{i=0}^{k}(g_1^{\gamma^i})^{f_i}$ and

$$A_j = g_1^{f^{[j]}(\gamma)} = \prod_{i=0}^{k}(g_1^{\gamma^i})^{f_i^{[j]}} \tag{9}$$

Rearranging, we have

$$\prod_{i=0}^{k}(g_1^{\gamma^i})^{M_{i,j}} = A_j, \text{ for } 0 \le j \le k, \tag{10}$$

where the $(k+1) \times (k+1)$ matrix $\bar{\mathbf{M}}$ is

$$\bar{\mathbf{M}} = [M_{i,j}]_{0 \le i,j \le k} = \begin{bmatrix} f_0 & f_1 & \cdots & f_k \\ 0 & f_1^{[1]} & \cdots & f_k^{[1]} \\ \vdots & & & \\ 0 & f_1^{[k]} & \cdots & f_k^{[k]} \end{bmatrix}$$

Note $f_i^{[j]} = \mathsf{S}_{k-1-i}(E \setminus \{e_j\})$ for all i and j, $1 \le i \le k-1$, $1 \le j \le k$, where $E = \{e_1, \cdots, e_k\}$ and $\mathsf{S}_a(\{x_1, \cdots, x_n\})$ is the a-th order symmetric function

$$\mathsf{S}_a(\{x_1, \cdots, x_n\}) = \sum_{1 \le i_1 < \cdots < i_a \le n} x_{i_1} \cdots x_{i_a}$$

Denote the $k \times k$ matrix $\mathbf{M} = [M_{i,j}]_{1 \le i,j \le k}$. We prove the following Lemma later:

Lemma 5 $\det(\mathbf{M}) = \prod_{1 \le i < j \le k}(e_i - e_j)$.

Therefore $\det(\bar{\mathbf{M}}) = (\prod_{\ell=1}^{k} e_\ell)(\prod_{1 \le i,j \le k}(e_i - e_j)) \neq 0$, and Equation (10) can be solved to obtain $g_1^{\gamma^i}$, for all i, $0 \le i \le k$. Invoking the k-SDH' solver $\mathcal{A}$ to obtain $g_1^{1/(\gamma+x)}$ and x.

Let $\bar{f}(\gamma) = \sum_{i=0}^{k-1} \bar{f}_i \gamma^i$ and $\bar{c}$ be such that $f(\gamma)/(\gamma + x) = \bar{f}(\gamma) + \bar{c}/(\gamma + x)$. Then compute

$$A_{k+1} = g_1^{f(\gamma)/(\gamma+x)} = g_1^{\bar{f}(\gamma)} (g_1^{1/(\gamma+x)})^{\bar{c}} = [\prod_{i=0}^{k-1} (g_1^{\gamma^i})^{\bar{f}_i}] (g_1^{1/(\gamma+x)})^{\bar{c}}$$

and we solve k-CAA Problem with (A_{k+1}, x). □

B.2.2 Proof of Lemma 5 Note $\mathbf{M}$ equals the following matrix:

$$\mathbf{M}(k, e_1, \cdots, e_k) = \begin{bmatrix} \mathsf{S}_{k-1}(E \setminus \{e_1\}) & \mathsf{S}_{k-2}(E \setminus \{e_1\}) & \cdots & \mathsf{S}_0(E \setminus \{e_1\}) \\ \mathsf{S}_{k-1}(E \setminus \{e_2\}) & \mathsf{S}_{k-2}(E \setminus \{e_2\}) & \cdots & \mathsf{S}_0(E \setminus \{e_2\}) \\ & \vdots & & \\ \mathsf{S}_{k-1}(E \setminus \{e_k\}) & \mathsf{S}_{k-2}(E \setminus \{e_k\}) & \cdots & \mathsf{S}_0(E \setminus \{e_k\}) \end{bmatrix}$$

By convention $\mathsf{S}_0 = 1$. We prove the the following statement:

$$\det(\mathbf{M}(k, e_1, \cdots, e_k)) = (\prod_{i=2}^{k} (e_1 - e_i)) \det(\mathbf{M}(k-1, e_2, \cdots, e_k)) \tag{11}$$

Then induction on k yields the Lemma.

Let matrix

$$\mathbf{U} = \begin{bmatrix} 1 & -1 & -1 & \cdots & -1 \\ & 1 & & & \\ & & \ddots & & 0 \\ & & & \ddots & \\ 0 & & & & \\ & & & & 1 \end{bmatrix}$$

Multiplying two matrices we obtain $\mathbf{M}(k, e_1, \cdots, e_k)\mathbf{U} =$

$$\begin{bmatrix} \mathsf{S}_{k-1}(E \setminus \{e_1\}) & \mathsf{S}_{k-2}(E \setminus \{e_1\}) & \cdots & \mathsf{S}_1(E \setminus \{e_1\}) & \mathsf{S}_0(E \setminus \{e_1\}) \\ (e_1 - e_2)\mathsf{S}_{k-2}(E \setminus \{e_1, e_2\}) & (e_1 - e_2)\mathsf{S}_{k-3}(E \setminus \{e_1, e_2\}) & \cdots & (e_1 - e_2)\mathsf{S}_0(E \setminus \{e_1, e_2\}) & 0 \\ (e_1 - e_3)\mathsf{S}_{k-2}(E \setminus \{e_1, e_3\}) & (e_1 - e_3)\mathsf{S}_{k-3}(E \setminus \{e_1, e_3\}) & \cdots & (e_1 - e_3)\mathsf{S}_0(E \setminus \{e_1, e_3\}) & 0 \\ & \vdots & & & \\ (e_1 - e_k)\mathsf{S}_{k-2}(E \setminus \{e_1, e_k\}) & (e_1 - e_k)\mathsf{S}_{k-3}(E \setminus \{e_1, e_k\}) & \cdots & (e_1 - e_k)\mathsf{S}_0(E \setminus \{e_1, e_k\}) & 0 \end{bmatrix}$$

Consider the lower left $(k-1) \times (k-1)$ matrix. Its i-th row is exactly the i-th row of $\mathbf{M}(k-1, E \setminus \{e_1\})$ multiplied by $e_1 - e_i$, . This proves Equation (11) and thus the Lemma. □

B.2.3 Proof Sketch of Theorem 4 Assume $\log_v(u) = \alpha$. The proof is similar to that of Theorem 3. We describe mainly the difference below. Given a PPT algorithm $\mathcal{A}$ which solves k-CAA2, and a k-SDH' Problem instance, randomly generate distinct nonzero e_i and x_i, $1 \leq i \leq k$. Let $f(\gamma)$, $f^{[i]}(\gamma)$ be as defined in the proof of Theorem 3. Then

$$A_j = v^{(1-x_i\alpha)/(\gamma+e_i)} = g_1^{(1-x_i\alpha)f^{[i]}(\gamma)}$$

Invoking $\mathcal{A}$ to obtain $(A_{k+1}, e_{k+1}, x_{k+1})$ satisfying $A_{k+1}^{\gamma+e_{k+1}} u^{x_{k+1}} = v$. The rest is similar to the proof of Theorem 3.

Given a PPT algorithm $\mathcal{A}$ which solves the k-SDH' Problem and a k-CCA2 Problem instance, we have $A_i^{\gamma+e_i} = v^{1-x_i\alpha}$. Let $g_1 = v^{1/f(\gamma)}$, then Equation (9) becomes

$$A_j = g_1^{(1-x_i\alpha)f^{[j]}(\gamma)}, 1 \leq j \leq k.$$

The non-singularity of the matrix $\bar{\mathbf{M}}$ ensures that a k-SDH' Problem instance can be computed from the A_j's. Invoke $\mathcal{A}$ to solve this problem instance, and then convert its answer to an answer for the k-CAA2 Problem is straightforward. □