The preliminary version of this paper appeared in the proceedings of *"The Ninth International Symposium on Computing and Networking, CANDAR 2021"* under the title "Group Signatures with Designated Traceability". This is the full version.

Group Signatures with Designated Traceability over Openers' Attributes ⋆

Hiroaki Anada[1], Masayuki Fukumitsu[2], and Shingo Hasegawa[3]

[1] Department of Mathematical Informatics, Meiji Gakuin University, Yokohama, Japan
hiroaki.anada@mi.meijigakuin.ac.jp

[2] Department of Information Security, University of Nagasaki, Nagayo, Japan
fukumitsu@sun.ac.jp

[3] Faculty of Symbiotic Systems Science, Fukushima University, Fukushima, Japan
hasegawa@sss.fukushima-u.ac.jp

October 25, 2024

Abstract. We propose a group signature scheme with a function of designated traceability; each opener has attributes, and a signer of a group signature can be traced by only the openers whose attributes satisfy the boolean formula designated by the signer. We describe syntax and security definitions of the scheme. Then we give a generic construction of the scheme by employing a ciphertext-policy attribute-based encryption scheme.

Keywords: group signatures · anonymity · traceability · accountability · openers · attribute-based.

1 Introduction

A group signature scheme proposed by Chaum and van Heyst [9] enables a signer to sign a message on behalf of a group to which he/she belongs. The signature is anonymous [9] in the sense that the signer is not identified in the group. Nonetheless, the scheme is traceable [1] by an authority called an opener who can identify the signer by using an opening key. So far group signature schemes with some characteristics and properties were proposed ([5,16,14], etc.). Also, rigorous foundations of security were proposed for the cases of static, partially dynamic and fully dynamic groups [3,4,7]. Especially, an authority called an issuer was introduced separately from an opener by Bellare et al. [4].

One of the view points on traceability is that it is excessive; an opener is able to open all the signatures. One direction to pursue the problem is "message-dependent opening" [19,17,10]. In a group signature scheme with message-dependent opening, there is an authority called an admitter who admits the opener to open signatures by specifying messages. That is, the admitter issues a token that corresponds to a message, and then the opener extracts the signer's identity from the signature using the token. Another direction is "accountable tracing" [15,12]. In an accountable tracing group signature scheme, users in a group are divided into two kinds. One is a kind of users who can be traced and the other is a kind of users who cannot be traced. A user is given a group-signing key by the issuer, where the key belongs to either the former kind or the latter. However, in the both schemes users themselves do not have the right to actively specify limitation on the opening function. As a remarkable work, Xu and Yung [20] introduced "accountable ring signatures" (ARS), in which an anonymous signer can designate an opener by indicating the opener's public key. Bootle et al. [8] described an efficient scheme of ARS based on the DDH assumption.

⋆ This work was supported by The Telecommunication Advancement Foundation (TAF).

1.1 Our Contribution

In this paper, we introduce a function of *designated traceability* in a group signature scheme, which concerns with users' right on the opening function. In our scheme there are more than one opener, and an opener has a set of attributes over all the possible attributes. Each attribute corresponds to a component of a public key, and the public key is maintained by the group manager. An opening key is issued to an opener by the group manager depending on the opener's attributes. When a user signs a message on behalf of a group, he/she can specify an *access structure* over the attributes, and generate a signature that has the access structure to his/her hidden identity. When an opener tries to open a group signature to identify the signer, the opener uses its opening key, but the signature can be opened to disclose the identity if and only if the attached access structure is satisfied by the attributes described in the opening key. Hence only the designated openers can open signatures. In this sense, our direction is an enhancement of the function of ARS [20,8].

In a realistic scenario, our designated traceability can be used as follows. Suppose that there is a company which has a chief information officer (CIO) and a number of departments each of which is under a head person. These head persons are enrolled as "openers" by CIO, while CIO itself is the "group manager". When an employee generates a group signature on behalf of the company, he/she designates the head of his/her department, or more freely, "the head or the heads of related departments", as his/her choice. More concretely, the designation is by means of specifying an access structure over the attributes that are maintained by CIO as components of a public key. Thus, only the opener(s) whose attributes satisfy the access structure is able to open and trace the signer, if it is needed. We note that, since an access structure Y is visibly attached to a generated signature σ_0, an opener, receiving the pair $\sigma = (Y, \sigma_0)$, sees whether the opener can open it or not. We also note that the group manager (CIO) should be able to open all the signatures by using the master secret key.

1.2 Outline of Our Construction and Security Proofs

After giving syntax and security definitions of our scheme, we give a generic construction of our scheme by modifying the construction of a partially dynamic group signature scheme proposed by Bellare et al. [4]. Our core idea is to replace a public-key encryption scheme (PKE), which is one of the building blocks in [4], with a (only-payload-hiding) ciphertext-policy attribute-based encryption scheme (CP-ABE). Other building blocks are a digital signature scheme (SIG) and a simulation-sound non-interactive zero-knowledge proof system (SS-NIZK), as is the same as the construction in [4]. The setup algorithm of CP-ABE is executed by the group manager to generate a set of public parameters, a public key and the master secret key. Also, the key-generation algorithm of CP-ABE is executed by the group manager to issue a secret key to an opener. That is, an opener joins dynamically. Then, when a user wants to join a group as a member, he/she generates a pair of a public key and a secret key of SIG in advance of joining. After that it executes a joining protocol with the issuer in the same way as [4]. When a user wants to generate a group signature, he/she first generates a signature s by using another signing key in joining the protocol. Then, specifying an access structure X, he/she encrypts s (and the identity data and certificate) by the encryption algorithm of CP-ABE. When one of the openers try to trace the signer of a group signature, it first decrypts the ABE ciphertext. This is executable if and only if the set of attributes X of the opener's secret key satisfies the ciphertext policy Y in the signature (i.e. $\mathcal{R}(X, Y) = 1$ for the relation R of ABE).

As for security proofs, only anonymity is affected by the replacement of PKE in [4] with ABE; traceability and non-frameability are not affected. To prove anonymity according to under a suitably modified definition of the experiment (in [4]), we have to introduce an "Add-an-opener oracle" and a "Corrupt-an-opener oracle". (See Section 3.2.)

2 Preliminaries

In this section, we fix our notation. Also, we survey the needed notions for the later sections.

The set of natural numbers is denoted by $\mathbb{N}$. The security parameter is denoted by λ, where $\lambda \in \mathbb{N}$. The bit length of a string s is denoted by $|s|$. A uniform random sampling of an element a from a set S is denoted as $a \leftarrow_R S$. When an algorithm A on input a outputs z, we denote it as $z \leftarrow A(a)$, or, $A(a) \rightarrow z$. When a probabilistic algorithm A on input a and with randomness r returns z, we denote it as $z \leftarrow A(a; r)$. St is the inner state of a particular algorithm. PPT means "probabilistic polynomial time". When an algorithm A on input a accesses an oracle O, we denote it as $A(a : \mathsf{O})$. A function $P(\lambda)$ is said to be negligible in λ if for any given positive polynomial $\mathrm{poly}(\lambda)$ $P(\lambda) < 1/\mathrm{poly}(\lambda)$ for sufficiently large λ.

2.1 Digital Signature ([11])

A digital signature scheme Sig consists of three PPT algorithms, KG, Sign and Vrfy. (If needed, we put a subscript $_\mathsf{s}$.)
• $\mathsf{KG}(1^\lambda) \to (pk, sk)$. This PPT algorithm takes as input the security parameter 1^λ. It returns a verification key pk and a signing key sk.
• $\mathsf{Sign}(sk, m) \to s$. This PPT algorithm takes as input a signing key sk and a message m. It returns a signature s.
• $\mathsf{Vrfy}(pk, m, s) \to d$. This deterministic polynomial-time algorithm takes as input a public key pk, a message m and a signature s. It returns a boolean value $d \in \{1, 0\}$.

Correctness of Sig is defined as follows; for any λ and any m, $\Pr[d = 1 \mid \mathsf{KG}(1^\lambda) \to (pk, sk); \mathsf{Sign}(sk, m) \to s; \mathsf{Vrfy}(pk, m, s) \to d] = 1$.

Existential unforgeability against chosen-message attacks of Sig is captured by the following experiment, where $\mathbf{A}$ is an algorithm.

$\mathsf{Expr}^{\text{euf-cma}}_{\mathsf{Sig},\mathbf{A}}(1^\lambda)$
$(pk, sk) \leftarrow \mathsf{KG}(1^\lambda); (m^*, s^*) \leftarrow \mathbf{A}(pk : \mathsf{SignO}(sk, \cdot))$
If $\mathsf{Vrfy}(pk, m^*, s^*) = 1$ and m^* was not queried
then return 1 else return 0

Here $\mathsf{SignO}(sk, m)$ returns $s \leftarrow \mathsf{Sign}(sk, m)$. m^* must not be a message that was queried to SignO. The advantage of $\mathbf{A}$ over Sig is defined by

$$\mathbf{Adv}^{\text{euf-cma}}_{\mathsf{Sig},\mathbf{A}}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathsf{Expr}^{\text{euf-cma}}_{\mathsf{Sig},\mathbf{A}}(1^\lambda) = 1]. \quad (1)$$

A digital signature scheme Sig is said to be EUF-CMA secure if, for any PPT $\mathbf{A}$, $\mathbf{Adv}^{\text{euf-cma}}_{\mathsf{Sig},\mathbf{A}}(\lambda)$ is negligible in λ.

2.2 Attribute-Based Encryption ([18,13,6,2])

An attribute-based encryption scheme ABE consists of four PPT algorithms, Setup, KG, Enc and Dec, and a function $\mathcal{R}^\kappa$. (If needed, we put a subscript $_\mathsf{a}$.)
• κ. This is an index s.t., for a constant c, $\kappa \in \mathbb{N}^c$. It indicates authorized attribute sets and a predicate function (below).
• $\mathbb{X}^\kappa$. This is the set of all key attributes.
• $\mathbb{Y}^\kappa$. This is the set of all ciphertext attributes.
• $\mathcal{R}^\kappa : \mathbb{X}^\kappa \times \mathbb{Y}^\kappa \to \{0, 1\}$. A predicate function on $\mathbb{X}^\kappa \times \mathbb{Y}^\kappa$, which determines a relation (i.e. a subset $\{(X, Y) \in \mathbb{X}^\kappa \times \mathbb{Y}^\kappa \mid \mathcal{R}^\kappa(X, Y) = 1\}$).
• $\mathsf{Setup}(1^\lambda, \kappa) \to (pk, msk)$. This PPT algorithm takes as input the security parameter 1^λ and the attribute index $\kappa \in \mathbb{N}^c$. It returns a public key pk and a master secret key msk.
• $\mathsf{KG}(msk, i, X) \to sk^i_X$. This PPT algorithm takes as input the master secret key msk, an identity index i and a key attribute X. It returns a secret key sk^i_X.
• $\mathsf{Enc}(pk, Y, M) \to C$. This PPT algorithm takes as input the public key pk, a ciphertext attribute Y and a plaintext M. It returns a ciphertext C. We assume that Enc is *only-payload-hiding*, that is, C can be parsed as (Y, C_0).
• $\mathsf{Dec}(pk, sk^i_X, C) \to \hat{M}$. This deterministic polynomial-time algorithm takes as input a secret key sk^i_X and a ciphertext C. It returns a decryption result $\hat{M}$.

Correctness of ABE is defined as follows; for any λ, any κ, any M, any i, any X and Y s.t. $\mathcal{R}^\kappa(X, Y) = 1$, $\Pr[M = \hat{M} \mid \mathsf{Setup}(1^\lambda, \kappa) \to (pk, msk); \mathsf{KG}(msk, i, X) \to sk^i_X; \mathsf{Enc}(pk, Y, M) \to C; \mathsf{Dec}(pk, sk^i_X, C) \to \hat{M}] = 1$;.

Indistinguishability against chosen-plaintext attack of ABE is captured by the following experiment, where $\mathbf{A}$ is an algorithm.

$$\begin{aligned}&\mathsf{Expr}^{\text{ind-cpa-}b}_{\mathsf{ABE},\mathbf{A}}(1^\lambda,\kappa)\\&\quad(pk,msk)\leftarrow\mathsf{Setup}(1^\lambda,\kappa)\\&\quad d\leftarrow\mathbf{A}(pk:\mathsf{KGO}(msk,\cdot,\cdot),\mathsf{LRO}_b(pk,\cdot,\cdot,\cdot))\\&\quad\text{Return } d\end{aligned}$$

Here $\mathsf{KGO}(msk,i,X)$ returns $sk^i_X \leftarrow \mathsf{KG}(msk,i,X)$, and $\mathsf{LRO}_b(pk,M_0,M_1,Y^*)$ returns $C^* \leftarrow \mathsf{Enc}(pk,Y^*,M_b)$. The challenge query (M_0,M_1,Y^*) must satisfy $|M_0|=|M_1|$ and $\mathcal{R}^\kappa(X,Y^*)\neq 1$ for all queried X to KGO. Y^* is called the target attribute. After accessing LRO_b, X cannot be queried to KGO if $\mathcal{R}^\kappa(X,Y^*)=1$. The advantage of $\mathbf{A}$ over ABE is defined by

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{ABE},\mathbf{A}}(\lambda)\stackrel{\text{def}}{=}|\Pr[\mathsf{Expr}^{\text{ind-cpa-}1}_{\mathsf{ABE},\mathbf{A}}(1^\lambda,\kappa)=1]-\Pr[\mathsf{Expr}^{\text{ind-cpa-}0}_{\mathsf{ABE},\mathbf{A}}(1^\lambda,\kappa)=1]|. \quad (2)$$

An attribute-based encryption scheme ABE is said to be adaptive IND-CPA secure if, for any PPT $\mathbf{A}$, $\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{ABE},\mathbf{A}}(\lambda)$ is negligible in λ.

An ABE scheme is called "ciphertext policy" if $\mathbb{X}$ is the set of all subsets of attributes and $\mathbb{Y}$ is the set of all access structures over the attributes [18,13,6,2].

2.3 Simulation-Sound Non-interactive Zero-Knowledge Proof (Argument) ([4], Section 5.1)

A simulation-sound non-interactive zero-knowledge proof system Π consists of two algorithms, P and V. We consider in this paper that not only V but also P are polynomial time (i.e. an argument system). We also assume that P is probabilistic and V is deterministic. An NP-relation over domain $\mathrm{Dom}\subseteq\{0,1\}^*$ is a subset ρ of $\{0,1\}^*\times\{0,1\}^*$ such that membership of (x,w) is decidable in time polynomial in $|x|$, $\forall x\in\mathrm{Dom}$. The language L_ρ is defined by $L_\rho\stackrel{\text{def}}{=}\{x\in\mathrm{Dom}\mid\exists w\in\{0,1\}^*\ (x,w)\in\rho\}$. P and V have access to a common reference string R. There exist two polynomials ℓ and p s.t. the following two properties hold;

• Completeness.

$$\begin{aligned}&\forall\lambda\in\mathbb{N}\ \forall(x,w)\in\rho \text{ s.t. } |x|\leq\ell(\lambda) \text{ and } x\in\mathrm{Dom}\\&\Pr[R\leftarrow_R\{0,1\}^{p(\lambda)};\pi\leftarrow\mathsf{P}(1^\lambda,x,w,R):\mathsf{V}(1^\lambda,x,\pi,R)=1]\\&\quad=1.\end{aligned}$$

• Soundness.

$$\begin{aligned}&\forall\lambda\in\mathbb{N}\ \forall\hat{\mathsf{P}}:\text{PPT}\ \forall x\in\mathrm{Dom} \text{ s.t. } x\notin L_\rho\\&\Pr[R\leftarrow\{0,1\}^{p(\lambda)};\pi\leftarrow\hat{\mathsf{P}}(1^\lambda,x,R):\mathsf{V}(1^\lambda,x,\pi,R)=1]\\&\quad<2^{-\lambda}.\end{aligned}$$

Further, we introduce the third property.

• Zero-Knowledge. For Π there exists a PPT algorithm Sim called a simulator. We consider the following experiment, where $\mathbf{D}$ is an algorithm.

$$\begin{aligned}&\mathsf{Exp}^{\text{zk-0}}_{\mathsf{P},\mathsf{Sim},\mathbf{D}}(1^\lambda)\\&\quad(R,St)\leftarrow\mathsf{Sim}(\mathtt{gen},1^\lambda);\ d\leftarrow\mathbf{D}(R:\mathsf{P}_1(\cdot,\cdot));\ \text{return } d\\&\mathsf{P}_1(x,w):\pi\leftarrow\mathsf{Sim}(\mathtt{prv},St,x);\ \text{return }\pi\\&\mathsf{Exp}^{\text{zk-1}}_{\mathsf{P},\mathsf{Sim},\mathbf{D}}(1^\lambda)\\&\quad R\leftarrow\{0,1\}^{p(\lambda)};\ d\leftarrow\mathbf{D}(R:\mathsf{P}_2(\cdot,\cdot));\ \text{return } d\\&\mathsf{P}_2(x,w):\pi\leftarrow\mathsf{P}(1^\lambda,x,w,R);\ \text{return }\pi\end{aligned}$$

The advantage of $\mathbf{D}$ over Π is defined by

$$\mathbf{Adv}^{\mathrm{zk}}_{\mathsf{P},\mathsf{Sim},\mathbf{D}}(\lambda) \stackrel{\mathrm{def}}{=} |\Pr[\mathsf{Exp}^{\mathrm{zk}\text{-}0}_{\mathsf{P},\mathsf{Sim},\mathbf{D}}(1^\lambda)=1] - \Pr[\mathsf{Exp}^{\mathrm{zk}\text{-}1}_{\mathsf{P},\mathsf{Sim},\mathbf{D}}(1^\lambda)=1]|. \quad (3)$$

A non-interactive proof system Π is said to be computational zero-knowledge if, for any PPT $\mathbf{D}$, $\mathbf{Adv}^{\mathrm{zk}}_{\mathsf{P},\mathsf{Sim},\mathbf{D}}(\lambda)$ is negligible in λ.

Besides, we need in this paper;

• Simulation Soundness.

$\mathsf{Exp}^{\mathrm{ss}}_{\Pi,\mathbf{A}}(1^\lambda)$
$(R, St) \leftarrow \mathsf{Sim}(\mathtt{gen}, 1^\lambda)$; $(x, \pi) \leftarrow \mathbf{A}(R : \mathsf{Sim}(\mathtt{prv}, St, \cdot))$
If $x \notin L_\rho \wedge \pi$ was not given to $\mathbf{A} \wedge \mathsf{V}(1^\lambda, x, \pi, R) = 1$
then return 1 else return 0

The advantage of $\mathbf{A}$ over Π is defined by

$$\mathbf{Adv}^{\mathrm{ss}}_{\Pi,\mathbf{A}}(\lambda) \stackrel{\mathrm{def}}{=} \Pr[\mathsf{Exp}^{\mathrm{ss}}_{\Pi,\mathbf{A}}(1^\lambda)=1]. \quad (4)$$

A non-interactive proof system Π is said to be simulation sound if, for any PPT $\mathbf{A}$, $\mathbf{Adv}^{\mathrm{ss}}_{\Pi,\mathbf{A}}(\lambda)$ is negligible in λ.

3 Syntax and Security Definitions

In this section, we give syntax and security definitions of our proposed group signature scheme that has designated traceability, GSdT.

3.1 Syntax

The scheme GSdT consists of nine PPT algorithms; (GKG, OKG, UKG, Join, Iss, GSign, GVrfy, Open, Judge).

• $\mathsf{GKG}(1^\lambda, \kappa) \rightarrow (gpk, ik, omk)$. This PPT algorithm takes as input the security parameter 1^λ and the attribute index κ. It returns a group public key gpk, an issuing key ik and an opening master key omk.

• $\mathsf{OKG}(gpk, omk, j, X) \rightarrow \boldsymbol{ok}[j]$. This PPT algorithm takes as input gpk, omk, an opener's index j and an opener's attribute X. It returns an opening key $\boldsymbol{ok}[j]$. Note that $\boldsymbol{ok}[j]$ includes the data of X.

• $\mathsf{UKG}(1^\lambda) \rightarrow (upk, usk)$. This PPT algorithm takes as input 1^λ. It returns a user public key upk and a user secret key usk.

• Join and Iss. These interactive PPT algorithms Join and Iss are explained in Fig.1. (Since these are essentially the same as in [4], we omit the explanation.)

• $\mathsf{GSign}(gpk, \boldsymbol{gsk}[i], Y, m) \rightarrow (Y, \sigma_0)$. This PPT algorithm takes as input gpk, a group signing key $\boldsymbol{gsk}[i]$ (see Fig.1) of a member i, an access structure Y and a message m. It returns a group signature (Y, σ_0).

• $\mathsf{GVrfy}(gpk, m, (Y, \sigma_0)) \rightarrow d$. This deterministic polynomial-time algorithm takes as input gpk, m and (Y, σ_0). It returns a boolean value $d \in \{0, 1\}$.

• $\mathsf{Open}(gpk, \boldsymbol{ok}[j], \boldsymbol{reg}, m, (Y, \sigma_0)) \rightarrow (i, \tau)$. This PPT algorithm takes as input gpk, $\boldsymbol{ok}[j]$, the user registration table $\boldsymbol{reg}$, m and (Y, σ_0). It returns a user identity index $i \in \mathbb{N} \cup \{0\}$ and a proof τ. In the case $i \geq 1$, the algorithm is claiming that the group member with identity i produced (Y, σ_0), and in the case $i = 0$, it is claiming that no group member produced (Y, σ_0).

• $\mathsf{Judge}(gpk, i, \boldsymbol{upk}[i], m, (Y, \sigma_0), \tau) \rightarrow d$. This deterministic polynomial-time algorithm takes as input gpk, i, $\boldsymbol{upk}[i]$, m, (Y, σ_0) and a proof τ. It returns a boolean value $d \in \{1, 0\}$.

Remark. In the above, OKG takes as input an index j to generate the j-th entry $\boldsymbol{ok}[j]$. This is so that two openers having the same attribute X can be separated in *realistic use*. However, in theory, we can introduce another syntax without the index j.

$\text{User}^i(gpk, i, \boldsymbol{upk}[i], \boldsymbol{usk}[i])$		$\text{Issuer}^i(gpk, ik, i, \boldsymbol{upk}[i])$
$St_{join} = (gpk, i, \boldsymbol{upk}[i], \boldsymbol{usk}[i])$		$St_{iss} = (gpk, ik, i, \boldsymbol{upk}[i])$
$M_{in} = \varepsilon$		
$(St'_{join}, M_{out}, \mathtt{cont}) \leftarrow \mathsf{Join}(St_{join}, M_{in})$		
$St_{join} = St'_{join}$	$(M_{out}, \mathtt{cont})$ $\longrightarrow$	$\boldsymbol{reg}[i] = M_{in} = M_{out}, dec = \mathtt{cont}$
		$(St'_{iss}, M_{out}, dec') \leftarrow \mathsf{Iss}(St_{iss}, M_{in}, dec)$
	M_{out} $\longleftarrow$	$St_{iss} = St'_{iss}, dec = dec'$
$M_{in} = M_{out}$		
$(St'_{join}, \varepsilon, \mathsf{acc}) \leftarrow \mathsf{Join}(St_{join}, M_{in})$		
$\boldsymbol{gsk}[i] = St'_{join}$		

Fig. 1. Group joining protocol.

$\mathsf{AddOO}(j, X)$
  If $j \in \text{OP}$ then return ε
  $\text{OP} \leftarrow \text{OP} \cup \{j\}$; $\boldsymbol{ok}[j] \leftarrow \mathsf{OKG}(gpk, omk, j, X)$
  Return 1
$\mathsf{AddUO}(i)$
  If $i \in \text{HU} \cup \text{CU}$ then return ε
  $\text{HU} \leftarrow \text{HU} \cup \{i\}$; $dec^i \leftarrow \mathtt{cont}$; $\boldsymbol{gsk}[i] \leftarrow \varepsilon$
  $(\boldsymbol{upk}[i], \boldsymbol{usk}[i]) \leftarrow \mathsf{UKG}(1^\lambda)$
  $St^i_{join} \leftarrow (gpk, \boldsymbol{upk}[i], \boldsymbol{usk}[i])$
  $St^i_{iss} \leftarrow (gpk, ik, i, \boldsymbol{upk}[i])$; $M_{join} \leftarrow \varepsilon$
  $(St^i_{join}, M_{join}, dec^i) \leftarrow \mathsf{Join}(St^i_{join}, M_{join})$
  While $dec^i = \mathtt{cont}$ do
    $(St^i_{iss}, M_{join}, dec^i) \leftarrow \mathsf{Iss}(St^i_{iss}, M_{iss}, dec^i)$
    If $dec^i = \mathsf{acc}$ then $\boldsymbol{reg}[i] \leftarrow St^i_{iss}$
    $(St^i_{join}, M_{iss}, dec^i) \leftarrow \mathsf{Join}(St^i_{join}, M_{join})$
  $\boldsymbol{gsk}[i] \leftarrow St^i_{join}$
  Return $\boldsymbol{upk}[i]$
$\mathsf{StoUO}(i, M_{in})$
  If $i \notin \text{HU}$ then
    $\text{HU} \leftarrow \text{HU} \cup \{i\}$; $(\boldsymbol{upk}[i], \boldsymbol{usk}[i]) \leftarrow \mathsf{UKG}(1^\lambda)$
    $\boldsymbol{gsk}[i] \leftarrow \varepsilon$; $M_{in} \leftarrow \varepsilon$;
    $St^i_{join} \leftarrow (gpk, \boldsymbol{upk}[i], \boldsymbol{usk}[i])$
  $(St^i_{join}, M_{out}, dec) \leftarrow \mathsf{Join}(St^i_{join}, M_{in})$
  If $dec = \mathsf{acc}$ then $\boldsymbol{gsk}[i] \leftarrow St^i_{join}$
  Return (M_{out}, dec)
$\mathsf{USKO}(i)$ Return $(\boldsymbol{gsk}[i], \boldsymbol{usk}[i])$
$\mathsf{GSignO}(i, Y^*, m)$
  If $i \notin \text{HU}$ then return $\perp$
  If $\boldsymbol{gsk}[i] = \varepsilon$ then return $\perp$
  Else return $\mathsf{GSign}(gpk, \boldsymbol{gsk}[i], Y^*, m)$

$\mathsf{CrptOO}(j)$
  If $j \notin \text{OP}$ then return ε
  $(X, \bar{ok}) \leftarrow \boldsymbol{ok}[j]$
  If $\exists (m, (Y^*, \sigma_0)) \in \text{MS}$ s.t. $\mathcal{R}^\kappa(X, Y^*) = 1$
    then return ε
  $\text{CO} \leftarrow \text{CO} \cup \{j\}$
  Return $\boldsymbol{ok}[j]$
$\mathsf{CrptUO}(i, upk)$
  If $i \in \text{HU} \cup \text{CU}$ then return ε
  $\text{CU} \leftarrow \text{CU} \cup \{i\}$; $\boldsymbol{upk}[i] \leftarrow upk$; $dec^i \leftarrow \mathtt{cont}$
  $St^i_{iss} \leftarrow (gpk, ik, i, \boldsymbol{upk}[i])$
  Return 1
$\mathsf{StoIO}(i, M_{in})$
  If $i \notin \text{CU}$ then return ε
  $(St^i_{iss}, M_{out}, dec^i) \leftarrow \mathsf{Iss}(St^i_{iss}, M_{in}, dec^i)$
  If $dec^i = \mathsf{acc}$ then $\boldsymbol{reg}[i] \leftarrow St^i_{iss}$
  Return M_{out}
$\mathsf{OpenO}(j, m, (Y, \sigma_0))$
  If $(m, (Y, \sigma_0)) \in \text{MS}$ then return $\perp$
  Return $\mathsf{Open}(gpk, \boldsymbol{ok}[j], \boldsymbol{reg}, m, (Y, \sigma_0))$
$\mathsf{RRegO}(i)$ Return $\boldsymbol{reg}[i]$
$\mathsf{WRegO}(i, \rho)$ $\boldsymbol{reg}[i] \leftarrow \rho$; Return 1
$\mathsf{ChaO}_b(i_0, i_1, m, Y^*)$
  If $i_0 \notin \text{HU}$ or $i_1 \notin \text{HU}$ then return $\perp$
  If $\boldsymbol{gsk}[i_0] = \varepsilon$ or $\boldsymbol{gsk}[i_1] = \varepsilon$ then return $\perp$
  If $\exists j \in \text{CO}$ s.t.
    $\mathcal{R}^\kappa(X, Y^*) = 1$ for $(X, \bar{ok}) \leftarrow \boldsymbol{ok}[j]$
   then return $\perp$
  $\sigma = (Y^*, \sigma_0) \leftarrow \mathsf{GSign}(gpk, \boldsymbol{gsk}[i_b], Y^*, m)$
  $\text{MS} \leftarrow \text{MS} \cup \{(m, (Y^*, \sigma_0))\}$
  Return σ

Fig. 2. Oracles for security definitions.

3.2 Security Definitions

We give four security definitions for our group signature scheme GSdT. First we introduce oracles as in Fig.2. Here AddOO is "add-opener" oracle. AddUO is "add-user" oracle. StoUO is "send to user" oracle. StoIO is "send to issuer" oracle. USKO is "user secret key" oracle. GSignO is "group-signing" oracle. CrptOO is "corrupt-opener" oracle. CrptUO is "corrupt user" oracle. OpenO is "opening signature" oracle. RRegO is "read registration table" oracle. WRegO is "write registration table" oracle. ChaO_b is "challenge for b" oracle. HU is the set of honest users. CU is the set of corrupted users. OP is the set of openers. MS is the set of "queried message and replied signature" pairs. CO is the set of corrupted openers oracles. Compared with the oracles in [4], these oracles are adopted to our security definitions, *except* AddOO and CrptOO, which are new oracles for our GSdT.

Remark. In this paper, we introduce a scenario that a query to the opening oracle OpenO is issued only with such $(j, m, (Y, \sigma_0))$ that there exists an opening key $\boldsymbol{ok}[j]$ that has been already issued to an opener j with an attribute X such that

$$\mathcal{R}^{\kappa}(X, Y) = 1. \tag{5}$$

Correctness The correctness of GSdT is captured by the following experiment, where $\mathbf{A}$ is an algorithm.

$$
\begin{array}{l}
\mathsf{Expr}^{\mathrm{corr}}_{\mathsf{GSdT},\mathbf{A}}(1^{\lambda}, \kappa) \\
\quad (gpk, ik, omk) \leftarrow \mathsf{GKG}(1^{\lambda}, \kappa),\ \ \mathrm{CU} \leftarrow \emptyset, \mathrm{HU} \leftarrow \emptyset, \mathrm{OP} \leftarrow \emptyset \\
\quad (i, m, Y) \leftarrow \mathbf{A}(gpk : \mathsf{AddOO}(\cdot,\cdot), \mathsf{AddUO}(\cdot), \mathsf{RRegO}(\cdot)) \\
\quad \text{If } i \notin \mathrm{HU} \text{ then return } 0;\ \text{If } \boldsymbol{gsk}[i] = \varepsilon \text{ then return } 0 \\
\quad \sigma \leftarrow \mathsf{GSign}(gpk, \boldsymbol{gsk}[i], Y, m) \\
\quad \mathrm{OS}_Y \leftarrow \{ j \in \mathrm{OP} \mid \mathcal{R}^{\kappa}(X, Y) = 1 \text{ for } (X, \bar{ok}) \leftarrow \boldsymbol{ok}[j] \} \\
\quad \text{If } \mathrm{OS}_Y \neq \emptyset \text{ and } \mathsf{GVrfy}(gpk, m, (Y, \sigma_0)) = 0 \text{ then return } 1 \\
\quad \text{For } j \in \mathrm{OS}_Y \text{ do} \\
\quad\quad (i', \tau) \leftarrow \mathsf{Open}(gpk, \boldsymbol{ok}[j], \boldsymbol{reg}, m, \sigma) \\
\quad\quad \text{If } i \neq i' \text{ or } \mathsf{Judge}(gpk, i, \boldsymbol{upk}[i], m, \sigma, \tau) = 0 \text{ then return } 1 \\
\quad \text{Return } 0
\end{array}
$$

The advantage of $\mathbf{A}$ over GSdT is defined by

$$\mathbf{Adv}^{\mathrm{corr}}_{\mathsf{GSdT},\mathbf{A}}(\lambda) \stackrel{\mathrm{def}}{=} \Pr[\mathsf{Expr}^{\mathrm{corr}}_{\mathsf{GSdT},\mathbf{A}}(1^{\lambda}, \kappa) = 1]. \tag{6}$$

A group signature scheme GSdT is said to be *correct* if, for any unbounded $\mathbf{A}$, $\mathbf{Adv}^{\mathrm{corr}}_{\mathsf{GSdT},\mathbf{A}}(\lambda) = 0$.
Anonymity The anonymity of GSdT is captured by the following experiment.

$$
\begin{array}{l}
\mathsf{Expr}^{\mathrm{anon}\text{-}b}_{\mathsf{GSdT},\mathbf{A}}(1^{\lambda}, \kappa)\ //\ b \in \{0, 1\} \\
\quad (gpk, ik, omk) \leftarrow \mathsf{GKG}(1^{\lambda}, \kappa) \\
\quad \mathrm{CU} \leftarrow \emptyset, \mathrm{HU} \leftarrow \emptyset, \mathrm{MS} \leftarrow \emptyset, \mathrm{CO} \leftarrow \emptyset, \mathrm{OP} \leftarrow \emptyset \\
\quad d \leftarrow \mathbf{A}(gpk, ik : \mathsf{ChaO}_b(\cdot,\cdot,\cdot,\cdot), \mathsf{AddOO}(\cdot,\cdot), \mathsf{OpenO}(\cdot,\cdot,\cdot), \mathsf{StoUO}(\cdot,\cdot), \mathsf{WRegO}(\cdot,\cdot), \\
\qquad\qquad\qquad \mathsf{USKO}(\cdot), \mathsf{CrptOO}(\cdot), \mathsf{CrptUO}(\cdot,\cdot)) \\
\quad \text{Return } d
\end{array}
$$

The advantage of $\mathbf{A}$ over GSdT is define by

$$\mathbf{Adv}^{\mathrm{anon}}_{\mathsf{GSdT},\mathbf{A}}(\lambda) \stackrel{\mathrm{def}}{=} |\Pr[\mathsf{Expr}^{\mathrm{anon}\text{-}0}_{\mathsf{GSdT},\mathbf{A}}(1^{\lambda}, \kappa) = 1] - \Pr[\mathsf{Expr}^{\mathrm{anon}\text{-}1}_{\mathsf{GSdT},\mathbf{A}}(1^{\lambda}, \kappa) = 1]|. \tag{7}$$

A group signature scheme GSdT is said to be *anonymous* if, for any PPT $\mathbf{A}$, $\mathbf{Adv}^{\mathrm{anon}}_{\mathsf{GSdT},\mathbf{A}}(\lambda)$ is negligible in λ.

Traceability The traceability of GSdT is captured by the following experiment.

$$
\begin{array}{l}
\mathsf{Expr}^{\mathrm{trace}}_{\mathsf{GSdT},\mathbf{A}}(1^\lambda,\kappa)\\
\quad (gpk, ik, omk) \leftarrow \mathsf{GKG}(1^\lambda,\kappa),\ \mathrm{CU}\leftarrow\emptyset, \mathrm{HU}\leftarrow\emptyset, \mathrm{OP}\leftarrow\emptyset\\
\quad (m,(Y,\sigma_0)) \leftarrow \mathbf{A}(gpk, omk : \mathsf{StoIO}(\cdot,\cdot), \mathsf{AddUO}(\cdot), \mathsf{RRegO}(\cdot), \mathsf{USKO}(\cdot), \mathsf{CrptUO}(\cdot,\cdot))\\
\quad \text{If } \mathsf{GVrfy}(gpk, m, (Y,\sigma_0)) = 0 \text{ then return } 0\\
\quad \text{Find } X \text{ s.t. } \mathcal{R}^\kappa(X,Y)=1;\ ok \leftarrow \mathsf{OKG}(gpk, omk, 0, X)\\
\quad (i,\tau) \leftarrow \mathsf{Open}(gpk, ok, \boldsymbol{reg}, m, (Y,\sigma_0))\\
\quad \text{If } i=0 \text{ or } \mathsf{Judge}(gpk, i, \boldsymbol{upk}[i], m, (Y,\sigma_0),\tau)=0 \text{ then return } 1 \text{ else return } 0
\end{array}
$$

The advantage of $\mathbf{A}$ over GSdT is defined by

$$\mathbf{Adv}^{\mathrm{trace}}_{\mathsf{GSdT},\mathbf{A}}(\lambda) \stackrel{\mathrm{def}}{=} \Pr[\mathsf{Expr}^{\mathrm{trace}}_{\mathsf{GSdT},\mathbf{A}}(1^\lambda,\kappa)=1]. \tag{8}$$

A group signature scheme GSdT is said to be *traceable* if, for any PPT $\mathbf{A}$, $\mathbf{Adv}^{\mathrm{trace}}_{\mathsf{GSdT},\mathbf{A}}(\lambda)$ is negligible in λ.
Non-frameability The non-frameability of GSdT is captured by the following experiment.

$$
\begin{array}{l}
\mathsf{Expr}^{\mathrm{nf}}_{\mathsf{GSdT},\mathbf{A}}(1^\lambda,\kappa)\\
\quad (gpk, ik, omk) \leftarrow \mathsf{GKG}(1^\lambda,\kappa),\ \mathrm{CU}\leftarrow\emptyset, \mathrm{HU}\leftarrow\emptyset, \mathrm{OP}\leftarrow\emptyset\\
\quad (m,(Y,\sigma_0),i,\tau) \leftarrow \mathbf{A}(gpk, ik, omk : \mathsf{StoUO}(\cdot,\cdot), \mathsf{WRegO}(\cdot,\cdot), \mathsf{GSignO}(\cdot,\cdot,\cdot,\cdot),\\
\qquad\qquad\qquad\qquad\qquad\qquad \mathsf{USKO}(\cdot), \mathsf{CrptUO}(\cdot,\cdot))\\
\quad \text{If the following are all true then return 1 else return 0 :}\\
\quad \text{- } i \in \mathrm{HU} \wedge \boldsymbol{gsk}[i] \neq \varepsilon\\
\quad \text{- } \mathsf{Judge}(gpk, i, \boldsymbol{upk}[i], m, (Y,\sigma_0),\tau)=1\\
\quad \text{- } A \text{ did not query } \mathsf{USKO}(i) \vee \mathsf{GSignO}(i,m)
\end{array}
$$

The advantage of $\mathbf{A}$ over GSdT is defined by

$$\mathbf{Adv}^{\mathrm{nf}}_{\mathsf{GSdT},\mathbf{A}}(\lambda) \stackrel{\mathrm{def}}{=} \Pr[\mathsf{Expr}^{\mathrm{nf}}_{\mathsf{GSdT},\mathbf{A}}(1^\lambda,\kappa)=1]. \tag{9}$$

A group signature scheme GSdT is said to be *non-frameable* if, for any PPT $\mathbf{A}$, $\mathbf{Adv}^{\mathrm{trace}}_{\mathsf{GSdT},\mathbf{A}}(\lambda)$ is negligible in λ.

4 Construction

In this section, we describe a generic construction of our proposed group signature scheme that has designated traceability; GSdT = (GKG, OKG, UKG, Join, Iss, GSign, GVrfy, Open, Judge). We follow the construction of [4] except that we use ciphertext-policy encryption instead of public-key encryption. There are three building blocks to construct our scheme GSdT; a digital signature scheme Sig, a ciphertext-policy attribute-based encryption scheme ABE and a simulation-sound non-interactive zero-knowledge proof scheme Π_1 and Π_2. We give overview below, and the details are given in Fig.3 and Fig.4.

The group public key gpk consists of the security parameter 1^λ, a public key pk_{a} of ABE, a verification key pk_{s} for digital signatures which we call the certificate verification key, and two common reference strings R_1 and R_2 for NIZK proofs. We denote by sk_{s} the signing key corresponding to pk_{s}, and call it the certificate creation key. The issuer secret key ik is sk_{s}. Each opener's secret key $\boldsymbol{ok}[j] (j = 1, 2, \ldots, J)$ is a decryption key sk^j_X (for some key attribute X) of ABE together with the random coins $r_{\mathsf{a},j}$ used to generate sk^j_X.

In the group-joining protocol, user i, who has a key pair $(\boldsymbol{upk}[i], \boldsymbol{usk}[i])$ of Sig prior to joining, first generates a verification key pk_i and the corresponding signing key sk_i. It uses its personal private key $\boldsymbol{usk}[i]$ to generate a signature sig_i on pk_i. (The signature sig_i prevents the user from being framed by a corrupt issuer.) The user sends (pk_i, sig_i) to the issuer. The issuer issues membership data $cert_i$ to i by signing pk_i

$\mathsf{GKG}(1^\lambda, \kappa)$
 $R_1 \leftarrow \{0,1\}^{p_1(\lambda)};\ R_2 \leftarrow \{0,1\}^{p_2(\lambda)}$
 $(pk_\mathsf{a}, msk_\mathsf{a}) \leftarrow \mathsf{Setup}_\mathsf{a}(1^\lambda, \kappa);\ (pk_\mathsf{s}, sk_\mathsf{s}) \leftarrow \mathsf{KG}_\mathsf{s}(1^\lambda)$
 $gpk = (1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s});\ omk = msk_\mathsf{a};\ ik = sk_\mathsf{s}$
 Return (gpk, ik, omk)
$\mathsf{OKG}(gpk, omk, j, X)$
 Parse omk as msk_a; $r_{\mathsf{a},j} \leftarrow \{0,1\}^{r(\lambda)}$
 $sk_X^j \leftarrow \mathsf{KG}_\mathsf{a}(msk_\mathsf{a}, j, X; r_{\mathsf{a},j});\ \boldsymbol{ok}[j] \leftarrow (sk_X^j, r_{\mathsf{a},j})$
 Return $\boldsymbol{ok}[j]$
$\mathsf{UKG}(1^\lambda)$ $(upk, usk) \leftarrow \mathsf{KG}_\mathsf{s}(1^\lambda)$; Return (upk, usk)
$\mathsf{GSign}(gpk, \boldsymbol{gsk}[i], Y, m)$
 Parse gpk as $(1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$
 Parse $\boldsymbol{gsk}[i]$ as $(i, pk_i, sk_i, cert_i)$
 $s \leftarrow \mathsf{Sign}(sk_i, m);\ r \leftarrow_R \{0,1\}^\lambda$
 $C = (Y, C_0) \leftarrow \mathsf{Enc}(pk_\mathsf{a}, Y, \langle i, pk_i, cert_i, s\rangle; r)$
 $\pi_1 \leftarrow \mathsf{P}_1(1^\lambda, (pk_\mathsf{a}, pk_\mathsf{s}, m, C), (i, pk_i, cert_i, s, r), R_1)$
 Return $\sigma = (C, \pi_1)$
$\mathsf{GVrfy}(gpk, (m, \sigma))$
 Parse gpk as $(1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$;
 Parse σ as (C, π_1)
 Return $\mathsf{V}_1(1^\lambda, (pk_\mathsf{a}, pk_\mathsf{s}, m, C), \pi_1, R_1)$

$\mathsf{Open}(gpk, \boldsymbol{ok}[j], \boldsymbol{reg}, m, \sigma)$
 Parse gpk as $(1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$
 Parse $\boldsymbol{ok}[j]$ as $(sk_X^j, r_{\mathsf{a},j})$; Parse σ as (C, π_1)
 $M \leftarrow \mathsf{Dec}(pk_\mathsf{a}, sk_X^j, C)$; Parse M as $\langle i, pk, cert, s\rangle$
 If $\boldsymbol{reg}[i] \neq \varepsilon$ then parse $\boldsymbol{reg}[i]$ as (pk_i, sig_i)
 Else $pk_i \leftarrow \varepsilon, sig_i \leftarrow \varepsilon$
 $\pi_2 \leftarrow \mathsf{P}_2(1^\lambda, (pk_\mathsf{a}, C, i, pk, cert, s), (sk_X^j, r_{\mathsf{a},j}), R_2)$
 If $\mathsf{V}_1(1^\lambda, (pk_\mathsf{a}, pk_\mathsf{s}, m, C), \pi_1, R_1) = 0$
  then return $(0, \varepsilon)$
 If $pk \neq pk_i$ or $\boldsymbol{reg}[i] = \varepsilon$ then return $(0, \varepsilon)$
 $\tau = (pk_i, sig_i, i, pk, cert, s, \pi_2)$
 Return (i, τ)
$\mathsf{Judge}(gpk, i, \boldsymbol{upk}[i], m, \sigma, \tau)$
 Parse gpk as $(1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$;
 Parse σ as (C, π_1)
 If $(i, \tau) = (0, \varepsilon)$ then
  Return $\mathsf{V}_1(1^\lambda, (pk_\mathsf{a}, pk_\mathsf{s}, m, C), \pi_1, R_1)$
 Parse τ as $(\overline{pk}, \overline{sig}, i', pk, cert, s, \pi_2)$
 If $\mathsf{V}_2(1^\lambda, (pk_\mathsf{a}, C, i', pk, cert, s), \pi_2, R_2) = 0$
  then Return 0
 If the following are true then return 1
 else return 0:
  $i = i' \ \wedge\ \mathsf{Vrfy}(\boldsymbol{upk}[i], \overline{pk}, \overline{sig}) = 1 \ \wedge\ \overline{pk} = pk$

Fig. 3. Our construction of GSdT.

$\mathsf{Join}(St_{join}, M_{in})$
 If $M_{in} = \varepsilon$ then
  Parse St_{join} as (gpk, i, upk_i, usk_i)
  $(pk_i, sk_i) \leftarrow \mathsf{KG}_\mathsf{s}(1^\lambda);\ sig_i \leftarrow \mathsf{Sign}(usk_i, pk_i)$
  $St'_{join} = (i, pk_i, sk_i);\ M_{out} = (pk_i, sig_i)$
  Return $(St'_{join}, M_{out}, \mathsf{cont})$
 Else
  Parse St_{join} as (i, pk_i, sk_i); Parse M_{in} as $cert_i$
  $St'_{join} = (i, pk_i, sk_i, cert_i)$
  Return $(St'_{join}, \varepsilon, \mathsf{acc})$

$\mathsf{Iss}(St_{iss}, M_{in}, dec)$
 $M_{out} = \varepsilon;\ dec' = \mathtt{rej}$
 If $dec = \mathtt{cont}$ then
  Parse St_{iss} as (gpk, ik, i, upk_i)
  Parse M_{in} as (pk_i, sig_i)
  Parse ik as sk_s
  If $\mathsf{Vrfy}(upk_i, pk_i, sig_i) = 1$ then
   $cert_i \leftarrow \mathsf{Sign}(sk_\mathsf{s}, \langle i, pk_i\rangle)$
   $St'_{iss} = (pk_i, sig_i)$
   $M_{out} = cert_i;\ dec' = \mathsf{acc}$
 Return $(St'_{iss}, M_{out}, dec')$

Fig. 4. Our construction of GSdT (Join and Iss).

using its certificate creation key $ik(= sk_{\mathsf{s}})$. The issuer then stores (pk_i, sig_i) at $\boldsymbol{reg}[i]$ in the registration table $\boldsymbol{reg}$ (see Fig.1 and Fig.4). (Later, sig_i can be used by the opener to produce proofs for its claims.) The issuer sends back $cert_i$ to the user. The user's group signing key $\boldsymbol{gsk}[i]$ is set as $\boldsymbol{gsk}[i] = (i, pk_i, sk_i, cert_i)$ (see Fig.1 and Fig.4).

When a group member i generates a group signature for a message m, it generates a signature for a message m under pk_i by using its secret key sk_i. To make it verifiable without losing anonymity, it encrypts pk_i into $C = (Y, C_0)$ under the public key pk_{a} and a policy Y of the ciphertext-policy encryption scheme ABE. Then it proves in zero-knowledge that verification succeeds with respect to pk_i. Also, to prevent someone from simply creating their own key pair (pk_i, sk_i) and doing this, it also encrypts i and its certificate $cert_i$, and proves in zero-knowledge that $cert_i$ is a signature of $\langle i, pk_i \rangle$ under pk_{s}. Therefore, the statement of the relation ρ_1 is $(pk_{\mathsf{a}}, pk_{\mathsf{s}}, m, C)$, the witness is $(i, pk_i, cert_i, s, r)$ and the common reference string is R_1. Hence, group signature verification is verification of the NIZK proofs π_1.

When an opener opens a group signature $((Y, C_0), \pi_1)$, it first decrypts the ciphertext $C = (Y, C_0)$ in the signature $((Y, C_0), \pi_1)$ by using its secret key sk^j_X, obtains the user identity i. This decryption is possible if and only if $\mathcal{R}^\kappa(X, Y) = 1$. When i is indeed an existing user, the opener proves its claim by supplying evidence that it decrypts the ciphertext correctly, and the user public key obtained from decryption is authentic (i.e. signed by user i using $\boldsymbol{usk}[i]$). The former is accomplished by a zero-knowledge proof, where the statement of the relation ρ_2 is $(pk_{\mathsf{a}}, C, i, pk, cert_i, s)$, the witness is $(sk^j_X, r_{\mathsf{a},j})$ and the common reference string is R_2. The judge algorithm simply checks if these proofs π_2 are correct.

5 Security

In this section, we show security properties of our scheme GSdT. The security proofs can be given in a similar manner to those of [4], We remark that the anonymity of our scheme can be proven just from the IND-CPA security of the underlying ABE, whereas the anonymity of the original scheme [4] is proven from the IND-CCA security of the underlying PKE because the decryption oracle is needed to simulate OpenO. On the other hand, we can simulate OpenO in the straightforward way because the needed opening key has been generated by AddOO, which can be simulated with KGO of ABE.

Theorem 1 (Correctness) *If Sig is correct, ABE is correct, $\Pi_1 = (\mathsf{P}_1, \mathsf{V}_1)$ is complete and $\Pi_2 = (\mathsf{P}_2, \mathsf{V}_2)$ is complete, then our group signature scheme GSdT is correct. More precisely, for any unbounded $\mathbf{A}$ that is according to $\mathsf{Expr}^{corr}_{\mathsf{GSdT},\mathbf{A}}(1^\lambda, \kappa)$,*

$$\mathbf{Adv}^{corr}_{\mathsf{GSdT},\mathbf{A}}(\lambda) = 0. \tag{10}$$

Proof. The perfect correctness of Sig (Section 2.1), the perfect correctness of ABE (Section 2.2) and the perfect compeleteness of Π_1 and Π_2 (Section 2.3) imply the perfect correctness of GSdT (Section 3.2).

Theorem 2 (Anonymity) *If ABE is adaptive IND-CPA secure, $\Pi_1 = (\mathsf{P}_1, \mathsf{V}_1)$ is simulation sound and computational zero-knowledge and $\Pi_2 = (\mathsf{P}_2, \mathsf{V}_2)$ is computational zero-knowledge, then our group signature scheme GSdT is anonymous. More precisely, for any given* PPT *algorithm $\mathbf{A}$ that is according to $\mathsf{Expr}^{anon\text{-}b}_{\mathsf{GSdT},\mathbf{A}}(1^\lambda, \kappa)$ $(b = 0, 1)$, there exist* PPT *algorithms $\mathbf{A}_0$, $\mathbf{A}_1$, $\mathbf{A}_s$, $\mathbf{D}_1$ and $\mathbf{D}_2$ that are according to $\mathsf{Expr}^{ind\text{-}cpa\text{-}b}_{\mathsf{ABE},\mathbf{A}_0}(1^\lambda, \kappa)$, $\mathsf{Expr}^{ind\text{-}cpa\text{-}b}_{\mathsf{ABE},\mathbf{A}_1}(1^\lambda, \kappa)$ $(b = 0, 1)$, $\mathsf{Exp}^{ss}_{\Pi,\mathbf{A}_s}(1^\lambda)$, $\mathsf{Exp}^{zk\text{-}b}_{\mathsf{P}_1,\mathsf{Sim}_1,\mathbf{D}_1}(1^\lambda)$ $(b = 0, 1)$ and $\mathsf{Exp}^{zk\text{-}b}_{\mathsf{P}_2,\mathsf{Sim}_2,\mathbf{D}_2}(1^\lambda)$ $(b = 0, 1)$, respectively, such that the following inequality holds.*

$$\begin{aligned}\mathbf{Adv}^{anon}_{\mathsf{GSdT},\mathbf{A}}(\lambda) \leq & \mathbf{Adv}^{ind\text{-}cpa}_{\mathsf{ABE},\mathbf{A}_0}(\lambda) + \mathbf{Adv}^{ind\text{-}cpa}_{\mathsf{ABE},\mathbf{A}_1}(\lambda) \\ & + \mathbf{Adv}^{ss}_{\Pi_1,\mathbf{A}_s}(\lambda) + 2 \cdot (\mathbf{Adv}^{zk}_{\mathsf{P}_1,\mathsf{Sim}_1,\mathbf{D}_1}(\lambda) + \mathbf{Adv}^{zk}_{\mathsf{P}_2,\mathsf{Sim}_2,\mathbf{D}_2}(\lambda)).\end{aligned} \tag{11}$$

To prove Theorem 2, we need the following four lemmata.

Lemma 1 *For any given* PPT *algorithm $\mathbf{A}$, there exists a* PPT *algorithm $\mathbf{D}_2$ described in Fig.7 and the following equality holds.*

$$2 \cdot \Pr[\mathsf{Exp}^{zk\text{-}1}_{\mathsf{P}_2,\mathsf{Sim}_2,\mathbf{D}_2}(1^\lambda) = 1] = 1 + \mathbf{Adv}^{anon}_{\mathsf{GSdT},\mathbf{A}}(\lambda). \tag{12}$$

$\mathbf{A}_c(pk_{\mathsf{a}} : \mathsf{Enc}(pk_{\mathsf{a}}, \mathsf{LRO}_b(\cdot,\cdot,\cdot,\cdot)), \mathsf{KGO}(msk_{\mathsf{a}},\cdot))$ // $(c = 0, 1)$
$(St_{S_1}, R_1) \leftarrow \mathsf{Sim}_1(\mathsf{gen}, 1^\lambda)$; $(St_{S_2}, R_2) \leftarrow \mathsf{Sim}_2(\mathsf{gen}, 1^\lambda)$
$(pk_{\mathsf{s}}, sk_{\mathsf{s}}) \leftarrow \mathsf{KG}_{\mathsf{s}}(1^\lambda)$; $gpk \leftarrow (1^\lambda, R_1, R_2, pk_{\mathsf{a}}, pk_{\mathsf{s}})$; $ik \leftarrow sk_{\mathsf{s}}$
$\mathrm{CU} \leftarrow \emptyset; \mathrm{HU} \leftarrow \emptyset; \mathrm{MS} \leftarrow \emptyset; \mathrm{CList} \leftarrow \emptyset; \mathrm{CO} \leftarrow \emptyset; \mathrm{OP} \leftarrow \emptyset; d \leftarrow \bot$
$d' \leftarrow \mathbf{A}(gpk, ik : \mathsf{ChaO}_c(\cdot,\cdot,\cdot,\cdot), \mathsf{AddOO}(\cdot,\cdot), \mathsf{OpenO}(\cdot,\cdot,\cdot), \mathsf{StoUO}(\cdot,\cdot),$
$\mathsf{WRegO}(\cdot,\cdot), \mathsf{USKO}(\cdot,\cdot), \mathsf{CrptOO}(\cdot), \mathsf{CrptUO}(\cdot,\cdot))$
If $d \neq \bot$ then return d else return d'
$\mathsf{ChaO}_c(i_0, i_1, m, Y^*)$
Parse gpk as $(1^\lambda, R_1, R_2, pk_{\mathsf{a}}, pk_{\mathsf{s}})$; Parse $\boldsymbol{gsk}[i_c]$ as $(i_c, pk_{i_c}, sk_{i_c}, cert_{i_c})$
$s_c \leftarrow \mathsf{Sign}(sk_{i_c}, m)$; $M_c \leftarrow \langle i_c, pk_{i_c}, cert_{i_c}, s_c\rangle$; $M_{\bar{c}} \leftarrow 0^{|M_c|}$
$C \leftarrow \mathsf{LRO}_b(pk_{\mathsf{a}}, M_0, M_1, Y^*)$; $\mathrm{CList} \leftarrow \mathrm{CList} \cup \{C\}$
$\pi_1 \leftarrow \mathsf{Sim}_1(\mathsf{prove}, St_{S_1}, (pk_{\mathsf{a}}, pk_{\mathsf{s}}, m, C))$
Return (C, π_1)
$\mathsf{AddOO}(j, X)$
If $j \in \mathrm{OP}$ then return ε
$\mathrm{OP} \leftarrow \mathrm{OP} \cup \{j\}$; $r_{\mathsf{a},j} \leftarrow \{0,1\}^{r(\lambda)}$; $sk_X^j \leftarrow \mathsf{KGO}(X)$; $\boldsymbol{ok}[j] \leftarrow (sk_X, r_{\mathsf{a},j})$; Return 1
$\mathsf{OpenO}(j, m, (Y, \sigma_0))$
Parse (Y, σ_0) as (C, π_1)
If $\mathsf{GVrfy}(gpk, m, (Y, \sigma_0)) = 1$ and $C \in \mathrm{CList}$ then $d \leftarrow c$
$(i, \tau) \leftarrow \mathsf{Open}'(gpk, \boldsymbol{ok}, \boldsymbol{reg}, m, (Y, \sigma_0))$ // Use Sim_2 instead of P_2

Fig. 5. Adversary $\mathbf{A}_c (c = 0, 1)$ on indistinguishability of ABE, which employs adversary $\mathbf{A}$ on GSdT .

$\mathbf{A}_s(R_1 : \mathsf{Sim}_1(\mathsf{prove}, St_{S_1}, \cdot))$
$r_{\mathsf{a}} \leftarrow_R \{0,1\}^{r(\lambda)}$; $(pk_{\mathsf{a}}, msk_{\mathsf{a}} \leftarrow \mathsf{KG}_{\mathsf{a}}(1^\lambda))$
$(St_{S_2}, R_2) \leftarrow \mathsf{Sim}_2(\mathsf{gen}, 1^\lambda)$
$gpk \leftarrow (1^\lambda, R_1, R_2, pk_{\mathsf{a}}, pk_{\mathsf{s}})$
$omk \leftarrow (msk_{\mathsf{a}}, r_{\mathsf{a}})$; $ik \leftarrow sk_{\mathsf{s}}$
$\mathrm{CU} \leftarrow \emptyset$; $\mathrm{HU} \leftarrow \emptyset$; $\mathrm{MS} \leftarrow \emptyset$; $\mathrm{CList} \leftarrow \emptyset$; $y \leftarrow \bot$
$\mathbf{A}(gpk, ik : \mathsf{ChaO}_b(\cdot,\cdot,\cdot,\cdot), \mathsf{OpenO}(\cdot,\cdot,\cdot), \mathsf{StoUO}(\cdot,\cdot), \mathsf{WRegO}(\cdot,\cdot), \mathsf{USKO}(\cdot,\cdot), \mathsf{CrptUO}(\cdot,\cdot))$
Return y $\mathsf{ChaO}_b(i_0, i_1, m, Y^*)$
Parse gpk as $(1^\lambda, R_1, R_2, pk_{\mathsf{a}}, pk_{\mathsf{s}})$; Parse $\boldsymbol{gsk}[i_1]$ as $(i_1, pk_{i_1}, sk_{i_1}, cert_{i_1})$
$s_1 \leftarrow \mathsf{Sign}(sk_{i_1}, m)$; $M_1 \leftarrow \langle i_1, pk_{i_1}, cert_{i_1}, s_1\rangle$; $M_0 \leftarrow 0^{|M_1|}$
$C \leftarrow \mathsf{Enc}(pk_{\mathsf{a}}, Y^*, M_0)$; $\mathrm{CList} \leftarrow \mathrm{CList} \cup \{C\}$
$\pi_1 \leftarrow \mathsf{Sim}_1(\mathsf{prove}, St_{S_1}, (pk_{\mathsf{a}}, pk_{\mathsf{s}}, m, C))$
Return (C, π_1)
$\mathsf{OpenO}(m, \sigma)$ Parse σ as (C, π_1)
If $\mathsf{GVrfy}(gpk, m, \sigma) = 1$ and $C \in \mathrm{CList}$ then $y \leftarrow ((pk_{\mathsf{a}}, pk_{\mathsf{s}}, m, C), \pi_1)$
Run Open using Sim_2 in place of P_2,
and return the result to $\mathbf{A}$

Fig. 6. Adversary $\mathbf{A}_s$ on simulation-soundness of Π_1, which employs adversary $\mathbf{A}$ on GSdT .

$\mathbf{D}_1(1^\lambda, R_1 : \mathsf{Prove}(\cdot,\cdot))$
$r_\mathsf{a} \leftarrow_R \{0,1\}^{r(\lambda)}$
$(pk_\mathsf{a}, msk_\mathsf{a} \leftarrow \mathsf{KG}_\mathsf{a}(1^\lambda));\ (pk_\mathsf{s}, sk_\mathsf{s} \leftarrow \mathsf{KG}_\mathsf{s}(1^\lambda))$
$(St_{S_2}, R_2) \leftarrow \mathsf{Sim}_2(\mathtt{gen}, 1^\lambda)$
$gpk \leftarrow (1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$
$omk \leftarrow (msk_\mathsf{a}, r_\mathsf{a});\ ik \leftarrow sk_\mathsf{s}\quad \mathrm{CU} \leftarrow \emptyset;\ \mathrm{HU} \leftarrow \emptyset;\ \mathrm{MS} \leftarrow \emptyset$
$b \leftarrow_R \{0,1\}$
$d \leftarrow \mathbf{A}(gpk, ik : \mathsf{ChaO}_b(\cdot,\cdot,\cdot,\cdot), \mathsf{OpenO}(\cdot,\cdot,\cdot), \mathsf{StoUO}(\cdot,\cdot), \mathsf{WRegO}(\cdot,\cdot), \mathsf{USKO}(\cdot,\cdot), \mathsf{CrptUO}(\cdot,\cdot))$
If $d = b$ then return 1 else return 0
$\mathsf{ChaO}_b(i_0, i_1, m, Y^*)$
Parse gpk as $(1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$; Parse $\boldsymbol{gsk}[i_b]$ as $(i_b, pk_{i_b}, sk_{i_b}, cert_{i_b})$
$r \leftarrow_R \{0,1\}^\lambda;\ s_b \leftarrow \mathsf{Sign}(sk_{i_b}, m);\ M_b \leftarrow \langle i_b, pk_{i_b}, cert_{i_b}, s_b\rangle$
$C \leftarrow \mathsf{Enc}(pk_\mathsf{a}, Y^*, M_b;\ r)$
$\pi_1 \leftarrow \mathsf{Prove}((pk_\mathsf{a}, pk_\mathsf{s}, m, C), (i_b, pk_{i_b}, cert_{i_b}, s_b, r))$
Return (C, π_1)
$\mathsf{OpenO}(m, \sigma)$ Parse σ as (C, π_1)
Run Open using Sim_2 in place of P_2, and return the result to $\mathbf{A}$

$\mathbf{D}_2(1^\lambda, R_2 : \mathsf{Prove}(\cdot,\cdot))$
$r_\mathsf{a} \leftarrow_R \{0,1\}^{r(\lambda)}$
$(pk_\mathsf{a}, msk_\mathsf{a} \leftarrow \mathsf{KG}_\mathsf{a}(1^\lambda));\ (pk_\mathsf{s}, sk_\mathsf{s} \leftarrow \mathsf{KG}_\mathsf{s}(1^\lambda))$
$R_1 \leftarrow_R \{0,1\}^{p(\lambda)}$
$gpk \leftarrow (1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$
$omk \leftarrow (msk_\mathsf{a}, r_\mathsf{a});\ ik \leftarrow sk_\mathsf{s}\quad \mathrm{CU} \leftarrow \emptyset;\ \mathrm{HU} \leftarrow \emptyset;\ \mathrm{MS} \leftarrow \emptyset$
$b \leftarrow_R \{0,1\}$
$d \leftarrow \mathbf{A}(gpk, ik : \mathsf{ChaO}_b(\cdot,\cdot,\cdot,\cdot), \mathsf{OpenO}(\cdot,\cdot,\cdot), \mathsf{StoUO}(\cdot,\cdot), \mathsf{WRegO}(\cdot,\cdot), \mathsf{USKO}(\cdot,\cdot), \mathsf{CrptUO}(\cdot,\cdot))$
If $d = b$ then return 1 else return 0
$\mathsf{ChaO}_b(i_0, i_1, m, Y^*)$
Parse gpk as $(1^\lambda, R_1, R_2, pk_\mathsf{a}, pk_\mathsf{s})$; Parse $\boldsymbol{gsk}[i_b]$ as $(i_b, pk_{i_b}, sk_{i_b}, cert_{i_b})$
$r \leftarrow_R \{0,1\}^\lambda;\ s_b \leftarrow \mathsf{Sign}(sk_{i_b}, m);\ M_b \leftarrow \langle i_b, pk_{i_b}, cert_{i_b}, s_b\rangle$
$C \leftarrow \mathsf{Enc}(pk_\mathsf{a}, Y^*, M_b;\ r)$
$\pi_1 \leftarrow \mathsf{P}_1(1^\lambda, (pk_\mathsf{a}, pk_\mathsf{s}, m, C), (i_b, pk_{i_b}, cert_{i_b}, s_b, r), R_1)$
Return (C, π_1)
$\mathsf{OpenO}(m, \sigma)$ Parse σ as (C, π_1)
Run Open using Prove oracle in place of P_2, and return the result to $\mathbf{A}$

Fig. 7. Distinguisher $\mathbf{D}_1$ and $\mathbf{D}_2$ on zero-knowledge of Π_1 and Π_2, respectively, which employs adversary $\mathbf{A}$ on GSdT.

Proof. The equality (12) is by a standard deformation (see, for example, [4], the equality (6)).

Lemma 2 *For any given* PPT *algorithm* $\mathbf{A}$*, there exist* PPT *algorithms* $\mathbf{A}_0$*,* $\mathbf{A}_1$ *and* $\mathbf{A}_s$ *described in Figs.5 and 6 and the following equality holds.*

$$\Pr[\mathit{Expr}^{ind\text{-}cpa\text{-}0}_{ABE,\mathbf{A}_1}(1^\lambda) = 1] - \Pr[\mathit{Expr}^{ind\text{-}cpa\text{-}1}_{ABE,\mathbf{A}_0}(1^\lambda) = 1] = \mathbf{Adv}^{ss}_{\Pi_1,\mathbf{A}_s}(\lambda). \tag{13}$$

Proof. The equality (13) is derived in a similar way to the discussion in [3], the equality (9). The only difference is that, instead of the decryption oracle, we can use the key-extraction oracle. This is due to our scenario concerning the relation (5).

Lemma 3 *For any given* PPT *algorithm* $\mathbf{A}$*, there exist* PPT *algorithms* $\mathbf{D}_1$*,* $\mathbf{A}_0$ *and* $\mathbf{A}_1$ *described in Figs.7 and 5 and the following equality holds.*

$$2 \cdot \Pr[\mathit{Exp}^{zk\text{-}0}_{P_1,Sim_1,\mathbf{D}_1}] \leq 1 + \Pr[\mathit{Expr}^{ind\text{-}cpa\text{-}1}_{ABE,\mathbf{A}_1}(1^\lambda) = 1] - \Pr[\mathit{Expr}^{ind\text{-}cpa\text{-}0}_{ABE,\mathbf{A}_0}(1^\lambda) = 1]. \tag{14}$$

Proof. The equality (14) is derived in the same way as the discussion in [3], the equality (11).

Lemma 4 *For any given* PPT *algorithm* $\mathbf{A}$*, there exist* PPT *algorithms* $\mathbf{D}_1$ *and* $\mathbf{D}_2$ *described in Fig.7 and the following equality holds.*

$$\Pr[\mathit{Exp}^{zk\text{-}1}_{P_1,Sim_1,\mathbf{D}_1}(1^\lambda) = 1] = \Pr[\mathit{Exp}^{zk\text{-}0}_{P_2,Sim_2,\mathbf{D}_2}(1^\lambda) = 1]. \tag{15}$$

Proof. This is due to the definitions of the experiments $\mathsf{Exp}^{\text{zk-1}}_{\mathsf{P}_1,\mathsf{Sim}_1,\mathbf{D}_1}$ and $\mathsf{Exp}^{\text{zk-0}}_{\mathsf{P}_2,\mathsf{Sim}_2,\mathbf{D}_2}$, and of $\mathbf{D}_1$ and $\mathbf{D}_2$ given in Fig.7.

Now, from Lemma 2 and Lemma 3, we obtain the following inequality to be true.

Proposition 1

$$2 \cdot \Pr[\mathit{Exp}^{zk\text{-}0}_{P_1,Sim_1,\mathbf{D}_1}(1^\lambda) = 1] \leq 1 + \mathbf{Adv}^{ind\text{-}cpa}_{ABE,\mathbf{A}_1}(\lambda) + \mathbf{Adv}^{ind\text{-}cpa}_{ABE,\mathbf{A}_0}(\lambda) + \mathbf{Adv}^{ss}_{\Pi,\mathbf{A}}(\lambda). \tag{16}$$

Proof. By adding the both sides of the equations (13) and (14), and adding and subtracting the corresponding terms, we achieve the inequality (16).

Finally, we attain Theorem 2.

Proof. Subtract the both sides of the equality (12) from the both sides of the inequality (16), respectively. Then, to the resulted inequality, add and subtract the equal terms of (15). We obtain the inequality (11).

Theorem 3 (Traceability) *If* Sig *is EUF-CMA secure,* $\Pi_1 = (P_1, V_1)$ *is sound and* $\Pi_2 = (P_2, V_2)$ *is sound, then our group signature scheme* GSdT *is traceable. More precisely, for any given* PPT *algorithm* $\mathbf{A}$ *that is according to* $\mathit{Expr}^{trace}_{GSdT,\mathbf{A}}(1^\lambda, \kappa)$*, there exists* PPT *algorithm* $\mathbf{F}$ *that is according to* $\mathit{Expr}^{euf\text{-}cma}_{Sig,\mathbf{F}}(1^\lambda)$ *such that the following inequality holds.*

$$\mathbf{Adv}^{trace}_{GSdT,\mathbf{A}}(\lambda) \leq 2^{-\lambda} + \mathbf{Adv}^{euf\text{-}cma}_{Sig,\mathbf{F}}(\lambda). \tag{17}$$

Proof. The proof goes basically in the same way as the deduction in [4].

Theorem 4 (Non-frameability) *If* Sig *is EUF-CMA secure,* $\Pi_1 = (P_1, V_1)$ *is sound and* $\Pi_2 = (P_2, V_2)$ *is sound, then our group signature scheme* GSdT *is non-frameable. More precisely, for any given* PPT *algorithm* $\mathbf{A}$ *that is according to* $\mathit{Expr}^{nf}_{GSdT,\mathbf{A}}(1^\lambda, \kappa)$ *and that generates at most* $N(\lambda)$ *honest users, there exist* PPT *algorithms* $\mathbf{F}_1$ *and* $\mathbf{F}_2$ *that are according to* $\mathit{Expr}^{euf\text{-}cma}_{Sig,\mathbf{F}_1}(1^\lambda)$ *and* $\mathit{Expr}^{euf\text{-}cma}_{Sig,\mathbf{F}_1}(1^\lambda)$*, respectively, such that the following inequality holds.*

$$\mathbf{Adv}^{nf}_{GSdT,\mathbf{A}}(\lambda) \leq 2^{-\lambda+1} + N(\lambda) \cdot (\mathbf{Adv}^{euf\text{-}cma}_{Sig,\mathbf{F}_1}(\lambda) + \mathbf{Adv}^{euf\text{-}cma}_{Sig,\mathbf{F}_2}(\lambda)). \tag{18}$$

Proof. The proof goes basically in the same way as the deduction in [4].

6 Conclusion

In this paper, we introduced the notion of designated traceability, which limits excessiveness of the opening function in that that users are capable of specifying access structures of openers. This study is a first step towards *mutual accountability* between the openers and the users in group signature schemes, and this direction should be our future work.

Acknowledgments

The authors would like to express our sincere thanks to the anonymous reviewers of CANDAR 2021 symposium for their constructive comments.

References

1. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme. In: Bellare, M. (ed.) Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1880, pp. 255–270. Springer (2000). https://doi.org/10.1007/3-540-44598-6_16, https://doi.org/10.1007/3-540-44598-6_16
2. Attrapadung, N.: Dual system encryption framework in prime-order groups via computational pair encodings. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10032, pp. 591–623 (2016). https://doi.org/10.1007/978-3-662-53890-6_20, https://doi.org/10.1007/978-3-662-53890-6_20
3. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In: Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings. pp. 614–629 (2003). https://doi.org/10.1007/3-540-39200-9_38, http://dx.doi.org/10.1007/3-540-39200-9_38
4. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: The case of dynamic groups. In: Menezes, A. (ed.) Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3376, pp. 136–153. Springer (2005). https://doi.org/10.1007/978-3-540-30574-3_11, https://doi.org/10.1007/978-3-540-30574-3_11
5. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. pp. 41–55 (2004). https://doi.org/10.1007/978-3-540-28628-8_3, http://dx.doi.org/10.1007/978-3-540-28628-8_3
6. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6597, pp. 253–273. Springer (2011). https://doi.org/10.1007/978-3-642-19571-6_16, https://doi.org/10.1007/978-3-642-19571-6_16
7. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J.: Foundations of fully dynamic group signatures. In: Manulis, M., Sadeghi, A., Schneider, S.A. (eds.) Applied Cryptography and Network Security - 14th International Conference, ACNS 2016, Guildford, UK, June 19-22, 2016. Proceedings. Lecture Notes in Computer Science, vol. 9696, pp. 117–136. Springer (2016). https://doi.org/10.1007/978-3-319-39555-5_7, https://doi.org/10.1007/978-3-319-39555-5_7
8. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short accountable ring signatures based on DDH. In: Pernul, G., Ryan, P.Y.A., Weippl, E.R. (eds.) Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I. Lecture Notes in Computer Science, vol. 9326, pp. 243–265. Springer (2015). https://doi.org/10.1007/978-3-319-24174-6_13, https://doi.org/10.1007/978-3-319-24174-6_13
9. Chaum, D., van Heyst, E.: Group signatures. In: Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings. pp. 257–265 (1991). https://doi.org/10.1007/3-540-46416-6_22, http://dx.doi.org/10.1007/3-540-46416-6_22
10. Emura, K., Hanaoka, G., Kawai, Y., Matsuda, T., Ohara, K., Omote, K., Sakai, Y.: Group signatures with message-dependent opening: Formal definitions and constructions. Secur. Commun. Networks **2019**, 4872403:1–4872403:36 (2019). https://doi.org/10.1155/2019/4872403, https://doi.org/10.1155/2019/4872403

11. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. **17**(2), 281–308 (1988). https://doi.org/10.1137/0217017, https://doi.org/10.1137/0217017
12. Kohlweiss, M., Miers, I.: Accountable metadata-hiding escrow: A group signature case study. Proc. Priv. Enhancing Technol. **2015**(2), 206–221 (2015). https://doi.org/10.1515/popets-2015-0012, https://doi.org/10.1515/popets-2015-0012
13. Lewko, A.B., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings. pp. 62–91 (2010). https://doi.org/10.1007/978-3-642-13190-5_4, https://doi.org/10.1007/978-3-642-13190-5_4
14. Ling, S., Nguyen, K., Wang, H., Xu, Y.: Constant-size group signatures from lattices. In: Abdalla, M., Dahab, R. (eds.) Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10770, pp. 58–88. Springer (2018). https://doi.org/10.1007/978-3-319-76581-5_3, https://doi.org/10.1007/978-3-319-76581-5_3
15. Ling, S., Nguyen, K., Wang, H., Xu, Y.: Accountable tracing signatures from lattices. In: Matsui, M. (ed.) Topics in Cryptology - CT-RSA 2019 - The Cryptographers' Track at the RSA Conference 2019, San Francisco, CA, USA, March 4-8, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11405, pp. 556–576. Springer (2019). https://doi.org/10.1007/978-3-030-12612-4_28, https://doi.org/10.1007/978-3-030-12612-4_28
16. Nakanishi, T., Funabiki, N.: Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps. In: Roy, B.K. (ed.) Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3788, pp. 533–548. Springer (2005). https://doi.org/10.1007/11593447_29, https://doi.org/10.1007/11593447_29
17. Ohara, K., Sakai, Y., Emura, K., Hanaoka, G.: A group signature scheme with unbounded message-dependent opening. In: Chen, K., Xie, Q., Qiu, W., Li, N., Tzeng, W. (eds.) 8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '13, Hangzhou, China - May 08 - 10, 2013. pp. 517–522. ACM (2013). https://doi.org/10.1145/2484313.2484382, https://doi.org/10.1145/2484313.2484382
18. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings. pp. 457–473 (2005). https://doi.org/10.1007/11426639_27, https://doi.org/10.1007/11426639_27
19. Sakai, Y., Emura, K., Hanaoka, G., Kawai, Y., Matsuda, T., Omote, K.: Group signatures with message-dependent opening. In: Abdalla, M., Lange, T. (eds.) Pairing-Based Cryptography - Pairing 2012 - 5th International Conference, Cologne, Germany, May 16-18, 2012, Revised Selected Papers. Lecture Notes in Computer Science, vol. 7708, pp. 270–294. Springer (2012). https://doi.org/10.1007/978-3-642-36334-4_18, https://doi.org/10.1007/978-3-642-36334-4_18
20. Xu, S., Yung, M.: Accountable ring signatures: A smart card approach. In: Quisquater, J., Paradinas, P., Deswarte, Y., Kalam, A.A.E. (eds.) Smart Card Research and Advanced Applications VI, IFIP 18th World Computer Congress, TC8/WG8.8 & TC11/WG11.2 Sixth International Conference on Smart Card Research and Advanced Applications (CARDIS), 22-27 August 2004, Toulouse, France. IFIP, vol. 153, pp. 271–286. Kluwer/Springer (2004). https://doi.org/10.1007/1-4020-8147-2_18, https://doi.org/10.1007/1-4020-8147-2_18