Unlinkable Randomizable Signature and Its Application in Group Signature

Sujing Zhou, Dongdai Lin
SKLOIS Lab, Institute of Software,
Graduate School of the Chinese Academy of Sciences, 100080, Beijing , P.R. China.
Email: {zhousujing,ddlin}@is.iscas.ac.cn

Abstract

Recently, a kind of randomizable signatures have been adopted to construct group signatures, which might result in more efficient group signatures because only part of the randomized member certificate has to be concealed.

In this paper, we formalize the characteristics of randomizable signatures that are required to build secure efficient group signatures. Specifically, we define unlinkable randomizable signature, indirectly signable signature, Σ-protocol friendly signature. Designing efficient secure group signatures can be boiled down to designing Σ-protocol friendly unlinkable randomizable signature with indirect signability.

We found that almost all secure efficient group signatures known so far are actually in this line of utilizing unlinkable randomizable signatures. We proposed the unlinkable randomizable version of a group signature based on q-SDH assumption for the first time.

We proposed two new unlinkable randomizable signatures called Wat05+, CL04* which will result in new efficient group signatures.

We also improved a group signature by replacing the member certificate generation signature with an unlinkable randomizable signature.

Keywords: Digital Signature, Group Signature, Unlinkable Randomizable Signature, Sigma-protocol.

1 Introduction

The proposal of group signatures [CvH91] are motivated by enabling members of a group to sign on behalf of the group without leaking their true individual identities; but the signer's identity is able to be traced, i.e., discovered by the group manager (GM) on disputes. The counterpart of a group signature in the real world is official seal, at the sight of which anyone is assured of its origin from the claimed authority, but has no idea of who that particular person is behind the seal.

In brief, a group signature scheme is composed of the following steps: (1) GM, the group manager, along with some third trusted party, chooses the security parameters as well as a group secret key and a group public key. (2) Any group member candidate is required to choose his *member secret key*, and run an interactive protocol with GM to join in the group, during which GM generates a signature on the member secret key blindly, i.e., without knowing the secret key value, the signature is also called *member certificate*. (3) Any group member can generate group signatures using his member secret key and member certificate, called *group signing key* all together.

A common paradigm of constructing group signatures is widely used so far [CS97, CM98b, CM98a, ACJT00]: GM uses an ordinary signature to generate membership certificate for group members, i.e. sign on some secret key known only to members. The group signature is in fact a non-interactive zero-knowledge proof of knowledge of member certificate and member secret key, transformed in Fiat-Shamir's heuristic method [FS87] from interactive proofs.

Recently, a kind of randomizable signatures have been adopted in some schemes [CL04, CL02, BBS04, ACdMH06] to generate membership certificates. The following construction of group signature has been widely recognized: to sign on a message, a member firstly randomizes its member certificate, then generates a proof of knowledge of member secret key with respect to part of the randomized member certificate. This method might result in more efficient group signature because part of the randomized member certificate is revealed instead of having to be concealed totally in previous constructions.

In this paper, we formalize the characteristics of randomizable signatures that are required to build secure efficient group signatures. Specifically, we define unlinkable randomizable signature, indirectly signable signature, Σ-protocol friendly signature. Designing efficient secure group signatures can be boiled down to designing Σ-protocol friendly unlinkable randomizable signature with indirect signability.

We found that almost all secure efficient group signatures known so far (except [BW07]) are actually in this line of utilizing unlinkable randomizable signatures. The scheme in [CL02] can be seen as the unlinkable randomizable signature version of the well known ACJT scheme [ACJT00]. The counterpart of the scheme [NSN04] in this line is also proposed for the first time.

We proposed two new unlinkable randomizable signatures called Wat05+, CL04* which will result in new efficient group signatures.

We also improved the scheme in [KY05] by replacing the member certificate generation signature with an unlinkable randomizable signature.

2 Preliminary

Notations. $x \stackrel{\$}{\leftarrow} X$ denotes x is chosen uniformly at random from the set X. $x \stackrel{\$}{\leftarrow} A(.,.,.)$ denotes x is generated from executing algorithm A where random variables are chosen uniformly at random.

G^k, $(Z_p^*)^k$ denote a k tuple from $\mathbb{G}$ and Z_p^* respectively. 0^k (1^k) denotes the string of k zeros (ones). $|M|$ denotes the binary length of M.

If (P,V) is a non-interactive proof for relation ρ, $P(x,w,R)$ denotes the operation of generating a proof for $(x,w) \in \rho$ under the common reference string R, $V(x,\pi,R)$ denotes the operation of verifying a proof π.

$\epsilon(k)$ is a *negligible function* if $\epsilon(k) \leq 1/P(k)$ for any polynomial $P(k)$ and all sufficiently large k.

Definition 1 (wUF-ATK[DK01]). *A signature scheme DS=(Gen, Sig, Ver) is wUF-ATK secure (ATK $\in$ $\{CMA, ACMA\}$), i.e., weakly unforgeable against ATK attack, if for every probabilistic polynomial-time algorithm $\mathcal{A}$, it holds that*

$$\mathsf{Adv}_{DS,\mathcal{A}}^{wUF-ATK} = \mathsf{Pr}\{Ver(pk,m,\sigma)=1, m \neq m_i, (m,\sigma) \leftarrow \mathcal{A}(m_i,\sigma_i,pk),$$
$$(pk,sk) \leftarrow Gen(1^k), (m_i,\sigma_i) \leftarrow Q_{\mathcal{A}}^{Sig(sk,.)}, i=1,...,q_{sig}\} < \epsilon(k)$$

where $\epsilon(k)$ is a negligible function, the probability is taken over the coin tosses of algorithms Gen, Sig and $\mathcal{A}$. $Q_{\mathcal{A}}^{Sig(sk,.)}$ denotes the finite set of queries to oracle $Sig(sk,.)$ made by $\mathcal{A}$.

3 Unlinkable Randomizable (UR) Signature

Definition 2 (Randomizable Signature). *A randomizable signature scheme is a digital signature scheme that has an efficient signature randomization algorithm Rnd besides algorithms (Gen,Sig,Ver):*

- *Gen: $N \rightarrow K$: a probabilistic polynomial-time algorithm with input k (called security parameter), output $(pk,sk) \in K$, where K is a finite set of possible keys; pk is called public key, sk is secret key kept to the signer, i.e., the owner of the instance of the signature scheme.*

- *Sig: $K \times M \rightarrow S$: a probabilistic polynomial-time algorithm with input (sk,m), where sk is the same output from K above, $m \in M$, M is a finite set of possible messages. Output is $\sigma = (\Upsilon, \Xi) \in S$, where Υ is randomly chosen and independent from m, Ξ is calculated from Υ, m and sk.*

- *Ver: $K \times M \times S \rightarrow \{0,1\}$: a deterministic polynomial-time algorithm with input (pk,m,σ), output 1 if σ is valid, i.e., σ is really computed by the owner of the signature instance, output 0 otherwise.*

- *Rnd:* $M \times S \to S$*: a probabilistic polynomial-time algorithm with a message* m *and a signature* (Υ, Ξ) *on it, output a* $(\Upsilon', \Xi') \neq (\Upsilon, \Xi)$ *that is also a signature on* m*.*

The following concept of *indirectly signable* is actually a restatement of signatures on committed message [CL04].

Definition 3 (Indirectly Signable)**.** *A signature is indirectly signable if there exists a one way function* f *(as defined in Chapter 9.2.4, [MvOV96] or more technically as in Chapter 2.2, [Gol01]) and an efficient algorithm* Sig_f *that* $Sig(sk, m) = Sig_f(sk, f(m))$*. That is*

$$\mathsf{Pr}\{(pk, sk, f) \xleftarrow{\$} Gen(1^k), m \xleftarrow{\$} M, v \leftarrow f(m), \sigma \leftarrow Sig_f(sk, v) : Ver(pk, m, \sigma) = 1\} = 1,$$

and for any probabilistic polynomial time algorithm $\mathcal{A}$*,*

$$\mathsf{Pr}\{(pk, sk, f) \xleftarrow{\$} Gen(1^k), m \xleftarrow{\$} M, v \leftarrow f(m), m' \leftarrow \mathcal{A}(sk, v) : m' = m\} < \epsilon(k).$$

Define experiment $\mathsf{Exp}_{\mathcal{A}}^{unlink-b}(k)$, $b \in \{0, 1\}$ as follows:

$\mathsf{Exp}_{\mathcal{A}}^{unlink-b}(k)$:
$(pk, sk) \xleftarrow{\$}$ Gen (1^k),
$(\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)$,
If Ver$(pk, m_0, \langle\Upsilon_0, \Xi_0\rangle) = 0$ or Ver$(pk, m_1, \langle\Upsilon_1, \Xi_1\rangle) = 0$, return 0.
$(\Upsilon', \Xi') \xleftarrow{\$}Rnd(m_b, \Upsilon_b, \Xi_b)$,
$b' \leftarrow \mathcal{A}(sk, \Xi')$.
return b'.

Definition 4 (Perfectly Unlinkable)**.** *A randomizable signature rDS=(Gen,Sig,Ver,Rnd) is perfectly unlinkable if for any algorithm* $\mathcal{A}$*, the distribution of output of* $\mathsf{Exp}_{\mathcal{A}}^{unlink-b}(k)$ *are the same for* $b \in \{0, 1\}$*, that is*

$$\mathsf{Pr}\{\mathsf{Exp}_{\mathcal{A}}^{unlink-1}(k) = 1\} = \mathsf{Pr}\{\mathsf{Exp}_{\mathcal{A}}^{unlink-0}(k) = 1\},$$

which is identical to

$$\begin{aligned}\mathsf{Pr}\{\Xi' \xleftarrow{\$} Rnd(m_1, \Upsilon_1, \Xi_1) | (pk, sk) \xleftarrow{\$} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\} \\ = \mathsf{Pr}\{\Xi' \xleftarrow{\$} Rnd(m_0, \Upsilon_0, \Xi_0) | (pk, sk) \xleftarrow{\$} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\}.\end{aligned}$$

Definition 5 (Statistically Unlinkable)**.** *A randomizable signature rDS=(Gen,Sig,Ver,Rnd) is statistically unlinkable if for any algorithm* $\mathcal{A}$*, the statistical distance between output of* $\mathsf{Exp}_{\mathcal{A}}^{unlink-b}(k)$ *for* $b \in \{0, 1\}$ *is negligible, that is*

$$\sum |\mathsf{Pr}\{\mathsf{Exp}_{\mathcal{A}}^{unlink-1}(k) = 1\} - \mathsf{Pr}\{\mathsf{Exp}_{\mathcal{A}}^{unlink-0}(k) = 1\}| < \epsilon(k),$$

where the sum is over all random choices of Gen, $\mathcal{A}$ *and Rnd.*

Definition 6 (Computationally Unlinkable)**.** *A randomizable signature rDS=(Gen,Sig,Ver,Rnd) is computationally unlinkable if for any probabilistic polynomial time algorithm* $\mathcal{A}$*, the probability between output of* $\mathsf{Exp}_{\mathcal{A}}^{unlink-b}(k)$ *for* $b \in \{0, 1\}$ *is negligible, that is*

$$\mathsf{Pr}\{\mathsf{Exp}_{\mathcal{A}}^{unlink-1}(k) = 1\} - \mathsf{Pr}\{\mathsf{Exp}_{\mathcal{A}}^{unlink-0}(k) = 1\} < \epsilon(k)$$

The above definitions of unlinkability can be further weakened by not allowing the adversary obtain the secret key, but granting access to signing oracle $\mathcal{O}_{sig}(sk, \cdot)$. That is, the corresponding experiment $\mathsf{Exp}_{\mathcal{A}}^{w-unlink-b}(k)$, $b \in \{0, 1\}$ is defined as follows:

$\mathsf{Exp}_{\mathcal{A}}^{w-unlink-b}(k)$:
$(pk, sk) \xleftarrow{\$}$ Gen (1^k),
$(\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{sig}(sk,\cdot)}(pk)$,
If Ver$(pk, m_0, \langle\Upsilon_0, \Xi_0\rangle) = 0$ or Ver$(pk, m_1, \langle\Upsilon_1, \Xi_1\rangle) = 0$, return 0.
$(\Upsilon', \Xi') \xleftarrow{\$}Rnd(m_b, \Upsilon_b, \Xi_b)$,
$b' \leftarrow \mathcal{A}(pk, \Xi')$.
return b'.

Then we get weak perfectly unlinkability, weak statistically unlinkability, weak computationally unlinkability respectively.

Definition 7 (Unlinkable Randomizable Signature)**.** *A (perfectly, statistically, computationally) UR signature urDS=(Gen,Sig, Ver,Rnd) is a randomizable signature that is also (perfectly, statistically, computationally) unlinkable respectively.*

Notice that the unlinkability defined as above does not mean that the whole randomized signature is unlinkable with the whole signature before randomization. An adversary of unlinkability of a signature is only given part of the randomized signature Ξ.

Take Scheme A in [CL04] as an example. Let $\mathbb{G}_1 = \langle g\rangle$, $\mathbb{G}_2 = \langle\tilde{g}\rangle$, $\mathbb{G}_3$ be p order cyclic groups, and there exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$.

- Gen. Choose $x \xleftarrow{\$} Z_p^*$ and $y \xleftarrow{\$} Z_p^*$, and sets $sk = (x, y)$, $pk = (p, g, \tilde{g}, \mathbb{G}_1, \mathbb{G}_2, e, X, Y)$, where $X = \tilde{g}^x$ and $Y = \tilde{g}^y$.
- Sig. On input message m, secret key $sk = (x, y)$, and public key pk, choose a random $a \in \mathbb{G}_1$, output the signature $\sigma = (a, a^y, a^{x+mxy})$.
- Ver. On input pk, message m, and purported signature (a, b, c), check that the following verification equations hold: $e(a, Y) = e(b, \tilde{g})$ and $e(a, X)e(b, X)^m = e(c, \tilde{g})$.
- Rnd. On input pk, message m, and a signature (a, b, c), choose random $r \in Z_p^*$, output $\sigma' = (a', b', c') = (a^r, b^r, c^r)$.

If we set $\Upsilon = ()$, $\Xi = (a, b, c)$, it is not even computationally unlinkable, because any one can check if (m_1, a', b', c') or (m_0, a', b', c') is a valid signature. That is why group signatures adopting the above signature only result in selfless anonymity (a weaker anonymity where the adversary should not know the message m)[ACdMH06].

If we set $\Upsilon = (a)$, $\Xi = (b, c)$, then it is still not even computationally unlinkable, but is weak computationally unlinkable assuming DDH is hard over group $\mathbb{G}_1$.

If we further set $\Upsilon = (a, b)$, $\Xi = (c)$, then it is perfectly unlinkable. So it is rather easy to come up with an unlinkable randomizable signature, just reveal the randomized signature as less as possible. But revealing too little of the randomized signature may cause to loose another important characteristic: Σ-protocol friendly.

Definition 8 (Σ -protocol Friendly Randomizable Signature)**.** *A randomizable signature rDS=(Gen,Sig, Ver,Rnd) is Σ -protocol friendly if there exits a Σ -protocol $\mathcal{P}$ for relation $\mathcal{R} = \{(\Xi, \langle\Upsilon,\ m\rangle)|Ver(pk,\ m,\ \langle\Upsilon,\ \Xi\rangle) = 1\}$, that is [Dam05]*

- *$\mathcal{P}$ is of 3-move form, and if Prover and Verifier follow the protocol, Verifier always accepts.*
- *From any Ξ and any pair of accepting conversations with different initial message from Prover on input the same Ξ, one can efficiently compute (Υ, m) such that $(\Xi, \langle\Upsilon, m\rangle) \in \mathcal{R}$.*
- *There exists a polynomial time simulator M, which on input Ξ, and a random second message sent from Verifier, outputs an accepting conversation with the same probability distribution as between the honest Prover, Verifier on input Ξ.*

Actually signatures with above characteristics have been proposed and adopted explicitly or implicitly [CL02, CL04, BBS04, ACdMH06], see the following table.

[ACJT00]	[CL02]
Let $n = pq$ be a RSA modulus. $S_e = [2^{l_e} - 2^{\mu_e}, 2^{l_e} + 2^{\mu_e}]$, $S_m = [2^{l_m} - 2^{\mu_m}, 2^{l_m} + 2^{\mu_m}]$, $S_s = [2^{l_s} - 2^{\mu_s}, 2^{l_s} + 2^{\mu_s}]$, $\mu_e > l_m$.	
Gen. $a, c \xleftarrow{\$} QR_n^*$, $sk = (p, q)$, $pk = (n, a, c, S_e, S_m)$. *Sig.* If $\|m\| = l_m$, $e \xleftarrow{\$} S_e \cap \mathsf{P}rime$, $A \leftarrow (a^m c)^{\frac{1}{e}} \bmod n$. *Ver.* Given m, (e, A), check if $\|m\| = l_m$, $A^e = a^m c \bmod n$. *Rnd.* -	*Gen.* $a, b, c \xleftarrow{\$} QR_n^*$, $sk = (p, q)$, $pk = (n, a, b, c, S_e, S_m, S_s)$. *Sig.* If $\|m\| = l_m$, $e \xleftarrow{\$} S_e \cap \mathsf{P}rime$, $s \xleftarrow{\$} S_s$, $A \leftarrow (a^m b^s c)^{\frac{1}{e}} \bmod n$. $\Upsilon = (e, s)$, $\Xi = (A)$ *Ver.* Given m, $(\Upsilon, \Xi) = (e, s, A)$, check if $\|m\| = l_m$, $\|s\| = l_s$, $A^e = a^m b^s c \bmod n$. *Rnd.* Given m, $(\Upsilon, \Xi) = (e, s, A)$, choose random r with length $l_r = l_s - l_e - 1$, $\Upsilon' = (e, s + re)$, $\Xi' = (Ab^r)$.
CL04	CL04+.
Let $\mathbb{G}_1 = \langle g \rangle$, $\mathbb{G}_2 = \langle \tilde{g} \rangle$ be p order cyclic groups that there exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$.	
Gen. $x, y \xleftarrow{\$} Z_p^*$, $sk = (x, y)$, $X = \tilde{g}^x$, $Y = \tilde{g}^y$,$pk = (p, g, \tilde{g}, \mathbb{G}_1, \mathbb{G}_2, e, X, Y)$.	
Sig. $d \xleftarrow{\$} \mathbb{G}_1$, $s \xleftarrow{\$} Z_p^*$, $\Upsilon = ()$, $\Xi = (d, d^y, d^{x+mxy})$. *Ver.* Given m, $(\Upsilon, \Xi) = (a, b, c)$, check if $e(a, Y) = e(b, \tilde{g})$, $e(a, X)e(b, X)^m = e(c, \tilde{g})$. *Rnd.* Given m, $(\Upsilon, \Xi) = (a, b, c)$. $r \xleftarrow{\$} Z_p^*$, $\Upsilon' = ()$, $\Xi' = (a', b', c') = (a^r, b^r, c^r)$.	*Sig.* $d \xleftarrow{\$} \mathbb{G}_1$, $s \xleftarrow{\$} Z_p^*$, $\Upsilon = (s)$, $\Xi = (d^s, d^{sy}, d^{x+mxy})$. *Ver.* Given m, $(\Upsilon, \Xi) = (s, a, b, c)$, check if $e(a, Y) = e(b, \tilde{g})$, $e(a, X)e(b, X)^m = e(c, \tilde{g})^s$. *Rnd.* Given m, $(\Upsilon, \Xi) = (s, a, b, c)$, $r_1, r_2 \xleftarrow{\$} Z_p^*$, $\Upsilon' = (s') = (r_2 s)$, $\Xi' = (a', b', c') = (a^{r_1 r_2}, b^{r_1 r_2}, c^{r_1})$.
BBS04	BBS04+[a]
Gen. $x \xleftarrow{\$} Z_p^*$, $w = \tilde{g}^x$, $h_1 \xleftarrow{\$} \mathbb{G}_1$. $sk = (x)$, $pk = (p, \mathbb{G}_1, \mathbb{G}_2, g, \tilde{g}, h_1, e, w)$.	
Sig. $s \xleftarrow{\$} Z_p^*$, $A \leftarrow (h_1^m g)^{\frac{1}{x+s}}$. *Ver.* Given m, (s, A), check if $e(A, w\tilde{g}^s) = e(h_1^m g, \tilde{g})$. *Rnd.* -	*Sig.* $s, t \xleftarrow{\$} Z_p^*$, $A \leftarrow (h_1^m g)^{\frac{t}{x+s}}$, $\Upsilon = (s, t)$, $\Xi = (A)$. *Ver.* Given m, $(\Upsilon, \Xi) = (s, t, A)$, check if $e(A, w\tilde{g}^s) = e(h_1^m g, \tilde{g}^t)$. *Rnd.* Given m, $(\Upsilon, \Xi) = (s, t, A)$, $r \xleftarrow{\$} Z_p^*$, $\Upsilon' = (s, rt)$, $\Xi' = (A^r)$.

[a] In the end of introduction part of [BBS04], a new group signature using this underlying signature following the method of [CL04] was implicitly and briefly mentioned ("Their methodology can also be applied to the SDH assumption, yielding a different SDH-based group signature.").

The scheme in [CL02] (noted CL02) can be viewed as the unlinkable randomizable signature version of the signature underlying the well known ACJT scheme [ACJT00], or the signature in [Fis03].

CL02 has been proved ACMA secure (in standard model [CL02]). Obviously it is indirectly signable if we define $f(m) = a^m$ assuming Computational Diffie-Hellman problem on QR_n^* is hard.

CL02 is Σ-protocol friendly, because there exists an efficient Σ-protocol for the relation $\{(m, e, s) \| A^e = a^m b^s c \bmod n, m \in S_m,\ s \in S_s\}$ using the standard of proof of knowledge of discrete representations [Gir91, CP93, FO97, CM99, Bou00, KTY04].

Intuitively it is also perfectly unlinkable because each randomized Ξ' only consists of one element that is generated independently and randomly each time.

Lemma 3.1. *CL02 signature is perfectly unlinkable.*

Proof.

$$\Pr\{\Xi' \xleftarrow{\$} Rnd(m_1, \Upsilon_1, \Xi_1) | (pk, sk) \xleftarrow{\$} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\}$$
$$= \Pr\{v_1^r | r \xleftarrow{\$} \{0,1\}^{l_r}\}$$
$$= \Pr\{v_1^r | r \leftarrow \tilde{r} + \log_b (v_0 v_1^{-1}), \tilde{r} \xleftarrow{\$} \{0,1\}^{l_r}\}$$
$$= \Pr\{v_0^{\tilde{r}} | \tilde{r} \xleftarrow{\$} \{0,1\}^{l_r}\}$$
$$= \Pr\{\Xi' \xleftarrow{\$} Rnd(m_0, \Upsilon_0, \Xi_0) | (pk, sk) \xleftarrow{\$} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\}$$

□

The scheme CL04+ (implied in [CL04]) is wUF-ACMA which can be proved similarly as Scheme A in [CL04]. Briefly, suppose $\mathcal{A}$ is an adversary of CL04+, then an adversary $\mathcal{B}$ of LRSW assumption (defined in [CL04]) is available: when $\mathcal{A}$ queries signature on m, $\mathcal{B}$ transfers the query to LRSW oracle $O_{x,y}$ (defined in [CL04]); $\mathcal{B}$ will get a response from LRSW oracle $O_{x,y}$, i.e., (a,b,c)=(a, a^y, a^{x+mxy}), then $\mathcal{B}$ sends (s, a^s, b^s, c) ($s \xleftarrow{\$} Z_p^*$) to $\mathcal{A}$. If $\mathcal{A}$ outputs a signature (s^*, a^*, b^*, c^*) on a message m^* that it has never queried, then $(m^*, {a^*}^{\frac{1}{s^*}}, {b^*}^{\frac{1}{s^*}}, c^*)$ is a resolution to LRSW assumption, $\mathcal{B}$ wins.

CL04+ is Σ-protocol friendly, because there exists an efficient Σ-protocol for the relation $\{(m,s) | e(a,X)\ e(b,X)^m = e(c,\tilde{g})^s\}$.

CL04+ is indirectly signable if define $f(m) = g^m$ assuming Computational Diffie-Hellman problem on $\mathbb{G}_1$ is hard.

Lemma 3.2. *CL04+ signature is perfectly unlinkable.*

Proof.

$$\Pr\{\Xi' \xleftarrow{\$} Rnd(m_1, \Upsilon_1, \Xi_1) | (pk, sk) \xleftarrow{\$} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\}$$
$$= \Pr\{(a_1^{r_1 r_2}, b_1^{r_1 r_2}, c_1^{r_1}) | r_1 \xleftarrow{\$} Z_p^*, r_2 \xleftarrow{\$} Z_p^*\}$$
$$= \Pr\{(a_1^{r_1 r_2}, b_1^{r_1 r_2}, c_1^{r_1}) | r_1 \leftarrow \tilde{r}_1 \alpha, r_2 \leftarrow \tilde{r}_2 \beta, \tilde{r}_1 \xleftarrow{\$} Z_p^*, \tilde{r}_2 \xleftarrow{\$} Z_p^*\}$$
$$(\text{ Note } \alpha = (\log_{a_1} a_0)\frac{s_1(x+m_0xy)}{s_0(x+m_1xy)}, \beta = \frac{s_0(x+m_1xy)}{s_1(x+m_0xy)})$$
$$= \Pr\{(a_0^{\tilde{r}_1 \tilde{r}_2}, b_0^{\tilde{r}_1 \tilde{r}_2}, c_0^{\tilde{r}_1}) | \tilde{r}_1 \xleftarrow{\$} Z_p^*, \tilde{r}_2 \xleftarrow{\$} Z_p^*\}$$
$$= \Pr\{\Xi' \xleftarrow{\$} Rnd(m_0, \Upsilon_0, \Xi_0) | (pk, sk) \xleftarrow{\$} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\}$$

□

BBS04+, implicitly mentioned in [BBS04], can be proved wUF-CMA similarly to the scheme [BBS04]. Briefly, suppose $\mathcal{A}$ is an adversary of BBS04+, then an adversary $\mathcal{B}$ of [BBS04] is available: when $\mathcal{A}$ queries signature on m, $\mathcal{B}$ transfers the query to signature oracle of [BBS04]; $\mathcal{B}$ will get a response from the signature oracle, i.e., (s, A), where $e(A, w\tilde{g}^s) = e(h_1^m g, \tilde{g})$, then $\mathcal{B}$ chooses $t \xleftarrow{\$} Z_p^*$, sends (s, t, A^t) to $\mathcal{A}$. If $\mathcal{A}$ outputs a signature (s^*, t^*, A^*) on a message m^* that it has never queried, then $(s^*, {A^*}^{\frac{1}{t^*}})$ is a valid BBS04+ signature on m^*, which $\mathcal{B}$ has never queried.

BBS04+ is indirectly signable if we define $f(m) = h_1^m$ assuming Computational Diffie-Hellman problem on $\mathbb{G}_1$ is hard. Obviously, BBS04+ is perfectly unlinkable because each randomized Ξ' only consists of one element that is generated independently and randomly each time.

BBS04+ is Σ-protocol friendly, because there exists an efficient Σ-protocol for the relation $\{(m,s,t) | e(A,w)\ e(A,\tilde{g})^s = e(g,\tilde{g})^t\}\ e(h_1,\tilde{g})^{mt}$.

3.1 NSN04*

The two most efficient group signatures are based on strong RSA assumption and q-SDH assumption respectively [ACJT00, NSN04]. As we have mentioned, the scheme of [ACJT00] has a counterpart scheme utilizing unlinkable randomizable signature CL02. As for the scheme in [NSN04], no similar counterpart has been proposed. In this section, we propose a new unlinkable randomizable signature NSN04*, which can be adopted to build a new efficient group signature.

[NSN04]	**NSN04*.**
Let $\mathbb{G}$ be a p order additive cyclic group, and there exists a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}'$. $\mathbb{G} = \langle P \rangle$.	
Gen. $\gamma \xleftarrow{\$} Z_p^*$, $P_{pub} = \gamma P$, $P_0 \xleftarrow{\$} \mathbb{G}$, $sk = (\gamma)$, $pk = (p, \mathbb{G}, \mathbb{G}', P, P_0, P_{pub}, e)$.	
Sig. $a \xleftarrow{\$} Z_p^*$, $A = \frac{1}{\gamma+a}[mP + P_0]$.	*Sig.* $(a, b, c) \xleftarrow{\$} Z_p^{*3}$, $A = \frac{1}{\gamma+a}[mP + (b + \gamma c)P_{pub} + P_0]$, $\Upsilon = (a, b, c)$, $\Xi = (A)$.
Ver. Given m, (a, A), check if $e(A, P_{pub} + aP) = e(mP + P_0, P)$.	*Ver.* Given m, $(\Upsilon, \Xi) = (a, b, c, A)$, check if $e(A, P_{pub} + aP) = e(mP + bP_{pub} + P_0, P)e(cP_{pub}, P_{pub})$.
Rnd. -	*Rnd.* Given m, $(\Upsilon, \Xi) = (a, b, c, A)$, $r \xleftarrow{\$} Z_p^*$, $\Upsilon' = (a', b', c') = (a, b + ra, c + r)$, $\Xi' = (A') = (A + rP_{pub})$.

The security if NSN04* is guaranteed by the following Lemma.

Lemma 3.3. *NSN04* is wUF-ACMA if q-SDH problem in $\mathbb{G}$ is hard, where q is polynomial in $|p|$. See Appendix A for the proof.*

NSN04* is indirectly signable if we define $f(m) = mP$ assuming Computational Diffie-Hellman problem on $\mathbb{G}$ is hard. Obviously, NSN04* is perfectly unlinkable because each randomized Ξ' only consists of one element that is generated independently and randomly each time.

NSN04* is Σ-protocol friendly, because there exists an efficient Σ-protocol for the relation $\{(m, a, b, c)| \; e(A, P_{pub})\; e(A, P)^a = e(P, P)^m \; e(P_{pub}, P)^b \; e(P_0, P) \; e(P_{pub}, P_{pub})^c\}$.

3.2 Wat05+

The recently proposed signature in [Wat05], which is provable secure under CBDH assumption (Computational Bilinear Diffie-Hellman assumption) without random oracle, is also an UR signature if only we change a bit on it, see the following restatement with an extra algorithm *Rnd*.

Wat05+.	Let $\mathbb{G}$, $\mathbb{G}'$ be two p order cyclic groups, and there exists a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}'$. $\mathbb{G} = \langle g \rangle$.
Gen.	Set secret key $sk = (x)$, $pk = (e, g_1, g_2, u', u_i, i = 0, .., l)$, where g_1, g_2, u', u_i are all elements from $\mathbb{G}$, $g_1 = g^x$, l is the maximum binary length of a message to be signed.
Sig.	Given a message m with length at most l, the signature (Υ, Ξ) is $\Upsilon = (s)$, $\Xi = (a, b) = (g^r, g_2^{sx}(u' \prod_{i=1}^{l} u_i^{m_i})^r)$, where $s \xleftarrow{\$} Z_p^*$. Note that $(a^{1/s}, b^{1/s})$ is a signature of m according to Scheme [Wat05].
Ver.	Given a message m and its signature $(\Upsilon, \Xi) = (s, a, b)$, it is a valid signature on m if $e(b, g) = e(u', a)e(g_2, g_1)^s \prod_{i=1}^{l} e(u_i, a)^{m_i}$.
Rnd.	On input pk, message m, and a signature (Υ, Ξ), where $\Upsilon = (s)$, $\Xi = (a, b)$, choose $(r_1, r_2) \xleftarrow{\$} Z_p^* \times Z_p^*$, set $\Upsilon' = (s') = (sr_1)$, $\Xi' = (a', b') = (a^{r_1} g^{r_2}, b^{r_1}(u' \prod_{i=1}^{l} u_i^{m_i})^{r_2})$. The new randomized signature on m is (Υ', Ξ').

Wat05+ is wUF-ACMA. Briefly, Suppose $\mathcal{A}$ is an adversary of Wat05+, then an adversary $\mathcal{B}$ of Wat05 is available: when $\mathcal{A}$ queries signature on m, $\mathcal{B}$ transfers the query to signature oracle of Wat05 obtaining $(a, b) = (g^r, g_2^x(u' \prod_{m_i=1}^{l} u_i)^r)$, which is then modified by $\mathcal{B}$ into (s, a^s, b^s) where $s \xleftarrow{\$} Z_p^*$; the modification, now a valid Wat05+ signature on m, is sent to $\mathcal{A}$. If $\mathcal{A}$ outputs a signature (s^*, a^*, b^*) on a message m^* that it has never queried, then $(a^{*\frac{1}{s^*}}, b^{*\frac{1}{s^*}})$ is a valid Wat05 signature on m^* that $\mathcal{B}$ has not queried.

Wat05+ is Σ-protocol friendly, because there exits efficient Σ-protocol for the relation $\{(m_1, ..., m_l, s)| e(b, g) = e(u', a)\ e(g_2, g_1)^s \prod_{i=1}^{l} e(u_i, a)^{m_i}\}$.

Wat05+ is indirectly signable if we define $f(m) = \prod_{i=1}^{l} u_i^{m_i}$, it is one way if $l = O(k)$, where k is the security parameter. That is because the probability of $f(m) = f(m')$ for $m \neq m'$ is about $1/p$, i.e., the solution to $f(m) = c$ for a random $c \in \mathbb{G}$ is unique non-negligibly. To obtain the unique solution, 2^l tests must be carried out.

Lemma 3.4. *Wat05+ signature is perfectly unlinkable.*

Proof.

$$\begin{aligned}
&\mathsf{Pr}\{\Xi' \xleftarrow{\$} \mathit{Rnd}(m_1, \Upsilon_1, \Xi_1) | (pk, sk) \xleftarrow{\$} \mathit{Gen}(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\} \\
&= \mathsf{Pr}\{(a_1^{r_1} g^{r_2}, b_1^{r_1}(u' \prod_{j=1}^{l} u_j^{m_{1j}})^{r_2}) | r_1 \xleftarrow{\$} Z_p^*, r_2 \xleftarrow{\$} Z_p^*\} \\
&= \mathsf{Pr}\{(a_1^{r_1} g^{r_2}, b_1^{r_1}(u' \prod_{j=1}^{l} u_j^{m_{1j}})^{r_2}) | r_1 \leftarrow A\tilde{r}_1 + B\tilde{r}_2, r_2 \leftarrow C\tilde{r}_1 + D\tilde{r}_2, \tilde{r}_1 \xleftarrow{\$} Z_p^*, \tilde{r}_2 \xleftarrow{\$} Z_p^*\} \\
&\quad (\text{ Note } A = \frac{\log_g a_0 \log_{b_1} U_1 - \log_{b_1} b_0}{F}, B = \frac{\log_{b_1} U_1 U_0^{-1}}{F}, \\
&\quad C = \frac{\log_{b_1} b_0 \log_g a_1 - \log_g a_0}{F}, D = \frac{\log_g a_1 \log_{b_1} U_0 - 1}{F}, \\
&\quad F = \log_g a_1 \log_{b_1} U_1 - 1, U_1 = u' \prod_{j=1}^{l} u_j^{m_{1j}}, U_0 = u' \prod_{j=0}^{l} u_j^{m_{0j}}) \\
&= \mathsf{Pr}\{(a_0^{\tilde{r}_1} g^{\tilde{r}_2}, b_0^{\tilde{r}_1}(u' \prod_{j=1}^{l} u_j^{m_{0j}})^{\tilde{r}_2}) | \tilde{r}_1 \xleftarrow{\$} Z_p^*, \tilde{r}_2 \xleftarrow{\$} Z_p^*\} \\
&= \mathsf{Pr}\{\Xi' \xleftarrow{\$} \mathit{Rnd}(m_0, \Upsilon_0, \Xi_0) | (pk, sk) \xleftarrow{\$} \mathit{Gen}(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \xleftarrow{\$} \mathcal{A}(sk)\}
\end{aligned}$$

□

Note that the original scheme Wat05 [Wat05] is utilized in the compact group signature [BW06]. But Wat05+ has not been adopted anywhere.

3.3 CL04*

CL04* is a new UR signature similar to the standard signature proposed in [ZL06], both of them are wUF-ACMA under the following assumption.

Assumption 1 ([ZL06])**.** *Suppose $\mathbb{G}_1$, $\mathbb{G}_2$ be two p ordered cyclic group that exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, $\mathbb{G}_1 = \langle g \rangle$, $\mathbb{G}_2 = \langle \tilde{g} \rangle$. Let $X = g^x$, $Y = g^y$, $\widetilde{X} = \tilde{g}^x$, $\widetilde{Y} = \tilde{g}^y$, $O_{x,y}(.)$ be an oracle that, on input a value $m \in Z_p^*$, outputs a pair $(g^r, g^{r(x+my)+xy})$ for a randomly chosen $r \in Z_p^* \setminus \{1\}$. Then for any probabilistic polynomial time bounded adversary $\mathcal{A}$, the following probability is negligible:*

$$\begin{aligned}
\mathsf{Pr}\{(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, g, \tilde{g}) \leftarrow \mathit{Setup}(1^k); x \xleftarrow{R} Z_p^*; y \xleftarrow{R} Z_p^*; X = g^x; Y = g^y; \widetilde{X} = \tilde{g}^x; \widetilde{Y} = \tilde{g}^y; \\
(m, a, b) \leftarrow \mathcal{A}^{O_{x,y}}(p, g, \tilde{g}, e, X, Y, \widetilde{X}, \widetilde{Y}) : m \in Z_p^* \backslash Q \ \wedge \\
a = g^r \ \wedge a \notin \{1_{\mathbb{G}_1}, g\} \ \wedge \ b = g^{r(x+my)+xy}\} < \epsilon,
\end{aligned}$$

where Q is the set of queries that $\mathcal{A}$ has made to $O_{x,y}(.)$, $1_{\mathbb{G}_1}$ is the unit element of $\mathbb{G}_1$.

The difference between CL04* and the standard signature in [ZL06] mainly lies in the algorithm Rnd.

CL04*.	Let $\mathbb{G}_1$ be a p order cyclic group that exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$. $\mathbb{G}_1 = \langle g \rangle$, $\mathbb{G}_2 = \langle \tilde{g} \rangle$.
Gen.	Select $(x, y) \overset{R}{\leftarrow} Z_p^* \times Z_p^*$, set $X = g^x$, $Y = g^y$, $\widetilde{X} = \tilde{g}^x$, $\widetilde{Y} = \tilde{g}^y$. The secret key is $sk = (x, y)$, public key is $pk = (X, Y, \widetilde{X}, \widetilde{Y}, g, \tilde{g}, e, p)$.
Sig.	Given a message $m \in Z_p^*$, its signature is (Υ, Ξ), where $\Upsilon = (s)$, $\Xi = (a, b) = (g^r, g^{r(x+my)+sxy})$, $(r, s) \overset{\$}{\leftarrow} Z_p^* \times Z_p^*$.
Ver.	Given a signature (Υ, Ξ)=(s, a, b) of m, check if $e(b, \tilde{g}) = e(a, \widetilde{X}\widetilde{Y}^m)e(X, \widetilde{Y})^s$. If the equation holds, then accept (Υ, Ξ) as a valid signature of m, otherwise reject it as invalid.
Rnd.	On input pk, message m, and a signature $(\Upsilon, \Xi) = (s, a, b)$, choose random $r_1, r_2 \in Z_p^* \times Z_p^*$, output (Υ', Ξ') where $\Upsilon' = (s') = (r_1 s)$, $\Xi' = (a', b') = (a^{r_1} g^{r_2}, b^{r_1}(XY^m)^{r_2})$.

Lemma 3.5. *CL04* is wUF-ACMA secure under Assumption 1.*

Proof. Briefly, Suppose $\mathcal{A}$ is an adversary of CL04*, then an adversary $\mathcal{B}$ of Assumption 1 is available: when $\mathcal{A}$ queries signature on m, $\mathcal{B}$ transfers the query to oracle $O_{x,y}$ obtaining $(a, b) = (g^r, g^{r(x+my)+xy})$, which is then modified by $\mathcal{B}$ into (s, a^s, b^s) where $s \overset{\$}{\leftarrow} Z_p^*$; the modification, now a valid CL04* signature on m, is sent to $\mathcal{A}$. If $\mathcal{A}$ outputs a signature (s^*, a^*, b^*) on a message m^* that it has never queried, then $(a^{*\frac{1}{s^*}}, b^{*\frac{1}{s^*}})$ is a resolution to Assumption 1. □

CL04* is Σ-protocol friendly, because there exits efficient Σ-protocol for the relation $\{(m, s)|\ e(b, \tilde{g}) = e(a, \widetilde{X})\ e(a, \widetilde{Y})^m\ e(X, \widetilde{Y})^s\}$.

CL04* is indirectly signable if define $f(m) = g^m$ assuming Computational Diffie-Hellman problem on $\mathbb{G}_1$ is hard.

Lemma 3.6. *CL04* signature is perfectly unlinkable.*

Proof.

$$
\begin{aligned}
&\mathsf{Pr}\{\Xi' \overset{\$}{\leftarrow} Rnd(m_1, \Upsilon_1, \Xi_1)|(pk, sk) \overset{\$}{\leftarrow} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \overset{\$}{\leftarrow} \mathcal{A}(sk)\}\\
&= \mathsf{Pr}\{(a_1^{r_1} g^{r_2}, b_1^{r_1}(XY^{m_1})^{r_2})|r_1 \leftarrow A\tilde{r}_1 + B\tilde{r}_2, r_2 \leftarrow C\tilde{r}_1 + D\tilde{r}_2, \tilde{r}_1 \overset{\$}{\leftarrow} Z_p^*, \tilde{r}_2 \overset{\$}{\leftarrow} Z_p^*\}\\
&\quad (\text{ Note } A = \frac{(x + m_1 y)\log_g a_0 - \log_g a_0 \log_g b_1}{F}, B = \frac{(m_1 - m_0)y}{F},\\
&\quad C = \frac{\log_g b_0 \log_g a_1 - \log_g b_0 \log_g a_0}{F}, D = \frac{(x + m_0 y)\log_g a_1 - \log_g b_1}{F},\\
&\quad F = (x + m_1 y)\log_g a_1 - \log_g b_1)\\
&= \mathsf{Pr}\{(a_0^{\tilde{r}_1} g^{\tilde{r}_2}, b_0^{\tilde{r}_1}(XY^{m_0})^{\tilde{r}_2})|\tilde{r}_1 \overset{\$}{\leftarrow} Z_p^*, \tilde{r}_2 \overset{\$}{\leftarrow} Z_p^*\}\\
&= \mathsf{Pr}\{\Xi' \overset{\$}{\leftarrow} Rnd(m_0, \Upsilon_0, \Xi_0)|(pk, sk) \overset{\$}{\leftarrow} Gen(1^k), (\langle m_0, \Upsilon_0, \Xi_0\rangle, \langle m_1, \Upsilon_1, \Xi_1\rangle) \overset{\$}{\leftarrow} \mathcal{A}(sk)\}
\end{aligned}
$$

□

Note that CL04* has been implicitly adopted to build group signatures with verifier-local revocation and backward unlinkability in the full version of [ZL06], but it has not been identified as a stand-alone signature.

The following is a summary table of all old and new UR signatures known so far. PU denotes perfectly unlinkability; F denotes Σ-protocol friendly.

Scheme	Signature	Randomization	OWF	Security	Σ
CL02	$\Upsilon=(e,s)$, $\Xi=(A)$	Υ'=$(e,s+re)$, Ξ'=(Ab^r)	$f(m)=a^m$	ACMA, PU	F
CL04+	$\Upsilon=(s)$, $\Xi=(a,b,c)$	Υ'=(sr_2), Ξ'=$(a^{r_1r_2},b^{r_1r_2},c^{r_1})$	$f(m)=g_1^m$	ACMA, PU	F
Wat05+	$\Upsilon=(s)$, $\Xi=(a,b)$	Υ'=(sr_1), Ξ'=$(a^{r_1}g^{r_2},b^{r_1}(u'\prod_{i=1}^{l}u_i^{m_i})^{r_2})$	$f(m)=\prod_{i=1}^{l}u_i^{m_i}$	ACMA, PU	F
BB04+[a]	$\Upsilon=(s,t)$, $\Xi=(A)$	Υ'=(s,rt), Ξ'=(A^r)	—	ACMA, PU	F
BBS04+	$\Upsilon=(s,t)$, $\Xi=(A)$	Υ'=(s,rt), Ξ'=(A^r)	$f(m)=h_1^m$	CMA, PU	F
NSN04*	$\Upsilon=(a,b,c)$, $\Xi=(A)$	Υ'= $(a$,b+ra,c+$r)$, Ξ'=$(A+rP_{pub})$	$f(m)=mP$	ACMA, PU	F
CL04*	$\Upsilon=(s)$, $\Xi=(a,b)$	Υ'=(sr_1), Ξ'=$(a^{r_1}g^{r_2},b^{r_1}(XY^m)^{r_2})$	$f(m)=g^m$	ACMA, PU	F

[a]See Section 4.3.2.

4 Group Signature from UR Signature

Definition 9 ([BSZ05])**.** *A group signature is a signature scheme composed of the following algorithms between GM (including IA, issuing authority, and OA, opening authority), group members and verifiers.*

- ***Setup:*** *an algorithm run by GM (IA and OA) to generate group public key gpk and group secret key gsk;*
- ***Join:*** *a probabilistic interactive protocol between GM (IA) and a group member candidate. If the protocol finishes successfully, the candidate becomes a new group member with a group signing key gsk_i including member secret key msk_i and member certificate $cert_i$; and GM (IA) adds an entry for i (denoted as reg_i) in its registration table reg storing the protocol transcript, e.g. $cert_i$. Sometimes the procedure is also separated into **Join** and **Iss**, where Join emphasize the part run by group members as well as Iss denotes the part run by IA.*
- ***GSig:*** *a probabilistic algorithm run by a group member, on input a message m and a group signing key $gsk_i=(msk_i,cert_i)$, returns a group signature σ;*
- ***GVer:*** *a deterministic algorithm which, on input a message-signature pair (m,σ) and GM's public key gpk, returns 1 or 0 indicating the group signature is valid or invalid respectively;*
- ***Open:*** *a deterministic algorithm which, on input a message-signature pair (m,σ), secret key gsk of GM (OA), returns identity of the group member who signed the signature, and a proof π.*
- ***Judge:*** *a deterministic algorithm with output of Open as input, returns 1 or 0, i.e., the output of OPEN is valid or invalid.*

4.1 Generic Construction of GS

Select an Σ-protocol friendly UR signature *DS*= (K_s,*Sig,Ver,Rnd*) which is indirectly signable with a one way function f, a probabilistic public encryption *PE*= (K_e, *Enc,Dec*).

Define the following relations:

ρ: $(x,w)\in\rho$ iff $x=f(w)$.

ρ_1: ((pk_e, pk_s, m, C, Ξ), (w, Υ, r)) $\in\rho_1$ iff *Ver* (pk_s, w, (Υ, Ξ))=1 and C=*Enc* (pk_e, $f(w)$, r) and $(pk_s,\cdot)\leftarrow K_s$, $(pk_e,\cdot)\leftarrow K_e$.

ρ_2: $((pk_e,C,m),(w))\in\rho_2$ iff $Dec(pk_e,w,C)=m$ and $(pk_e,\cdot)\leftarrow K_e$.

Assume (P, V), (P_1, V_1) and (P_2, V_2) are non-interactive proofs for relation ρ, ρ_1 and ρ_2, which have access to common reference string R, R_1 and R_2 respectively. Let SIM, SIM_1, SIM_2 be their corresponding simulation algorithm. The detailed definition of non-interactive proof is referred to [BSZ05].

(P, V) is also defined to be with an online extractor (in the random oracle model), i.e., it has the following features (let k be the security parameter) [Fis05]:

Completeness: For any random oracle H, any $(x, w) \in \rho$, and any $\pi \leftarrow P^H(x, w, R)$, it satisfies $\Pr\{V^H(x, \pi, R) = 1\} \geq 1 - \epsilon_1(k)$, where $\epsilon_1(k)$ is a negligible function.

Online Extractor: There exists a probabilistic polynomial time algorithm K, the online extractor, such that the following holds for any algorithm A. Let H be a random oracle, $Q_H(A)$ be the answer sequence H to queries from A. Let $w \leftarrow K(x, \pi, Q_H(A))$, then as a function of k, $\Pr\{(x, w) \notin \rho, V^H(x, \pi, R) = 1\} < \epsilon_2(k)$, where $\epsilon_2(k)$ is a negligible function.

GS is constructed as follows, and the details are described in Table 1 and Table 2.

Setup. Select an instance of DS and PE, let secret key of DS be the secret key of IA, secret key of PE be the secret key of OA.

Join. User i selects its member secret key sk_i in message space of DS, computes $pk_i \leftarrow f(sk_i)$, generates π, a non-interactive zero-knowledge proof of knowledge of sk_i for relation ρ. IA checks the correctness of π and generates a DS signature on sk_i: $cert_i = Sig_f(sk_s, pk_i) = Sig(sk_s, sk_i)$, sets $reg_i = pk_i$. The group signing key of i is $gsk_i = (cert_i, sk_i)$.

GSig. On input (gpk, gsk_i, m), parse $cert_i$ into (Υ, Ξ), firstly derive a new certification $(\Upsilon', \Xi') = Rnd(gpk, sk_i, \Upsilon, \Xi)$; encrypt pk_i with PE: $C \leftarrow Enc(pk_e, pk_i, r_i)$, where r_i is random; then generate π_1, a non-interactive zero-knowledge of proof of knowledge of (sk_i, Υ', r_i) for relation ρ_1; in the end, transfer π_1 into a signature on m using any method of transferring a non-interactive zero-knowledge proof into a signature [FS87, BG90, CD95, CL06], we also use π_1 to note the transferred signature for simplicity. The group signature on m is $\sigma = (C, \Xi', \pi_1)$.

GVer. On input (gpk, m, σ), parse σ as (C, Ξ', π_1), check the correctness of π_1, return 1 if it is correct, return 0 otherwise.

Open. On input $(gpk, ok, reg, m, \sigma)$, parse σ as (C, Ξ', π_1). OA firstly checks the validity of the group signature σ on m, if it is not valid, stops; otherwise decrypts C to get M, and generates π_2, a proof of knowledge of decryption key ok for relation ρ_2. If $M = pk_i$ for some pk_i in reg, return the corresponding index or identity and π_2, else returns zero and π_2.

Judge. Check the validity of the group signature and the output of Open.

User i		Issue Authority
Select sk_i, $pk_i \leftarrow f(sk_i)$,		
$\pi \leftarrow P(pk_i, sk_i, R)$	$\xrightarrow{pk_i, \pi}$	If $V(pk_i, \pi, R) = 1$,
		$cert_i \leftarrow Sig_f(sk_s, pk_i)$,
$gsk_i \leftarrow (pk_i, sk_i, cert_i)$	$\xleftarrow{cert_i}$	$reg_i = pk_i$.

Table 1: Algorithm Join of GS.

Comparison. The above generic construction is a particular case of the construction in [BSZ05]:

In [BSZ05], the group signature is $\sigma = (C, \pi_1) = (Enc(pk_e, < i, pk_i, \Upsilon, \Xi, s >, r_i), \pi_1)$, where $s = S(sk_i, m)$ and π_1 is a proof of knowledge of $(pk_i, \Upsilon, \Gamma, s, r_i)$ satisfying $Ver(pk_s, < i, pk_i >, (\Upsilon, \Xi)) = 1$, $C = Enc(pk_e, < i, pk_i, \Upsilon, \Xi, s >, r_i)$, and $V(pk_i, m, s) = 1$. (S, V) is the signature generation and verification algorithms of an independent signature scheme.

However in this construction, the group signature is $\sigma = (C, \Xi', \pi_1) = (Enc(pk_e, pk_i, r_i), \Xi', \pi_1)$, where π_1 is a transformed signature of the proof of knowledge of (sk_i, Υ', r_i) satisfying $Ver(pk_s, sk_i, (\Upsilon', \Xi')) = 1$ and $C = Enc(pk_e, f(sk_i), r_i)$.

The construction is more efficient in that less items are encrypted in C enabling efficient proof of knowledge of encrypted context.

Algorithm **Setup**(1^k):
$R \xleftarrow{\$} \{0,1\}^{P(k)}$, $R_1 \xleftarrow{\$} \{0,1\}^{P_1(k)}$,
$R_2 \xleftarrow{\$} \{0,1\}^{P_2(k)}$, $(pk_s, sk_s) \leftarrow K_s(1^k)$,
$(pk_e, sk_e) \leftarrow K_e(1^k)$,
$gpk = (R, R_1, R_2, pk_e, pk_s)$,
$ok = (sk_e)$, $ik = (sk_s)$.
return (gpk, ok, ik).

Algorithm **GVer**(gpk, m, σ):
Parse σ as (C, Ξ', π_1),
Parse gpk as $gpk = (R, R_1, R_2, pk_e, pk_s)$,
Return $V_1((pk_e, pk_s, m, C, \Xi'), \pi_1, R_1)$.

Algorithm **Judge**$(gpk, reg, m, \sigma, i, M, \pi_2)$:
Parse gpk as $gpk = (R, R_1, R_2, pk_e, pk_s)$,
Parse σ as (C, Ξ', π_1),
If $GVer(gpk, m, \sigma) = 0$, return $\perp$.
If $i = 0$, and $M \neq Reg_j$ for all j,
return $V_2((pk_e, C, M), \pi_2, R_2)$,
else if $M = reg_i$,
return $V_2((pk_e, C, M), \pi_2, R_2)$.

Algorithm **GSig**(gpk, gsk_i, m):
Parse $cert_i$ as (Υ, Ξ),
Parse gpk as $(R, R_1, R_2, pk_e, pk_s)$,
$(\Upsilon', \Xi') = Rnd(gpk, sk_i, \Upsilon, \Xi)$;
$C \leftarrow Enc(pk_e, pk_i, r_i)$, r_i random;
$\pi_1 = P_1((pk_e, pk_s, m, C, \Xi'), (sk_i, \Upsilon', r_i), R_1)$.
$\sigma = (C, \Xi', \pi_1)$.
return σ.

Algorithm **Open**$(gpk, ok, reg, m, \sigma)$:
Parse gpk as $gpk = (R, R_1, R_2, pk_e, pk_s)$,
Parse σ as (C, Ξ', π_1),
If $GVer(gpk, m, \sigma) = 0$, return $\perp$.
$M \leftarrow Dec(sk_e, C)$,
If $M = reg_i$, $\exists i$,
$\pi_2 = P_2((pk_e, C, M), (sk_e), R_2)$,
return (i, τ), where $\tau = (M, \pi_2)$;
else return $(0, \tau)$, where $\tau = (M, \pi_2)$.

Table 2: Algorithms Setup, GSig, GVer, Open, Judge of GS.

4.2 Security Proofs

The security of the generic group signature utilizing unlinkable randomizable signature follows the generic construction in [BSZ05] since the former is a particular case of the latter.

To be complete, we provide the following lemmas with proofs (see Appendix C.1, C.2, C.3 respectively) under a variant model (see Appendix B).

Lemma 4.1. *GS is anonymous if PE is IND-CCA2, (P_1, V_1) is a simulation sound, computational zero-knowledge proof, (P_2, V_2) is a computational zero-knowledge proof.*

Lemma 4.2. *GS is traceable if* DS *is wUF-ACMA, (P_1, V_1), (P_2, V_2) are sound proofs of knowledge and (P, V) is a proof of knowledge with online extractor (in random oracle model).*

Lemma 4.3. *GS is non-frameable if f is one way function, (P, V) is a computational zero-knowledge proof, (P_1, V_1) and (P_2, V_2) are sound proofs of knowledge.*

4.3 Improvement to a Group Signature

4.3.1 Review of KY05's Group Signature

Setup. At first, select the following public parameters:

1. two groups $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$ of order p(length is l_p bits), and there exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.
2. an RSA modulus n of l_n bits.
3. three integer ranges $S = \{2^{l-1} - 2^{l_p-2} + 1, ..., 2^{l-1} + 2^{l_p-2} - 1\}$, $S' = \{2^{l'} - 2^{\mu'} + 1, ..., 2^{l'} + 2^{\mu'} - 1\}$, $S = \{2^{l''} - 2^{\mu''} + 1, ..., 2^{l''} + 2^{\mu''} - 1\}$, where $l' = l - 1$, $\mu' = \lfloor (l_p - 4)/\epsilon \rfloor - k$, $l'' = l'/2$, $\mu'' = \mu'/2$, $\varepsilon > 1$. $S' \subset S \subset Z_{\phi(n)}$, the upper bound of S'' is smaller than the lower bound of S.
4. an RSA modulus N of l_N bits, choose $G \in QR_{N^2}$ so that $\langle G \rangle$ is also N-th residues, $\sharp\langle G \rangle = \phi(N)/4$.

Then IA selects

- $\gamma, \delta \xleftarrow{\$} Z_p$, set $w = g_2^{\gamma}$, $v = g_2^{\delta}$.
- $\alpha, \beta \xleftarrow{\$} Z_p$, $u \xleftarrow{\$} \mathbb{G}_1$, set $u' = u^{\alpha/\beta}$, $h = u^{\alpha}(u')^{\beta} = u^{2\alpha}$.
- $g, f_1, f_2, f_3 \xleftarrow{\$} QR_n$.
- a collision resistant hash function HASH.

OA selects

- $a_1, a_2, a_3 \xleftarrow{\$} Z_{\lfloor N/4 \rfloor}$, set $H_1 = G^{a_1}$, $H_2 = G^{a_2}$, $H_3 = G^{a_3}$.
- a universal one-way hash function family UOHF, and a hash key hk.

Group public key $gpk = \{g_1, g_2, u, u', h, w, v, g, f_1, f_2, f_3, n, N, G, H_1, H_2, H_3, hk, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathsf{UOHF}\}$. Group secret key $gsk = \{\gamma, \delta, a_1, a_2, a_3\}$.

Join. The protocol is diagrammed in the following table.

User		Issue Authority
$x = x_1 x_2$, where $x \in \xleftarrow{\$} S'$ and $x_1 \in \xleftarrow{\$} S''$,	$\xrightarrow{x}$	If $x \in S'$, $r \xleftarrow{\$} Z_p^*$, $s \xleftarrow{\$} Z_p^*$
$e(\sigma, w g_2^x v^r)? = e(g_1, g_2)^s$	$\xleftarrow{(r,s,\sigma)}$	$\sigma \leftarrow g_1^{\frac{s}{\gamma + x + \delta r}}$
$cert = (x, r, s, \sigma)$, $msk = (x_1, x_2)$		

Table 3: Algorithm Join of [KY05].

GSig. If a user with member certificate (x, σ, r) and member secret key (x_1, x_2) wants to generate a group signature on m, he firstly computes $T_1, T_2, T_3, T_4, T_5, C_0, C_1, C_3$ as described in the following table.

$T_1 = u^z$	$z \xleftarrow{\$} Z_p$ in $\mathbb{G}_1$
$T_2 = (u')^{z'}$	$z' \xleftarrow{\$} Z_p$ in $\mathbb{G}_1$
$T_3 = h^{z+z'}\sigma$	in $\mathbb{G}_1$
$T_4 = g^y f_1^{x_1}$	$y \xleftarrow{\$} S(1, 2^{l_n - 2})$ in QR_n
$T_5 = g^{y'} f_2^{x_2} f_3^t$	$y' \xleftarrow{\$} S(1, 2^{l_n - 2})$ in QR_n
$C_0 = G^t$	$t \xleftarrow{\$} S(1, 2^{l_N - 2})$ in $Z_{N^2}^*$
$C_1 = H_1^t (1+N)^x$	in $Z_{N^2}^*$
$C_2 = \parallel (H_2 H_3^{\mathcal{H}(hk, C_0, C_1)})^t \parallel$	in $Z_{N^2}^*$

Then he generates a signature of knowledge by applying the Fiat-Shamir heuristic [FS87] on a proof of knowledge of the fourteen witnesses θ_z, $\theta_{z'}$, θ_x, θ_{xz}, $\theta_{xz'}$, θ_r, $\theta_{rz'}$, θ_{x_1}, θ_{x_2}, θ_y, $\theta_{y'}$, θ_{yx_2}, θ_t that satisfy the following relations:

$$\boxed{\begin{array}{llll}
T_1 = u^{\theta_z}, & T_2 = (u')^{\theta_{z'}}, & 1 = T_1^{-\theta_x} u^{\theta_{xz}}, & 1 = T_2^{-\theta_x} (u')^{\theta_{xz'}}, \\
1 = T_1^{-\theta_r} u^{\theta_{rz}}, & 1 = T_2^{-\theta_r} (u')^{\theta_{rz'}}, & T_4 = g^{\theta_y} f_1^{\theta_{x_1}}, & 1 = T_4^{-\theta_{x_2}} g^{\theta_{yx_2}} f_1^{\theta_x}, \\
T_5 = g^{\theta_{y'}} f_2^{\theta_{x_2}} f_3^{\theta_t}, & \theta_x \in S', & \theta_{x'} \in S'', & C_0 = G^{\theta_t}, \\
C_1 = H_1^{\theta_t} (1+N)^{\theta_x}, & C_2^2 = (H_2 H_3^{\mathcal{H}(hk, C_0, C_1)})^{2\theta_t}, & & \\
\multicolumn{4}{l}{e(g_1, g_2)/e(T_3, w) = e(T_3, v)^{\theta_r} e(T_3, g_2)^{\theta_x} e(h, g_2)^{-\theta_{xz} - \theta_{xz'}} e(h, v)^{-\theta_{rz} - \theta_{rz'}} e(h, w)^{-\theta_z - \theta_{z'}}}
\end{array}}$$

The realization of the above signature of knowledge is quite standard, so we omit it here. The output is (T_1, T_2, T_3, T_4, T_5, C_0, C_1, C_2, c, s_z, $s_{z'}$, s_{xz}, $s_{xz'}$, s_r, s_{rz}, $s_{rz'}$, s_x, s_{x_1}, s_{x_2}, s_y, $s_{y'}$, s_{yx_2}, s_t).

GVer. The verification is achieved by checking if all the following conditions hold:

$$s_x \in \pm\{0,1\}^{\epsilon\mu'+k+1} \ \wedge \ s_{x_1} \in \pm\{0,1\}^{\epsilon\mu''+k+1} \ \wedge \ C_0, C_1, c_2 \in Z^*_{N^2} \ \wedge \ C_2 \leq N^2/2$$

$$\begin{aligned} c = \mathsf{HASH}(&m\|T_1\|T_2\|T_3\|T_4\|T_5\|u^{s_z}T_1^c\|(u')^{s_{z'}}T_2^c\|T_1^{s_x+c2^{l'}}u^{s_{xz}} \\ &\|T_2^{s_x+c2^{l'}}(u')^{s_{xz}}\|T_1^{s_r}u^{-s_{rz}}\|T_2^{s_r}(u')^{-s_{rz}} \\ &\|e(T_3,v)^{s_r}e(T_3,g_2)^{s_x-c2^{l'}}e(h,g_2)^{-s_{xz}-s_{xz'}} \\ &e(h,v)^{-s_{rz}-s_{rz'}}e(h,w)^{s_z+s_{z'}}e(g_1,g_2)^c e(T_3,w)^{-c} \\ &\|T_4^c g^{s_y-c}f_1^{s_{x_1}-c2^{l''}}\|T_4^{s_{x_2}-c}g^{-s_{yx_2}+2c}f_1^{-s_x+c2^{l'}} \\ &\|T_5^c g^{s_{y'}-c}f_2^{s_{x_2}-c}f_3^{s_t-c}\|C_0^c G^{s_t-c} \\ &\|C_1^c H_1^{s_t-c}(1+N)^{s_x-c2^{l'}}\|C_2^c(H_2^2 H_3^{2\mathcal{H}(hk,C_0,C_1)})^{s_t-c}) \end{aligned}$$

Open. Firstly the group signature is verified as well as the relation $C_2^2 = C_0^{2(a_2+a_3\mathcal{H}(hk,C_0,C-1))}$ is checked. If all the tests pass, OA computes $x = (C_1C_0^{-a_1} - 1)/N$, then checks if there exists a matching member certificate in the database maintained by IA.

4.3.2 Group Signature KY05+

Replacing the member certificate signature with the following BB04+ signature, the scheme in [KY05] can be improved.

BB04+.	Let $\mathbb{G}_1$, $\mathbb{G}_2$ be two p order cyclic groups, and there exists a bilinear map $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$. $\mathbb{G}_1 = \langle g \rangle$, $\mathbb{G}_2 = \langle \tilde{g} \rangle$.
Gen.	It chooses $x \xleftarrow{\$} Z_p^*$, $y \xleftarrow{\$} Z_p^*$, and sets $sk = (x, y)$, $pk = (p, \mathbb{G}_1, \mathbb{G}_2, g, \tilde{g}, X, Y, e)$, where $X = \tilde{g}^x$, $Y = \tilde{g}^y$.
Sig.	On input message m, secret key sk, and public key pk, choose $(s,t) \xleftarrow{\$} Z_p^{*2}$, compute $A = g^{\frac{t}{x+m+ys}}$, output the signature (Υ, Ξ) where $\Upsilon = (s,t)$, $\Xi = (A)$. Note that $(s, A^{\frac{1}{t}})$ is a valid [BB04] signature on m.
Ver.	On input pk, message m, and purported signature $(\Upsilon, \Xi) = (s,t,A)$, check that $e(A, XY^s\tilde{g}^m) = e(g^t, \tilde{g})$.
Rnd.	On input pk, message m, and a signature $(\Upsilon, \Xi) = (s,t,A)$, choose $r \xleftarrow{\$} Z_p^*$, output (Υ', Ξ') where $\Upsilon' = (s', t') = (s, rt)$, $\Xi' = (A') = (A^r)$.

BB04+ can be proved wUF-ACMA similarly to the scheme [BB04]. Briefly, suppose $\mathcal{A}$ is an adversary of BB04+, then an adversary $\mathcal{B}$ of [BB04] is available: when $\mathcal{A}$ queries signature on m, $\mathcal{B}$ transfers the query to signature oracle of [BB04]; $\mathcal{B}$ will get a response from the signature oracle, i.e., (s, A), where $e(A, XY^s\tilde{g}^m) = e(g, \tilde{g})$, then $\mathcal{B}$ chooses $t \xleftarrow{\$} Z_p^*$, sends (s, t, A^t) to $\mathcal{A}$. If $\mathcal{A}$ outputs a signature (s^*, t^*, A^*) on a message m^* that it has never queried, then $(s^*, A^{*\frac{1}{t^*}})$ is a valid BB04+ signature on m^*, which $\mathcal{B}$ has never queried.

Obviously, BB04+ is perfectly unlinkable because each randomized Ξ' only consists of one element that is generated independently and randomly each time, but it is not indirectly signable because m must be known to calculate a signature on it.

BB04+ is Σ-protocol friendly, because there exists an efficient Σ-protocol for the relation $\{(m,s,t)|\ e(A,X)\ e(A,Y)^s\ e(A,\tilde{g})^m = e(g,\tilde{g})^t\}$.

Now we turn back to the group signature of KY05+. Public parameters and algorithms Setup, Join (Table 3), Open are exactly as [KY05], except that key-setup for linear ElGamal encryption is eliminated.

GSig. If a user with member certificate (x, σ, r) and member secret key (x_1, x_2) wants to generate a group signature on m, he firstly computes $(\sigma', s', T_4, T_5, C_0, C_1, C_2)$ as described in the following table.

$\sigma' = \sigma^{r'}$, $s' = r's$	$r' \xleftarrow{\$} Z_p^*$ in $\mathbb{G}_1$
$T_4 = g^y f_1^{x_1}$	$y \xleftarrow{\$} S(1, 2^{l_n-2})$ in QR_n
$T_5 = g^{y'} f_2^{x_2} f_3^t$	$y' \xleftarrow{\$} S(1, 2^{l_n-2})$ in QR_n
$C_0 = G^t$	$t \xleftarrow{\$} S(1, 2^{l_N-2})$ in $Z_{N^2}^*$
$C_1 = H_1^t(1+N)^x$	in $Z_{N^2}^*$
$C_2 = \| (H_2 H_3^{\mathcal{H}(hk,C_0,C_1)})^t \|$	in $Z_{N^2}^*$

Then he generates a signature of knowledge by applying the Fiat-Shamir heuristic [FS87] on a proof of knowledge of the nine witnesses $(\theta_x, \theta_{x_1}, \theta_{x_2}, \theta_y, \theta_{y'}, \theta_{yx_2}, \theta_t, \theta_r, \theta_{s'})$ that satisfy the specified relations in the following table.

$g^{\theta_y} f_1^{\theta_{x_1}} = T_4,$	$g^{\theta_{y'}} f_2^{\theta_{x_2}} f_3^{\theta_t} = T_5,$
$T_4^{-\theta_{x_2}} g^{\theta_{yx_2}} f_1^{\theta_x} = 1,$	$e(\sigma', w g_2^{\theta_x} v^{\theta_r}) = e(g_1, g_2)^{\theta_{s'}},$
$G^{\theta_t} = C_0,$	$H_1^{\theta_t}(1+N)^{\theta_x} = C_1,$
$(H_2 H_3^{\mathcal{H}(hk,C_0,C_1)})^{2\theta_t} = C_2^2,$	$\theta_x \in S',\ \theta_{x'} \in S''.$

Note that the number of witnesses that need proving is fewer than that of [KY05]. Thus a group signature of KY05+ is $(\sigma', T_4, T_5, C_0, C_1, C_2, c, s_r, s_x, s_{x_1}, s_{x_2}, s_y, s_{y'}, s_{yx_2}, s_t, s_{s'})$, about $7|p| = 1190$ bits shorter than [KY05].

If we view $x = x_1 x_2$ as a one way function since factoring of x is hard, KY05+ is an application of the proposed generic construction on BB04+ except that a non-interactive zero-knowledge proof of knowledge with online extractor is not adopted in Join. The security of it follows from that of proposed generic construction and [KY05].

5 Conclusion

We have formalized the characteristics of randomizable signatures that are required to build secure group signatures. Design of efficient secure group signature can be boiled down to designing Σ-protocol friendly unlinkable randomizable signature with indirect signability.

We also found almost all efficient group signatures known so far are actually in this line of unlinkable randomizable signatures. For the first time, we proposed the unlinkable randomizable signature version of the scheme [NSN04]. We proposed two new unlinkable randomizable signatures called Wat05+, CL04* which will result in new group signatures. We also improved the scheme in [KY05] by replacing the member certificate generation signature with an unlinkable randomizable signature.

References

[ACdMH] Giuseppe Ateniese, Jan Camenisch, Breno de Medeiros, and Susan Hohenberger. Practical group signatures without random oracles. Cryptology ePrint Archive, Report 2005/385.

[ACJT00] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Advances in Cryptology - CRYPTO'00*, LNCS 1880, pages 255–270. Springer, 2000.

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *Advances in Cryptology - EUROCRYPT'04*, LNCS 3027, pages 56–73. Springer, 2004.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Advances in Cryptology - CRYPTO'04*, LNCS 3152, pages 45–55. Springer, 2004.

[BG90] Mihir Bellare and Shafi Goldwasser. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. In *Advances in Cryptology - CRYPTO'89*, LNCS 435, pages 194–211. Springer, 1990.

[Bou00] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *Advances in Cryptology - EUROCRYPT'00*, LNCS 1807, pages 431–444. Springer, 2000.

[BS04] Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In *ACM conference on Computer and Communications Security - CCS'04*, pages 168–177. ACM Press, 2004.

[BSZ05] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In *Topics in Cryptology – CT-RSA'05*, LNCS 3376, pages 136–153. Springer, 2005. Full Paper at http://www-cse.ucsd.edu/˜mihir/papers/dgs.html.

[BW06] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In *Advances in Cryptology - EUROCRYPT'06*, pages 427–444. Springer, 2006. Full paper at http://eprint.iacr.org/2005/381.

[BW07] Xavier Boyen and Brent Waters. Full-domain subgroup hiding and constant-size group signatures. In *Public Key Cryptography - PKC'07*, LNCS 4450, pages ???–??? Springer, 2007.

[CD95] Ronald Cramer and Ivan Damgård. Secure signature schemes based on interactive protocols. In *Advances in Cryptology - CRYPTO'95*, LNCS 963, pages 297–310. Springer, 1995.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Advances in Cryptology - CRYPTO'02*, LNCS 2442, pages 61–76. Springer, 2002.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Advances in Cryptology - CRYPTO'04*, LNCS 3152, pages 56–72. Springer, 2004.

[CL06] Melissa Chase and Anna Lysyanskaya. On signatures of knowledge. In *Advances in Cryptology - CRYPTO'06*, LNCS 4117, pages 78–96. Springer, 2006.

[CM98a] Jan Camenisch and Markus Michels. A group signature scheme based on an RSA-variant. In *Technical Report RS-98-27*. BRICS, University of Aarhus, 1998.

[CM98b] Jan Camenish and Markus Michels. A group signature scheme with improved efficiency. In *Advances in Cryptology - ASIACRYPT'98*, LNCS 1514, pages 160–174. Springer, 1998.

[CM99] Jan Camenisch and Markus Michels. Separability and efficiency for generic group signature schemes. In *Advances in Cryptology - CRYPTO'99*, LNCS 1666, pages 413–430. Springer, 1999.

[CP93] David Chaum and Torben Pryds Pedersen. Wallet database with observers. In *Advances in Cryptology - CRYPTO'92*, LNCS 740, pages 89–105. Springer, 1993.

[CS97] Jan Camenisch and Markus Stadler. Efficient group signatures schemes for large groups. In *Advances in Cryptology - CRYPTO'97*, LNCS 1296, pages 410–424. Springer, 1997.

[CvH91] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology - EUROCRYPT'91*, LNCS 547, pages 257–265. Springer, 1991.

[Dam05] Ivan Damgård. *On Sigma-protocols*. http://www.daimi.au.dk/˜ivan/CPT.html, 2005.

[DK01] Hans Delfs and Helmut Knebl. *Introduction to Cryptography : Principles and Applications*. Springer, December 2001.

[Fis03] Marc Fischlin. The Cramer-Shoup Strong-RSA signature scheme revisited. In *Public Key Cryptography - PKC'03*, LNCS 2567, pages 116–129. Springer, 2003.

[Fis05] Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractor. In *Advances in Cryptology - CRYPTO'05*, LNCS 3621, pages 152–168. Springer, 2005.

[FO97] E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology - CRYPTO'97*, LNCS 1294, pages 16–30. Springer, 1997.

[FS87] A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO'86*, LNCS 263, pages 186–194. Springer, 1987.

[Gir91] Marc Girault. An identity-based identification scheme based on discrete logarithms modulo a composite number. In *Advances in Cryptology - EUROCRYPT'90*, LNCS 473, pages 481–486. Springer, 1991.

[Gol01] Oded Goldreich. *Foundations of Cryptography*, volume Basic Tools. Cambridge University Press, 2001.

[KTY04] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable signatures. In *Advances in Cryptology - EUROCRYPT'04*, LNCS 3027, pages 571–589. Springer, 2004.

[KY05] Aggelos Kiayias and Moti Yung. Group signatures with efficient concurrent join. In *Advances in Cryptology - EUROCRYPT'05*, LNCS 3494, pages 198–214. Springer, 2005.

[KY06] Aggelos Kiayias and Moti Yung. Secure scalable group signature with dynamic joins and separable authorities. *International Journal of Security and Networks*, 1(1/2):24–45, 2006. Also titled with "Group Signatures: Provable Security, Efficient Constructions and Anonymity from Trapdoor-Holders" at http://eprint.iacr.org/2004/076.

[MvOV96] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[NSN04] Lan Nguyen and Rei Safavi-Naini. Efficient and provably secure trapdoor-free group signature schemes from bilinear pairings. In *Advances in Cryptology - ASIACRYPT'04*, LNCS 3329, pages 372–386. Springer, 2004.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT'05*, LNCS 3494, pages 114–127. Springer, 2005.

[ZL06] Sujing Zhou and Dongdai Lin. Shorter verifier-local revocation group signatures from bilinear maps. In *Cryptology and Network Security - CANS'06*, LNCS 4301, pages 126–143. Springer, 2006. Full text at http://eprint.iacr.org/2006/286.

A Proof of Lemma 3.3

Proof. Suppose there exists an adversary $\mathcal{B}$ to the signature, we now construct an adversary $\mathcal{A}$ to resolve q-SDH problem ([BB04]) in $\mathbb{G}$: to calculate $(c, \frac{1}{z+c}Q), c \in Z_p^*$ given a random tuple $(Q, zQ, ..., z^qQ)$.

$\mathcal{B}$ should be given public key of the signature and access to oracle Sig answered by $\mathcal{A}$, obtaining $q_{sig}(\leq q-1)$ message-signature pairs $(m_i, a_i, b_i, c_i, A_i), i = 1, ..., q_{sig}$, $\mathcal{B}$ wins by outputting a forgery, i.e., a new message-signature $(m^*, a^*, b^*, c^*, A^*)$ that $m^* \notin \{m_1, ..., m_{q_{sig}}\}$. There may be two different types of forgeries. The first type, $a^* \neq a_i, \forall i$; the second type, $a^* = a_l, \exists l \in [1, q_{sig}]$. $\mathcal{A}$ will choose a random bit from $\{1, 2\}$ to indicate its guess for the forgery type, and simulate accordingly. (Note that $A = \frac{1}{\gamma+a}[mP + (b+\gamma c)P_{pub} + P_0] = \frac{1}{\gamma+a}[mP + (b-ac)P_{pub} + P_0] + cP_{pub}$).

Type 1. $a^* \neq a_i, \forall i$.

$\mathcal{A}$ selects $a_i \xleftarrow{\$} Z_p^*, i \in [1, q_{sig}]$ that are not equal to each other, and $s \xleftarrow{\$} Z_p^*$, let $f(y) = \prod_{i=1}^{q_{sig}}(y+a_i)$, $\gamma = z$, sets public key as $P = f(z)Q$, $P_{pub} = zf(z)Q$, $P_0 = sf(z)Q$, which are computable from $(Q, zQ, ..., z^qQ)$.

When $\mathcal{B}$ queries about a signature on m_i, $\mathcal{A}$ firstly selects $b_i, c_i \xleftarrow{\$} Z_p^*$, calculates $A_i = \frac{1}{z+a_i}[m_iP + (b_i - a_ic_i)P_{pub} + P_0] + c_iP_{pub}$, which is computable from $(Q, zQ, ..., z^qQ)$ since $(z+a_i)|f(z)$.

The forgery $(m^*, a^*, b^*, c^*, A^*)$ satisfies $A^* = \frac{1}{z+a^*}[m^*P + (b^* - a^*c^*)P_{pub} + P_0] + c^*P_{pub}$, i.e., $A^* - c^*P_{pub} = \frac{1}{z+a^*}[(m^*+s+(b^*-a^*c^*)z)\prod_{i=1}^{q_{sig}}(z+a_i)Q]$, the probability of $m^*+s = (b^*-a^*c^*)a^*$ is negligible otherwise $\mathcal{B}$ can be invoked to solve discrete logarithm problem in $\mathbb{G}$ if z is chosen by $\mathcal{A}$ and sQ is given as a discrete logarithm challenge. Then there exist $g(z)$, $r \neq 0 \bmod p$ that $(m^* + s + (b^* - a^*c^*)z)\prod_{i=1}^{q_{sig}}(z+a_i) = g(z)(z+a^*) + r$, so $(a^*, \frac{1}{z+a^*}Q)$, computable from A^* and $(Q, zQ, ..., z^qQ)$, is a resolution to the q-SDH challenge.

Type 2. $a^* = a_l, \exists l \in [1, q_{sig}]$.

$\mathcal{A}$ selects $a_i \xleftarrow{\$} Z_p^*, i \in [1, q_{sig}]$ that are not equal to each other, $t \xleftarrow{\$} Z_p^*$, and $d \xleftarrow{\$} Z_p^*$, let $f(y) = \prod_{i=1}^{q_{sig}}(y+a_i)$, $\gamma = z - a_l$, sets public key as $P = \frac{f(z-a_l)}{z}Q = \prod_{i=1, i\neq l}^{q_{sig}}(z - a_l + a_i)Q$, $P_{pub} = (z-a_l)P$, $P_0 = tzP + dP = t\prod_{i=1}^{q_{sig}}(z - a_l + a_i)Q + dP$, which are computable from $(Q, zQ, ..., z^qQ)$.

When $\mathcal{B}$ queries about a signature on $m_i, i \neq l$, $\mathcal{A}$ firstly selects $b_i, c_i \xleftarrow{\$} Z_p^*$, calculates $A_i = \frac{1}{z-a_l+a_i}[m_iP + (b_i - a_ic_i)P_{pub} + P_0] + c_iP_{pub}$, which is computable from $(Q, zQ, ..., z^qQ)$ since $(z - a_l + a_i)|f(z-a_l)$.

When $\mathcal{B}$ queries about a signature on m_l, $\mathcal{A}$ firstly selects $b_l, c_l, s \in Z_p^*$ so that $b_l - a_lc_l = (d+m_l)a_l^{-1}$, and $s = t + (d+m_l)a_l^{-1}$, then it can be verified that $m_lP + (b_l - a_lc_l)P_{pub} + P_0 = szP$, so $A_l = \frac{1}{\gamma+a_l}[m_lP + (b_l - a_lc_l)P_{pub} + P_0] + c_lP_{pub} = sP + c_lP_{pub}$ is computable.

The forgery $(m^*, a^*, b^*, c^*, A^*)$ satisfies $A^* = \frac{1}{\gamma+a^*}[m^*P + (b^* - a^*c^*)P_{pub} + P_0] + c^*P_{pub}$, i.e., $A^* - c^*P_{pub} = \frac{1}{z}[m^* - a_l(b^* - a^*c^*) + d + (b^* - a^*c^* + t)z]\prod_{i=1, i\neq l}^{q_{sig}}(z - a_l + a_i)Q$, the probability of $m^* - a_l(b^* - a^*c^*) + d = 0 \bmod p$ is negligible otherwise $\mathcal{B}$ can be invoked to solve discrete logarithm problem in $\mathbb{G}$ if z is chosen by $\mathcal{A}$ and dQ is given as a discrete logarithm challenge. Then there exist $g(z)$, $r \neq 0 \bmod p$ that $[m^* - a_l(b^* - a^*c^*) + d + (b^* - a^*c^* + t)z]\prod_{i=1, i\neq l}^{q_{sig}}(z - a_l + a_i) = g(z)z + r$, so $(0, \frac{1}{z}Q)$, computable from A^* and $(Q, zQ, ..., z^qQ)$, is a resolution to the q-SDH challenge. Note that any algorithm for $\frac{1}{z}Q$ can be used to calculate a $(c \neq 0, \frac{1}{z+c}Q)$. $\square$

B A Formal Model of Group Signature - A Variant of [BSZ05]

[BSZ05]'s model assumes that IA can not delete contents of the registration table Reg; OA is assumed only partially corrupted in considering traceability, i.e., OA will abide by specified algorithm Open. The existence of a secure (private and authentic) channel between any prospective group member and IA is also assumed.

For simplicity, we additionally assume that IA will not generate a new group signing key for an existing member, nor will IA modify existing records in Reg; OA will not report an existing member to be non-existent or another existing member after it has opened a group signature according to specified algorithms.

The additional assumption about IA can be guaranteed by introducing an additional trusted third authority CA independent from IA as explicitly defined in the model of [BSZ05]: every member is given a user public key from CA and a user secret key kept to himself; in Join, a member signed on whatever he has generated and sent to IA; IA stores the signed transcript in registration table; execution of Open should reveal the signer identity and stored transcript carrying a signature by the signer.

The additional assumption about OA can be guaranteed by granting accesses of reading/seaching Reg to judgers (the executors of algorithm Judge).

We define the oracles similar to [BSZ05]. It is assumed that several global variables are maintained by the oracles: *HU*, a set of honest users; *CU*, a set of corrupted users; *GSet*, a set of message signature pairs; and *Chlist*, a set of challenged message signature pairs. Note that not all the oracles will be available to adversaries in defining a certain security feature.

AddU (i): If $i \in HU \cup CU$, the oracle returns $\perp$, else adds i to HU, executes algorithm Join.

CrptU (i): If $i \in HU \cup CU$, the oracle returns $\perp$, else $CU \leftarrow CU \cup \{i\}$, and awaits an oracle query to *SndToI*.

SndToI(i, M_{in}): If $i \notin CU$, the oracle returns $\perp$; else it plays the role of IA in algorithm Join replying to M_{in}.

SndToU(i, M_{in}): If $i \in HU \cup CU$, the oracle returns $\perp$, else it plays the role of user i in algorithm Join, $HU \leftarrow HU \cup \{i\}$.

USK (i): If $i \in HU$, the oracle returns sk_i and gsk_i, $CU \leftarrow CU \cup \{i\}$, $HU \leftarrow HU \setminus \{i\}$; else returns $\perp$.

RReg (i): The oracle returns reg_i.

WReg $(i,\ s)$: The oracle sets $reg_i = s$ if i has not been added in reg.

GSig $(i,\ m)$: If $i \notin HU$, the oracle returns $\perp$, else returns a group signature σ on m by user i. $GSet \leftarrow GSet \cup \{(i, m, \sigma)\}$.

Ch $(b,\ i_0,\ i_1,\ m)$: If $i_0 \notin HU \cup CU$ or $i_1 \notin HU \cup CU$, the oracle returns $\perp$, else generates a valid group signature σ with i_b being the signer. $Chlist \leftarrow Chlist \cup \{(m, \sigma)\}$.

Open $(m,\ \sigma)$: If $(m, \sigma) \in Chlist$, the oracle returns $\perp$, else if (m, σ) is valid, the oracle returns Open(m, σ).

CrptIA: The oracle returns the secret key ik of IA.

CrptOA: The oracle returns the secret key ok of OA.

We say an oracle is over another oracle if availability of the oracle implies functions of another oracle. For example, *WReg* is over *RReg* since the adversary can try to remember everything it has written to Reg; *CrptIA* is over *CrptU*, *SndToI* since knowledge of ik enables the adversary answer the two oracles itself; *CrptOA* is over *Open*. Note that we do not let *CrptIA* (*CrptOA*) over *WReg* (*RReg*) to provide flexibility when accesses to the database Reg are granted by an independent DBA (database administrator).

Correctness. For any adversary that is not computationally restricted, a group signature generated by an honest group member is always valid; algorithm Open will always correctly identify the signer given the above group signature; the output of Open will always be accepted by algorithm Judge.

Experiment $\mathsf{Exp}^{corr}_{GS,A}(k)$
$(gpk, ik, ok) \xleftarrow{\$} \text{Setup}(1^k)$; $HU \leftarrow \varnothing$;
$(i, m) \xleftarrow{\$} A(gpk : AddU, RReg)$,
If $i \notin HU$, return 0;
$\sigma \leftarrow \text{GSig}(gpk, gsk_i, m)$; $(j, \tau) \leftarrow \text{Open}(gpk, ok, reg, m, \sigma)$,
If $\text{GVer}(gpk, m, \sigma) = 0$, or $j \neq i$, or $\text{Judge}(gpk, i, reg, m, \sigma, \tau) = 0$,
then return 1 else return 0.

Table 4: Correctness.

Anonymity. Imagine a polynomial time adversary $\mathcal{A}$, whose goal is to distinguish the signer of a group signature $\sigma \leftarrow Ch(b, i_0, i_1, m)$ between i_0, i_1, where i_0, i_1, m are chosen by $\mathcal{A}$ itself.

Naturally the adversary $\mathcal{A}$ might want to get the group signing keys of i_0, i_1 or some other honest group members (through oracle *USK*); it might want to obtain some group signatures signed by i_0, i_1 (through oracle *GSig*); it might want to see some outputs of OA (through oracle *Open* except (m, σ)); it might also try to corrupt some group members by running Join with IA (through oracles *CrptU* and *SndToI*); it might observe the communication of some honest members joining in (through *SndToU* if IA is corrupted, not available otherwise); it might want to write to, read from Reg (through oracles *WReg*, *RReg*); or $\mathcal{A}$ might corrupt IA (through oracle *CrptIA*). Obviously $\mathcal{A}$ should not be allowed to corrupt OA.

A group signature GS=(Setup, Join, GSig, GVer, Open, Judge) is anonymous if the probability for any polynomial time adversary to win is negligible, i.e., the value of $\mathsf{Adv}^{anon}_{GS,\mathcal{A}}$ defined below is negligible.

$$\mathsf{Adv}^{anon}_{GS,\mathcal{A}}(k) = \Pr\{\mathsf{Exp}^{anon-1}_{GS,\mathcal{A}}(k) = 1\} - \Pr\{\mathsf{Exp}^{anon-0}_{GS,\mathcal{A}}(k) = 1\},$$

where experiments $\mathsf{Exp}^{anon-b}_{GS,\mathcal{A}}(k)$ are defined as in the above description.

If $\{i_0, i_1\} \subseteq HU$, and *CrptIA* is not queried, the group signature is *selfless anonymous* [BS04].

If $\{i_0, i_1\} \subseteq CU$, and *CrptIA* is not queried, the group signature is *anonymous* in the sense of [KY04].

If $\{i_0, i_1\} \subseteq HU$, and *CrptIA* is queried, the group signature is *anonymous* in the sense of [BSZ05].

We define a group signature GS is *anonymous* if $\{i_0, i_1\} \subseteq CU$ and *CrptIA* is queried in the above game, (in this case *GSig* is implied if *CrptIA* is queried), i.e., the corresponding experiments are defined as in Table 5.

Experiment $\mathsf{Exp}^{anon-b}_{GS,A}(k)$, $b \in \{0,1\}$
$(gpk, ik, ok) \xleftarrow{\$} \text{Setup}(1^k)$; $CU \leftarrow \varnothing$, $HU \leftarrow \varnothing$, $Chlist \leftarrow \varnothing$;
$d \xleftarrow{\$} A(gpk\text{: } CrptIA,\ Open,\ SndToU,\ USK,\ Ch(b,.,.,.),\ WReg)$,
Return d.

Table 5: Anonymity.

Traceability. Imagine a polynomial time adversary $\mathcal{A}$, whose goal is to produce a valid group signature (m, σ), the output of Open on which points to a non-existent member or an existing corrupted member but can not pass Judge.

Naturally the adversary $\mathcal{A}$ might corrupt some group members by running Join with IA (through oracles *CrptU* and *SndToI*); it might want to see some outputs of OA (through oracle *Open*); it might want to read from (through oracles *RReg*); or $\mathcal{A}$ might corrupt OA directly (through oracle $CrptOA$). Obviously $\mathcal{A}$ should not be allowed to corrupt IA and query *WReg*. Note that $\mathcal{A}$ might not bother to query about honest group members for they are of little help for it.

A group signature GS is traceable if the probability for any polynomial time adversary to win is negligible, i.e., the value of $\mathsf{Adv}^{trace}_{GS,\mathcal{A}}$ defined below is negligible.

$$\mathsf{Adv}^{trace}_{GS,\mathcal{A}}(k) = \mathsf{Pr}\{\mathsf{Exp}^{trace}_{GS,\mathcal{A}}(k) = 1\},$$

where experiment $\mathsf{Exp}^{trace}_{GS,\mathcal{A}}(k)$ is defined as in the above description.

If *CrptOA* is not queried, the group signature is secure against *misidentification attack* [KY04].

If *CrptOA* is queried, the group signature is *traceable* in the sense of [BSZ05].

We define a group signature GS is *traceable* if *CrptOA* is queried in the above game, i.e., the corresponding experiment is defined as in Table 6.

Experiment $\mathsf{Exp}^{trace}_{GS,A}(k)$
$(gpk, ik, ok) \xleftarrow{\$} \text{Setup}(1^k)$; $CU \leftarrow \varnothing$, $HU \leftarrow \varnothing$;
$(m, \sigma) \xleftarrow{\$} A(gpk : CrptOA,\ CrptU,\ SndToI,\ RReg)$.
If $\text{GVer}(gpk, m, \sigma) = 0$, return 0,else $(i, \tau) \leftarrow \text{Open}(gpk, ok, Reg, m, \sigma)$.
If $i = 0$ or $(\text{Judge}(gpk, reg, m, \sigma, \tau) = 0$ and $i \in CU)$ then return 1, else return 0.

Table 6: Traceability.

Non-frameability. Imagine a polynomial time adversary $\mathcal{A}$, whose goal is to produce a valid group signature (m, σ), the output of Open on which points to an existing honest member i_h and the result passes Judge.

Naturally the adversary $\mathcal{A}$ might want to get the group signing keys of some group members (through oracle USK); it might want to obtain some group signatures signed by some honest group members (through oracle *GSig*); it might want to see some outputs of OA (through oracle *Open*); it might also try to corrupt some group members by running Join with IA (through oracles *CrptU* and *SndToI*); it might observe the communication of some honest members joining in (through *SndToU* if *CrptIA* is queried, not available otherwise); it might wait until more group members has joined in (through $AddU$); it might want to write to, read from, Reg (through oracles *WReg*, *RReg*); or $\mathcal{A}$ might corrupt OA or IA directly (through oracle *CrptOA* and *CrptIA*). Obviously $\mathcal{A}$ should not be allowed to query *CrptU* (i_h), *SndToI* (i_h,.), *USK* (i_h).

A group signature GS is non-frameable if the probability for any polynomial time adversary to win is negligible, i.e., the value of $\mathsf{Adv}^{nf}_{GS,\mathcal{A}}$ defined below is negligible.

$$\mathsf{Adv}^{nf}_{GS,\mathcal{A}}(k) = \mathsf{Pr}\{\mathsf{Exp}^{nf}_{GS,\mathcal{A}}(k) = 1\},$$

where experiment $\mathsf{Exp}^{nf}_{GS,\mathcal{A}}(k)$ is defined as in the above description.

If *CrptIA* and *CrptOA* are queried, the group signature is secure against *framing attack* [KY04] or non-frameable [BSZ05].

We define a group signature GS is *non-frameable* if *CrptIA*, *CrptOA* are queried in the above game, and the corresponding experiment is defined as in Table 7.

Experiment $\mathsf{Exp}^{nf}_{GS,A}(k)$
$(gpk, ik, ok) \overset{\$}{\leftarrow}$ Setup(1^k); $CU \leftarrow \varnothing$, $HU \leftarrow \varnothing$, $GSet \leftarrow \varnothing$;
$(m, \sigma, i, \tau) \overset{\$}{\leftarrow} A(gpk : CrptIA,\ CrptOA,\ SndToU,\ GSig,\ USK,\ WReg)$.
If GVer$(gpk, m, \sigma) = 0$, return 0.
Else if $i \in HU$ and Judge$(gpk, reg, m, \sigma, \tau) = 1$
and $(i, m, .) \notin GSet$, return 1, else return 0.

Table 7: Non-frameability.

Definition 10. *A group signature scheme is secure if it is anonymous, traceable and non-frameable.*

C Security Proofs of the Generic Construction

C.1 Proof of Lemma 4.1

Note that the difference between our construction 4 and the generic construction in [BSZ05] is that, our ultimate group signature is $\sigma = (C, \Xi', \pi_1) = (Enc(pk_e, pk_i, r_i),\ \Xi',\ \pi_1)$, where π_1 is a proof of knowledge of (sk_i, Υ', r_i) satisfying $Ver(pk_s, sk_i, (\Upsilon', \Xi')) = 1$ and $C = Enc(pk_e, f(sk_i), r_i)$; while the ultimate group signature of [BSZ05] is $\sigma = (C, \pi_1) = (Enc(pk_e, < i, pk_i, \Upsilon, \Xi, s >, r_i),\ \pi_1)$, where $s = S(sk_i, m)$ and π_1 is a proof of knowledge of $(pk_i, \Upsilon, \Gamma, s, r_i)$ satisfying $Ver(pk_s, < i, pk_i >, (\Upsilon, \Xi)) = 1$, $C = Enc(pk_e, < i, pk_i, \Upsilon, \Xi, s >, r_i)$, and $V(pk_i, m, s) = 1$. (S, V) is the signature generation and verification algorithms of an independent signature scheme.

So we have more information to expose than [BSZ05], i.e., Ξ', because the signature we adopted is *perfectly unlinkable*, it does not affect the anonymity of the generated group signature at all. Then we can follow the proof of [BSZ05].

The proof follows [BSZ05]. Suppose $\mathcal{B}$ is an adversary to anonymity of GS, it can be invoked to construct an adversary $\mathcal{A}_c, c \in \{0, 1\}$ to the public encryption scheme PE, an adversary $\mathcal{A}_s$ to simulation soundness of (P_1, V_1), adversaries $\mathcal{D}_1$ and $\mathcal{D}_2$ to zero-knowledge of P_1 and P_2 respectively, these adversaries will answer the oracle queries from $\mathcal{B}$.

Description of $\mathcal{A}_c$. $\mathcal{A}_c$ is given the public key pk_e and accesses to oracles $Ch_{PE}(b, ., .)$ and $Dec(sk_e, .)$.

$\mathcal{A}_c$ selects keys (pk_s, sk_s) for DS, chooses common reference strings (R, R_1, R_2) for proofs P, SIM_1, SIM_2. $\mathcal{A}_c$ gives $gpk = (pk_e, pk_s, R, R_1, R_2)$ to $\mathcal{B}$. $\mathcal{A}_c$ answers oracle queries from $\mathcal{B}$ as follows:

CrptIA: returns sk_s.

Open (m, σ): If $(m, \sigma) = (m, C, \Xi', \pi_1)$ is valid and C is not returned by $Ch(c, ., .)$, queries oracle $Dec(sk_e, .)$, and generates a simulation proof for ρ_2.

SndToU (i,.): Runs algorithm Join, adds i to the honest member set HU.

USK (i): Returns $(pk_i, sk_i, \Upsilon, \Xi)$, deletes i from HU and adds i to the corrupted member set CU.

$Ch(c, i_0, i_1)$: If i_0, i_1 are existing members, runs algorithm GSig on input (gpk, gsk_{i_c}, m) except that the encryption is replaced by the response from a query to $Ch_{PE}(b, M_0, M_1)$ $(M_c = (pk_{i_c}),\ M_{\overline{c}} = (0^{|M_c|}))$, and the proof for ρ_1 is replaced by SIM_1.

WReg $(i,\ s)$: If i is a new member, sets $reg_i = s$.

$\mathcal{A}_c$ outputs what $\mathcal{B}$ outputs unless $\mathcal{B}$ has generated a new group signature $(m, \hat{\sigma})$=$(m, C, \hat{\Xi}, \hat{\pi})$ from the challenge (m, σ)=(m, C, Ξ', π_1), in which case $\mathcal{A}_c$ outputs c.

Description of $\mathcal{A}_s$. $\mathcal{A}_s$ is given the common reference string R_1 of SIM_1 and access to oracle SIM_1.

$\mathcal{A}_s$ setups GS as in algorithm Setup except that P_2 is replaced by its simulation SIM_2.

$\mathcal{A}_s$ gives $gpk = (pk_e, pk_s, R, R_1, R_2)$ to $\mathcal{B}$. $\mathcal{A}_s$ answers oracle queries from $\mathcal{B}$ as follows:

CrptIA: returns sk_s.

Open $(m,\ \sigma)$: If $(m, \sigma) = (m, C, \Xi', \pi_1)$, is valid and C is not returned by $Ch(b, ., .)$, runs algorithm Open since $\mathcal{A}_s$ knows $ok(= sk_e)$, and generates a simulation proof for ρ_2.

SndToU $(i,.)$: Runs as algorithm Join, adds i to the honest member set HU.

USK (i): Returns $(pk_i, sk_i, \Upsilon, \Xi)$, deletes i from HU and adds i to the corrupted member set CU.

$Ch(b, i_0, i_1)$: If i_0, i_1 are existing members, runs algorithm GSig on input (gpk, gsk_{i_1}, m) except that always encrypts $M_0 = (0^{|pk_1|})$ no matter the value of b, and the proof for ρ_1 is replaced by the response from a query to SIM_1, returns (C, Ξ', π_1).

WReg $(i,\ s)$: If i is a new member, sets $reg_i = s$.

$\mathcal{A}_s$ fails unless $\mathcal{B}$ has generated a new group signature $(m, \hat{\sigma}) = (m, C, \hat{\Xi}, \hat{\pi})$ from the challenge $(m, \sigma) = (m, C, \Xi', \pi_1)$, in which case $\mathcal{A}_s$ outputs $(pk_e, pk_s, m, C, \hat{\Xi})$ and $\hat{\pi}$.

Description of $\mathcal{D}_1$. $\mathcal{D}_1$ is given the common reference string R_1, and access to oracle $Prove_1(.)$ which may be P_1 or SIM_1.

$\mathcal{D}_1$ setups GS as in algorithm Setup except that P_2 is replaced by a simulation SIM_2.

$\mathcal{D}_1$ gives $gpk = (pk_e, pk_s, R, R_1, R_2)$ to $\mathcal{B}$ and answers oracle queries from $\mathcal{B}$ as follows:

CrptIA: returns sk_s.

Open $(m,\ \sigma)$: If (m, σ) is valid, runs algorithm Open since $\mathcal{D}_1$ knows $ok(= sk_e)$, and generates a simulation proof for ρ_2.

SndToU $(i,.)$: Runs as algorithm Join, adds i to the honest member set HU.

USK (i): Returns $(pk_i, sk_i, \Upsilon, \Xi)$, deletes i from HU and adds i to the corrupted member set CU.

$Ch(b, i_0, i_1)$: If i_0, i_1 are existing members, runs algorithm GSig on input (gpk, gsk_{i_b}, m) except that generates π_1 by querying oracle $Prove_1$.

WReg $(i,\ s)$: If i is a new member, sets $reg_i = s$.

$\mathcal{D}_1$ returns 1 if output of $\mathcal{B}$ equals b, returns 0 otherwise.

Description of $\mathcal{D}_2$. $\mathcal{D}_2$ is given the common reference string R_2, and access to oracle $Prove_2(.)$ which may be P_2 or SIM_2.

$\mathcal{D}_2$ setups GS as in algorithm Setup.

$\mathcal{D}_2$ gives $gpk = (pk_e, pk_s, R, R_1, R_2)$ to $\mathcal{B}$ and answers oracle queries from $\mathcal{B}$ as follows:

CrptIA: returns sk_s.

Open $(m,\ \sigma)$: If (m, σ) is valid, runs algorithm Open since $\mathcal{D}_2$ knows $ok(= sk_e)$, and generates the proof for ρ_2 by querying oracle $Prove_2$.

SndToU $(i,.)$: Runs as algorithm Join, adds i to the honest member set HU.

USK (i): Returns $(pk_i, sk_i, \Upsilon, \Xi)$, deletes i from HU and adds i to the corrupted member set CU.

$Ch(b, i_0, i_1)$: If i_0, i_1 are existing members, runs algorithm GSig on input (gpk, gsk_{i_b}, m).

WReg $(i,\ s)$: If i is a new member, sets $reg_i = s$.

$\mathcal{D}_2$ returns 1 if output of $\mathcal{B}$ equals b, returns 0 otherwise.

It follows from the same analysis in [BSZ05] that

$$\begin{aligned}\mathsf{Adv}^{anon}_{GS,\mathcal{B}}(k) \leq &\mathsf{Adv}^{ind-cca}_{PE,\mathcal{A}_0}(k) + \mathsf{Adv}^{ind-cca}_{PE,\mathcal{A}_1}(k) + \mathsf{Adv}^{ss}_{SIM_1,\mathcal{A}_s}(k) \\ &+ 2(\mathsf{Adv}^{zk}_{P_1,SIM_1,\mathcal{D}_1}(k) + \mathsf{Adv}^{zk}_{P_2,SIM_2,\mathcal{D}_2}(k)).\end{aligned}$$

C.2 Proof of Lemma 4.2

The proof follows [BSZ05]. Suppose $\mathcal{B}$ is an adversary to traceability of GS, it can be invoked to construct an adversary $\mathcal{A}_{ds}$ to the digital signature scheme *DS*, the adversary will answer the oracle queries from $\mathcal{B}$.

Description of $\mathcal{A}_{ds}$. $\mathcal{A}_{ds}$ is given the public key pk_s and access to oracle $Sig(sk_s, .)$.

$\mathcal{A}_{ds}$ selects keys (pk_e, sk_e) for PE, chooses common reference strings R, R_1, R_2 for relation ρ, ρ_1 and ρ_2 respectively. $\mathcal{A}_{ds}$ gives $gpk = (pk_e, pk_s, R, R_1, R_2)$ to $\mathcal{B}$. $\mathcal{A}_{ds}$ answers oracle queries from $\mathcal{B}$ as follows:

CrptOA: returns sk_e.

CrptU (i): If i is not a group member yet, adds i to the corrupted members set CU.

SndToI $(i,.)$: Parses the input into (pk_i, π) from which extracts sk_i using the online extractor algorithm K of (P, V) by manipulating the random oracle, queries oracle $Sig(sk_s, sk_i)$.

RReg (i): If i exists in Reg, returns reg_i.

If $\mathcal{B}$ wins with non-negligible probability, i.e., outputs a valid group signature $(m, \sigma) = (m, C, \Xi', \pi_1)$ and $i = 0$, where $(i, \tau) \leftarrow Open(sk_e, m, \sigma)$. Another case that $i > 0$ will not occur because of the correctness of GS and the assumptions for GS in our model (Appendix B).

From generalized forking lemma [KY04], (GVer be the predicate), in random oracle model, there exist (m, C, Ξ', c, s), (m, C, Ξ', c', s') from which (w, Υ', r) can be extracted, (Υ', Ξ') is a valid DS signature on w, and w is not queried to $Sig(sk_s, .)$.

It follows from the same analysis in [BSZ05] that

$$\mathsf{Adv}^{trace}_{GS,\mathcal{B}}(k) \leq 2^{-k} + \mathsf{Adv}^{wUF-acma}_{DS,\mathcal{A}_{ds}}(k).$$

C.3 Proof of Lemma 4.3

The proof follows [BSZ05]. Suppose $\mathcal{B}$ is an adversary to non-frameability of GS, it can be invoked to construct an adversary $\mathcal{A}_f$ to the one way function f, the adversary will answer the oracle queries from $\mathcal{B}$.

Description of $\mathcal{A}_f$. $\mathcal{A}_f$ is given y in the range of the one way function f.

$\mathcal{A}_f$ sets up GS as in algorithm Setup, selects a random variable $\iota \in [1, n(k)]$, $n(k)$ is the maximum number of queries from $\mathcal{B}$.

$\mathcal{A}_f$ gives $gpk = (pk_e, pk_s, R, R_1, R_2)$ to $\mathcal{B}$ and answers oracle queries from $\mathcal{B}$ as follows:

CrptIA: returns sk_s.

CrptOA: returns sk_e.

SndToU $(i,.)$: If $i = \iota$, sets $pk_i = y$, and runs Join by simulating a proof for relation ρ; otherwise runs exactly as algorithm Join. Then adds i to the honest member set HU.

USK (i): If $i = \iota$, $\mathcal{A}_f$ stops and restarts again; otherwise if $i \in HU$, returns $(pk_i, sk_i, \Upsilon, \Xi)$, deletes i from HU and adds i to the corrupted member set CU.

GSig (i, m): If $i \in HU$ and $i = \iota$, runs algorithm GSig except that replacing proof P_1 by the simulation SIM_1; otherwise if $i \in HU$, runs GSig exactly. $GSet \leftarrow GSet \cup \{(i, m, \sigma)\}$.

WReg (i, s): If i is a new member, sets $reg_i = s$.

$\mathcal{A}_f$ returns 1 if $\mathcal{B}$ outputs a valid group signature that $(\iota, m, \sigma) \notin GSet$ and $Judge(gpk, reg, m, \sigma, \tau) = 1$ where $(\iota, \tau) = Open(m, \sigma)$.

Parse (ι, m, σ) into $(\iota, m, C, \Xi', c, s)$, then there exist $(\iota, m, C, \Xi', c, s)$, (m, C, Ξ', c', s') in random oracle model according to generalized forking lemma [KY04], (GVer be the predicate), so (w, Υ', r) can be extracted, where (Υ', Ξ') is a valid DS signature on w, and $f(w) = y$.

It follows from a similar analysis in [BSZ05] that $\mathsf{Adv}^{nf}_{GS,\mathcal{B}}(k) \leq \epsilon(k) + n(k)\mathsf{Adv}^{ow}_{f,\mathcal{A}_f}(k)$, where $\epsilon(k)$ is negligible.