論文 / 著書情報
Article / Book Information

題目(和文)	グループ署名に関する種々の研究
Title(English)	Studies on Group Signature
著者(和文)	石田愛
Author(English)	Ai Ishida
出典(和文)	学位:博士(理学), 学位授与機関:東京工業大学, 報告番号:甲第10755号, 授与年月日:2018年3月26日, 学位の種別:課程博士, 審査員:田中 圭介,渡辺 治,鹿島 亮,伊東 利哉,尾形 わかは
Citation(English)	Degree:Doctor (Science), Conferring organization: Tokyo Institute of Technology, Report number:甲第10755号, Conferred date:2018/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Studies on Group Signature
(グループ署名に関する種々の研究)

Ai Ishida

Supervisor: Keisuke Tanaka

Department of Mathematical and Computing Science
Tokyo Institute of Technology

February 23, 2018

Acknowledgement

First and foremost, I would like to express my deepest and sincere gratitude to my supervisor Keisuke Tanaka. Throughout these six years, he gave me a lot of helpful advice not only on my studies but also on my personal problems. I wouldn't have been able to finish my doctoral course without his supports. I am really glad that I have met him and could join Tanaka Laboratory.

I would like to thank the referees of my doctoral thesis, Prof. Toshiya Ito, Prof. Wakaha Ogata, Prof. Ryo Kashima, and Prof. Osamu Watanabe for taking their time to reading this thesis and to listening to my presentations. Their feedbacks helped me to improve my work.

Special thanks also to Goichiro Hanaoka for giving me a chance to join Shin-Akarui-Angou-Benkyou-Kai. I would like to express my gratitude to all the members of Shin-Akarui-Angou-Benkyou-Kai. I have had a lot of experience in the discussion with them. I especially thank to Keita Emura, Yusuke Sakai, and Shota Yamada for fruitful discussions. Without their guidance and persistent help, this thesis would not be done.

I would like to offer my special thanks to current members and ex-members of Tanaka Laboratory. I have spent enjoyable and delightful time and have a lot of wonderful memories with them. Especially, I thank to my colleagues and ex-colleagues, Wang Yuyu, Ryosuke Nakata, Tomoyuki Komatsu, and Fuyuki Kitagawa, who shared much time and developed through friendly competition with me.

Last but not least, I owe my deepest gratitude to my parents, Shigeru and Michiyo, and my sister, Miyako for their encouragement and understanding. I have been supported by them all the time of my student life.

This work was supported by JSPS KAKENHI Grant Number 17J07416.

Contents

Chapter 1

Introduction

1.1 Background

Cryptography is the study of techniques for achieving secure communication channels in the presence of third parties. The goal of this study is constructing protocols which provide security functions such as confidentiality, integrity, authenticity, non-repudiation, and anonymity. In modern society, cryptographic protocols are nowadays essential in our lives since they are used in various electronic systems (e.g., mobile phones, e-mail, web browsers, smart cards, online shopping, and online banking).

Digital signature is one of the most important primitives in public key cryptography. Intuitively, it has the same role for digital messages that physical signature has for paper documents. Concretely, digital signature is a useful tool for providing integrity, authenticity, and non-repudiation. Digital signature schemes have been used for stand-alone applications such as certifying contracts, documents, and authenticating individuals and also for a building block in the design of other primitives.

Although a signer's privacy is not originally considered in digital signature, it needs to protect a user information in some applications. For such applications, special types of digital signature which take into account signer's privacy were proposed, e.g., group signature [39], ring signature [97], direct anonymous attestation [32], and list signature [38]. This type of digital signature allows users to anonymously sign a message, but still has the functionality that only legitimate users can generate a valid signature. Due to such a functionality, these digital signature schemes help to realize many more attractive systems such as e-voting and e-bidding [9, 73, 41], privacy-preserving attestation [32, 19], and vehicle safety communication [40] in practice.

Especially, group signature introduced by Chaum and van Heyst [39] is the most classical digital signature considering users' privacy. By using a group signature scheme, a user can sign a message on behalf of the group without revealing his identity. However,

in the case of disputes, the group manager can identify the signer from a signature. In the following, we look back on the brief history of group signature.

History of Standard Group Signature. As mentioned, the notion of group signature was introduced by Chaum and van Heyst [39], and they also proposed the first schemes. After their work, the first provably coalition-resistant group signature scheme was proposed by Ateniese, Camenisch, Joye, and Tsudik [9] through some proposals [33, 37, 11]. At the time, however, there was no proper formalization of the security requirements that can be naturally expected from group signature.

Bellare, Micciancio, and Warinschi [14] proceeded to this problem for static group signature schemes, in which the number of members is fixed in the setup phase, and a member cannot join after the setup. More precisely, they introduced formal definitions for the core requirements of anonymity and traceability, and showed that these imply the existing informal requirements in the literature. The Boneh-Boyen-Shacham scheme [27] is an efficient group signature in (a relaxation of) this model. This scheme was proved to be secure by using the random oracle model methodology [15].

Later on, group signature in the dynamic setting was independently formalized by Bellare, Shi, and Zhang [17] and Kiayias and Yung [75]. In the dynamic setting, neither the number nor the identities of members are fixed or known in the setup phase. The Nguyen-Safavi-Naini scheme [94], the Furukawa-Imai scheme [54], and the Delerablée-Pointcheval scheme [46] are given as examples of efficient schemes in these dynamic models, and all of them are secure in the random oracle model.

The efficient schemes that is provably secure without random oracles also have been investigated. In the static setting, Boyen and Waters [29] presented the first efficient scheme in the standard model with a novel adaptation of the Groth-Ostrovsky-Sahai non-interactive zero-knowledge proof system [62]. They subsequently proposed the scheme with constant-size signatures by the adaptation of q-type assumption [30]. In the dynamic setting, Groth [60] proposed the first group signature scheme without random oracles where the size of a signature is constant. However, this constant is enormous, and thus the scheme is not practical. Thereafter, Groth [61] gave a group signature scheme where all parts of the scheme are constant, and these constants are reasonable for practical use. Most recently, in both the static and the dynamic settings, Libert, Peters, and Yung [82] proposed short group signature schemes based on simple assumptions via a structure-preserving signature scheme (its notion is proposed in the paper [8]). For the currently recommended 128-bit security level, the signature sizes of the Libert-Peters-Yung schemes are 1 kilobytes and 1.8 kilobytes in the static and the dynamic settings, respectively.

History of Revocable Group Signature. One of important research topics in group signature is membership revocation, and this has been widely investigated so far [31, 102, 10, 36, 103, 27, 28, 87, 88, 86, 80, 81, 50, 51].

When a member cheats, he must lose his rights to sign. Moreover, there are more rational reasons that a user is revoked prematurely: the loss of a signing key, the withdrawal from the group, and the change of an affiliation. The simplest solution is that the group manager generates a new group public key and user signing keys, and sends them to non-revoked members. However, this solution is not desirable since it is inconvenient to re-distribute user signing keys especially in large groups.

Bresson and Stern [31] proposed the first non-trivial approach that a signer proves that his membership certificate is not included in the list of revoked certificates. Subsequently, Song [102] and Ateniese, Song, and Tsudik [10] presented solutions for revocation in group signatures. However, signing and/or verification requires a computation that is linear in the number of removed members in all of these schemes.

Camenisch and Lysyanskaya [36] suggested an elegant approach by using accumulators [18], and this approach leads $\mathcal{O}(1)$ costs for both signing and verifying with respect to the group size N and the number of removed users r. Their technique was also used by Tsudik and Xu [103] and Camenisch, Kohlweiss, and Soriente [35]. However, in the Camenisch-Lysyanskaya scheme [36], a signer has to modify his signing key with a computation of $\mathcal{O}(N)$ when a revocation occurs. Although this computation was relaxed into $\mathcal{O}(r)$ by Boneh, Boyen, and Shacham [27] and Camenisch and Groth [34], group members still had to update their signing keys.

Boneh and Shacham [28] proposed the notion of another revocable group signature called group signature with verifier-local revocation (VLR-GS). In a VLR-GS scheme, verifiers need to download the up-to-date information of the revoked users to verify signatures but signers are not required to do so. This means that a user can sign massages without referring information regarding revoked users, that is, not only the signing cost is completely independent of the number of removed users, but also group members do not need to update their signing keys. However, in contrast, the verification cost is inevitably linear in the number of removed users.

Nakanishi, Fuji, Hira, and Funabiki [86] proposed a revocable group signature scheme where the computational costs of signing and verifying are $\mathcal{O}(1)$, and updates of secret keys are not required. Unfortunately, this scheme suffers from a linear-size group public key in the number of group members. If we impose signing and verifying on constant extra costs, the group public key size is reduced to $\mathcal{O}(\sqrt{N})$, but this is still large.

The first scalable scheme was proposed by Libert, Peters, and Yung [81] where they used a technique based on the Naor-Naor-Lotspiech broadcast encryption framework [90]. Although their scheme has membership certificates with poly-logarithmic size in the group

size, constant size signing keys were subsequently achieved by the same authors [80].

1.2 Contributions of the Thesis

This thesis has four contributions in the field of group signature: to discuss the minimal assumptions of group signature, to overcome the barriers of full anonymity in VLR-GS, to introduce new useful functionality of group signature, and to give a cryptanalysis of the standardized group signature scheme. We outline each contribution in the following.

Discussion of the Minimal Assumptions of Group Signature. We discuss the minimum assumptions for the existence of group signature. Specifically, we point out that these minimal assumptions are depends on whether the target group signature is fully anonymous or selfless anonymous.

The previous works [6, 95, 48, 49] showed that a public key encryption (PKE) scheme (and its extended schemes [44, 57, 26]) can be constructed from a group signature scheme that satisfies full anonymity. Therefore, it is unlikely to construct a fully anonymous group signature scheme only from a one-way function (OWF). This is because, if a fully anonymous group signature scheme can be constructed from a OWF, this fact contradicts to the impossibility result by Impagliazzo and Rudich [67]. On the other hand, there still remains a possibility that a group signature scheme which satisfies selfless anonymity [28] can be constructed without a PKE scheme since a conversion from a selfless anonymous group signature scheme to a PKE scheme is not known.

In this thesis, we give a construction of a selfless anonymous group signature scheme without any PKE scheme. Concretely, we construct a selfless anonymous group signature scheme from a symmetric key encryption scheme, a commitment scheme, a digital signature scheme, and a non-interactive zero-knowledge (NIZK) proof system. This result indicates that a selfless anonymous group signature scheme can be constructed from a OWF and a NIZK proof system. Moreover, from the result, we discuss the gap between fully anonymous group signature and selfless anonymous group signature from the practical and theoretical aspects.

The Realization of a Fully Anonymous VLR-GS Scheme. We propose the first VLR-GS scheme that satisfies full anonymity, which is considered to be the de-facto standard security notion.

As mentioned, VLR-GS is a special type of revocable group signature which enables a user to sign massages without referring information regarding revoked users. After the first scheme was given by Boneh and Shacham [28] in 2004, there have been several proposals of VLR-GS schemes [28, 87, 88, 106, 83, 78]. However, all of these schemes only

achieve a weak security notion, selfless anonymity. This security notion is strictly weaker than the de-facto standard security notion, full anonymity. Therefore, for more than a decade, it has been an open problem whether a fully anonymous VLR-GS scheme can be achieved since it is known that there is a big theoretical gap between selfless anonymous group signature and fully anonymous group signature.

In this thesis, we give an affirmative answer to this problem. Concretely, we show a construction of a fully anonymous VLR-GS scheme from a digital signature scheme, a PKE scheme, and a non-interactive zero-knowledge proof system. Although the building blocks are essentially the same as those of a standard group signature scheme [14], we additionally require the underlying PKE scheme to satisfy key privacy [13] which is essential to ensure that the VLR-GS scheme is fully anonymous. Moreover, we give VLR-GS schemes with backward unlinkability [87], which ensures that even after a user is revoked, signatures produced by the user before the revocation remain anonymous.

New Functionality of Group signature: Deniability. We propose new functionality of group signature, deniability. This functionality allows the opener to generate a proof showing that the specified user is not the signer without revealing the actual signer.

By using a group signature scheme, a user can sign a message on behalf of a specific group without revealing his identity, but in the case of a dispute, the opener can expose the identity of the signer. Although such a functionality seems to be quite useful for protecting users' anonymity and tracing a malicious user simultaneously, this is insufficient for some situations in which a user wants to only show that he is not the signer of a signature.

In this thesis, we introduce the notion of deniable group signature, which allows an authority to prove that the specified user is not the signer. More precisely, in addition to all the functionalities of standard group signature, deniable group signature provides another functionality that the opener can generate a denial proof that proves non-ownership of a signature for a user. Moreover, we propose a method for designing a deniable group signature scheme and a concrete instantiation.

Cryptanalysis of the Standardized Group Signature Scheme. We give a cryptanalysis of the scheme denoted as Mechanism 6 in the ISO/IEC 20008-2 standard [2]. Mechanism 6 is the only standardized group signature scheme that does not aim at providing additional functionalities.

In this thesis, we firstly break the anonymity of Mechanism 6 in the standard security model, i.e., the Bellare-Shi-Zhang model [17]. We then discuss possible countermeasures against our attack. Consequently, we provide a detailed analysis of the security properties offered by Mechanism 6 and characterize the conditions under which its anonymity is preserved. From this analysis, we also derive a simple patch for Mechanism 6.

1.3 Organization

In Chapter 2, we review bilinear map and complexity assumptions. Moreover, models and security requirements of cryptographic primitives used in this thesis (pseudorandom function, commitment, digital signature, symmetric key encryption, public key encryption, identity-based encryption, and non-interactive proof) are given in this chapter.

In Chapter 3, we review the model of static group signature. Here, we introduce the model given by Bellare, Micciancio, and Warinschi [14]. The syntax of the Bellare-Micciancio-Warinschi model is described in Section 3.1, and its security requirements are defined in Section 3.2. Furthermore, in Section 3.3, we introduce the construction of a static group signature scheme by Bellare et al. [14].

In Chapter 4, we review the model of group signature in the dynamic setting. We review the model by Sakai, Schuldt, Emura, Hanaoka, and Ohta [100], which is an extended model based on the Bellare-Shi-Zhang model [17]. As in Chapter 3, we review both the syntax and the security requirements in Section 4.1 and 4.2, respectively. In Section 4.3, we introduce the modified Groth scheme [100], which is one of efficient schemes secure in this dynamic setting.

In Chapter 5, we discuss the minimum assumptions for the existence of group signature. Firstly, we review results of some previous works and recall the minimal assumptions of fully anonymous group signature in Section 5.1. Secondly, we give a construction of a selfless anonymous group signature scheme via an SKE scheme satisfies the two properties: ciphertext-pseudorandomness and key-robustness in Section 5.2. Lastly, from the results of previous sections, we consider the gap between fully anonymous group signature and selfless anonymous group signature from the practical and theoretical aspects in Section 5.3.

In Chapter 6, we focus on group signature with verifier-local revocation (VLR-GS) [28]. Especially, here, we show the first construction of a fully anonymous VLR-GS scheme. We firstly review the syntax of VLR-GS in Section 6.1 and then give the definition of the security requirements for VLR-GS schemes in Section 6.2. Finally, we provide constructions of fully anonymous VLR-GS schemes in Section 6.3.

In Chapter 7, we introduce the notion of deniable group signature. Firstly, we formalize its syntax and security requirements in Section 7.1 and 7.2, respectively. Secondly, we show a generic construction of deniable group signature scheme and explain the difficulty of instantiating an efficient scheme even though a generic construction is given in Section 7.3. Then, in Section 7.4, we give a fairly practical deniable group signature scheme with security proofs. Lastly, we introduce some primitives similar to deniable group signature as related works in Section 7.5.

In Chapter 8, we give a cryptanalysis of the scheme denoted as Mechanism 6 in

the ISO/IEC 20008-2 standard. Firstly, a description of Mechanism 6 is provided in Section 8.1. Then, in Section 8.2, we give a cryptanalysis of Mechanism 6 and show that the scheme does not satisfies the expected security level. Moreover, we discuss possible countermeasures against our attack in this section. After that, in Section 8.3, we provide a detailed analysis of the security properties offered by Mechanism 6 and characterize the conditions under which its anonymity is preserved. Finally, we provide a simple patched scheme which achieves the expected security level in Section 8.4.

Lastly, in Chapter 9, we conclude this thesis and leave future works.

Chapter 2

Preliminaries

In this chapter, we review bilinear map and some complexity assumptions, and models and security requirements of several cryptographic primitives used in this thesis. First of all, we define some notations.

Notations. $x \xleftarrow{\$} X$ denotes choosing an element from a finite set X uniformly at random. If A is a probabilistic algorithm, $y \leftarrow \mathsf{A}(x; r)$ denotes the operation of running A on an input x and a randomness r, and letting y be the output. When it is not necessary to specify the randomness, we omit it and simply write $y \leftarrow \mathsf{A}(x)$. If we describe the statement that the output of $\mathsf{A}(x)$ is y, then we denote $\mathsf{A}(x) = y$. If $\mathcal{O}$ is a function or an algorithm, $\mathsf{A}^{\mathcal{O}}$ denotes that A has oracle access to $\mathcal{O}$. λ denotes a security parameter. PPT stands for probabilistic polynomial time. A function $f(\lambda)$ is called negligible and denoted as $\mathsf{negl}(\lambda)$ if for any $c > 0$, there exists an integer Λ such that $f(\lambda) < \frac{1}{\lambda^c}$ for all $\lambda > \Lambda$.

2.1 Bilinear Map and Complexity Assumptions

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be multiplicative cyclic groups of order p where p is a λ-bit prime. Let G_1 and G_2 be a generator of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Let e be a computable map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with bilinearity: for all $a, b \in \mathbb{Z}$, $e({G_1}^a, {G_2}^b) = e(G_1, G_2)^{ab}$, and non-degeneracy: $e(G_1, G_2) \neq 1$. We say that groups $(\mathbb{G}_1, \mathbb{G}_2)$ are a bilinear group pair if there exists the bilinear map e as above, and the group operations in $\mathbb{G}_1$ and $\mathbb{G}_2$ and the bilinear map e are efficiently computable. If $\mathbb{G}_1 = \mathbb{G}_2$, we use a notation $\mathbb{G}$ to denote $\mathbb{G}_1$ and $\mathbb{G}_2$ and say that e is a type-I pairing. If $\mathbb{G}_1 \neq \mathbb{G}_2$ and there exists an isomorphism Ψ from $\mathbb{G}_2$ to $\mathbb{G}_1$ with $\Psi(G_2) = G_1$, we say that e is a type-II pairing. Otherwise, we say that e is a type-III pairing. In this thesis, we consider bilinear maps $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ where $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are groups of prime order p.

We introduce the several assumptions which are used in this thesis.

Definition 2.1.1 (Discrete Logarithm (DL) Assumption)**.** *We say that the DL assumption holds in* $\mathbb{G}_1$ *if for any PPT adversary* $\mathcal{A}$*, the advantage* $\mathsf{Adv}_{\mathcal{A}}^{DL}(\lambda) := \Pr[H = {G_1}^x | x \leftarrow \mathcal{A}(G_1, G_2, H)]$ *is negligible, where the probability is taken over the random choices of a generator* $G_2 \in \mathbb{G}_2$ *with* $G_1 = \Psi(G_2)$*, of an element* $H \in \mathbb{G}_1$*, and a random coin of* $\mathcal{A}$*.*

Definition 2.1.2 (Decisional Linear Assumption (DLIN) Assumption [27])**.** *We say that the DLIN assumption holds in* $\mathbb{G}_1$ *if for any PPT adversary* $\mathcal{A}$*, the advantage* $\mathsf{Adv}_{\mathcal{A}}^{DLIN}(\lambda) := |\Pr[1 \leftarrow \mathcal{A}(G_1, f, g, h, f^a, g^b, h^{a+b})] - \Pr[1 \leftarrow \mathcal{A}(G_1, f, g, h, f^a, g^b, h^c)]|$ *is negligible, where the probability is taken over the random choices of a generator* $G_1 \in \mathbb{G}_1$ *and of elements* $a, b, c \in \mathbb{Z}_p$*, and a random coin of* $\mathcal{A}$*.*

Definition 2.1.3 (External Diffie-Hellman (XDH) Assumption)**.** *We say that the XDH assumption holds in* $\mathbb{G}_1$ *if for any PPT adversary* $\mathcal{A}$*, the advantage* $\mathsf{Adv}_{\mathcal{A}}^{XDH}(\lambda) := |\Pr[1 \leftarrow \mathcal{A}(G_1, G_2, {G_1}^a, {G_1}^b, {G_1}^{ab})] - \Pr[1 \leftarrow \mathcal{A}(G_1, G_2, {G_1}^a, {G_1}^b, W)]|$ *is negligible, where the probability is taken over the random choices of a generator* $G_2 \in \mathbb{G}_2$ *with* $G_1 = \Psi(G_2)$*, of elements* $a, b \in \mathbb{Z}_p$*, and of an element* $W \in \mathbb{G}_1$*, and a random coin of* $\mathcal{A}$*.*

Definition 2.1.4 (q-Strong Diffie-Hellman (q-SDH) Assumption [24, 25])**.** *We say that the q-SDH assumption holds in* $(\mathbb{G}_1, \mathbb{G}_2)$ *if for any PPT adversary* $\mathcal{A}$*, the advantage* $\mathsf{Adv}_{\mathcal{A}}^{q\text{-}SDH}(\lambda) := \Pr[e(C, {G_2}^\gamma \cdot {G_2}^x) = e(G_1, G_2) | (C, x) \leftarrow \mathcal{A}(G_1, G_2, {G_2}^\gamma, {G_2}^{\gamma^2}, \ldots, {G_2}^{\gamma^q})]$ *is negligible, where the probability is taken over the random choices of a generator* $G_2 \in \mathbb{G}_2$ *with* $G_1 = \Psi(G_2)$ *and of a value* $\gamma \in \mathbb{Z}_p^*$*, and a random coin of* $\mathcal{A}$*.*

In this thesis, we introduce the simplified q-strong Diffie-Hellman (simplified q-SDH) assumption to simplify the proof.

Definition 2.1.5 (Simplified q-Strong Diffie-Hellman Assumption [24])**.** *We say that the simplified q-SDH assumption holds in* $(\mathbb{G}_1, \mathbb{G}_2)$ *if for any PPT adversary* $\mathcal{A}$*, the advantage* $\mathsf{Adv}_{\mathcal{A}}^{sim\text{-}q\text{-}SDH}(\lambda) := \Pr[x \neq x_i \wedge e(C, {G_2}^\gamma \cdot {G_2}^x) = e(G_1, G_2) | (C, x) \leftarrow \mathcal{A}(G_1, G_2, {G_2}^\gamma, \{{G_1}^{\frac{1}{\gamma + x_i}}, x_i\}_{i=1}^q)]$ *is negligible, where the probability is taken over the random choices of a generator* $G_2 \in \mathbb{G}_2$ *with* $G_1 = \Psi(G_2)$*, of a value* $\gamma \in \mathbb{Z}_p^*$*, and of values* $x_i \in \mathbb{Z}_p$*, and a random coin of* $\mathcal{A}$*.*

The following theorem is known between the q-SDH assumption and the simplified $(q-1)$-SDH assumption. Therefore, we use the simplified $(q-1)$-SDH assumption instead of the q-SDH assumption in our security proof.

Theorem 2.1.1 ([24])**.** *For any PPT adversary* $\mathcal{A}$ *and any integer* $q > 0$*, it holds that* $\mathsf{Adv}_{\mathcal{A}}^{sim\text{-}(q-1)\text{-}SDH}(\lambda) \leq \mathsf{Adv}_{\mathcal{A}}^{q\text{-}SDH}(\lambda)$*. That is, if the q-SDH assumption holds, the simplified* $(q-1)$*-SDH assumption also holds.*

Definition 2.1.6 (q-Unfakeability (q-U) Assumption [61])**.** *We say that the q-U assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ if for any PPT adversary $\mathcal{A}$, the advantage* $\mathsf{Adv}^{q\text{-}U}_{\mathcal{A}}(\lambda) := \Pr[V \notin \{{G_2}^{x_1}, \ldots, {G_2}^{x_{q(\lambda)}}\} \wedge e(A, hV)e(f, B) = T \wedge e(S, V \cdot {G_1}^{m}) = e(G_1, G_2) | x_1, r_1, \ldots, x_{q(\lambda)}, r_{q(\lambda)} \leftarrow \mathbb{Z}_p^*; f, h, z \leftarrow \mathbb{G}_1; T = e(f, x); a_i = f^{r_i};\ b_i = h^{r_i} {G_1}^{x_i r_i} z; (V, A, B, m, S) \leftarrow \mathcal{A}(G_1, G_2, f, h, T, x_1, a_1, b_1, \ldots, x_{q(\lambda)}, a_{q(\lambda)}, b_{q(\lambda)})]$ *is negligible, where the probability is taken over the random choices of a generator $G_2 \in \mathbb{G}_2$ with $G_1 = \Psi(G_2)$ and of values $x_1, r_1, \ldots, x_{q(\lambda)}, r_{q(\lambda)} \in \mathbb{Z}_p^*$ and $f, h, z \in \mathbb{G}_1$, and a random coin of $\mathcal{A}$.*

Furthermore, we consider a multiplicative cyclic group $\mathbb{G}$ of order p in which the decisional Diffie-Hellman assumption holds and define this assumption in the following.

Definition 2.1.7 (Decisional Diffie-Hellman (DDH) Assumption)**.** *We say that the DDH assumption holds in $\mathbb{G}$ if for any PPT adversary $\mathcal{A}$, the advantage* $\mathsf{Adv}^{DDH}_{\mathcal{A}}(\lambda) := |\Pr[1 \leftarrow \mathcal{A}(G, G^a, G^b, G^{ab})] - \Pr[1 \leftarrow \mathcal{A}(G, G^a, G^b, W)]|$ *is negligible, where the probability is taken over the random choices of a generator $G \in \mathbb{G}$, of elements $a, b \in \mathbb{Z}_p$, and of an element $W \in \mathbb{G}$, and a random coin of $\mathcal{A}$.*

2.2 Pseudorandom Function Family

A function family $\mathsf{PRF} = \{\mathsf{PRF}(K, \cdot) : \{0,1\}^{\ell(\lambda)} \to \{0,1\}^{L(\lambda)}\}_{K \in \{0,1\}^\lambda}$ is called a pseudorandom function (PRF) family if for all $K \in \{0,1\}^\lambda$, $\mathsf{PRF}(K, \cdot)$ is computable in polynomial time, and for any PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{prf}_{\mathsf{PRF},\mathcal{A}}(\lambda) = |\Pr[\mathcal{A}^{\mathsf{PRF}(K,\cdot)}(\lambda) = 1 | K \xleftarrow{\$} \{0,1\}^\lambda] - \Pr[\mathcal{A}^{\mathsf{Rand}(\cdot)}(\lambda) = 1 | \mathsf{Rand} \xleftarrow{\$} \mathcal{F}(\ell, L)]|$ is negligible. Here, $\mathcal{F}(\ell, L)$ is the set of all the possible functions $F : \{0,1\}^\ell \to \{0,1\}^L$.

2.3 Commitment

A commitment scheme $\mathcal{COM}$ consists of two algorithms ($\mathsf{COM.Setup}, \mathsf{Commit}$). The setup algorithm $\mathsf{COM.Setup}$ takes 1^λ and outputs a committing key ck. The committing algorithm Commit takes ck and a message m, and returns a commitment $c \leftarrow \mathsf{Commit}_{\mathsf{ck}}(m; r)$ where r is a randomness. We say that a commitment scheme is secure if it satisfies hiding and binding. In this thesis, we use a commitment scheme with computational hiding and statistical binding. These security notions are defined as follows.

Definition 2.3.1 (Computational Hiding)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for a commitment scheme $\mathcal{COM}$. We define an experiment* $\mathsf{Exp}^{hiding}_{\mathcal{COM},\mathcal{A}}(\lambda)$ *as follows.*

$$\begin{aligned} \mathsf{Exp}^{hiding}_{\mathcal{COM},\mathcal{A}}(\lambda): \quad & \mathsf{ck} \leftarrow \mathsf{COM.Setup}(1^\lambda);\ (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{ck}) \\ & b \xleftarrow{\$} \{0,1\};\ r \xleftarrow{\$} \{0,1\}^\lambda;\ c^* \leftarrow \mathsf{Commit}_{\mathsf{ck}}(m_b; r);\ \widetilde{b} \leftarrow \mathcal{A}_2(\mathsf{st}, c^*) \end{aligned}$$

$$\text{Return 1 if } b = \widetilde{b}, \text{ otherwise return 0}$$

We say that $\mathcal{COM}$ is computational hiding if the advantage

$$\mathsf{Adv}^{hiding}_{\mathcal{COM},\mathcal{A}}(\lambda) = \left| \ \Pr[\ \mathsf{Exp}^{hiding}_{\mathcal{COM},\mathcal{A}}(\lambda) = 1\ ] - 1/2 \ \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

Definition 2.3.2 (Statistical Binding)**.** *Let $\mathcal{A}$ be an adversary for a commitment scheme $\mathcal{COM}$. We define an experiment $\mathsf{Exp}^{binding}_{\mathcal{COM},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{binding}_{\mathcal{COM},\mathcal{A}}(\lambda): \quad & \mathsf{ck} \leftarrow \mathsf{COM.Setup}(1^\lambda);\ (m, r, \widehat{m}, \widehat{r}) \leftarrow \mathcal{A}(\mathsf{ck});\\ & c \leftarrow \mathsf{Commit}_{\mathsf{ck}}(m; r);\ \widehat{c} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(\widehat{m}; \widehat{r})\\ & \text{Return 1 if } c = \widehat{c} \ \wedge\ m \neq \widehat{m}, \text{ otherwise return 0}\end{aligned}$$

We say that $\mathcal{COM}$ is statistical binding if the advantage

$$\mathsf{Adv}^{binding}_{\mathcal{COM},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{binding}_{\mathcal{COM},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any unbounded adversary $\mathcal{A}$.

2.4 Digital Signature

A signature scheme $\mathcal{SIG}$ consists of three algorithms $(\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify})$. The key generation algorithm $\mathsf{SIG.Gen}$ takes 1^λ and outputs a verification/signing key pair $(\mathsf{vk}, \mathsf{sk})$. The signing algorithm $\mathsf{SIG.Sign}$ takes sk and a message $\mathsf{msg} \in \mathcal{M}_{\mathcal{SIG}}$, and outputs a signature σ on msg where $\mathcal{M}_{\mathcal{SIG}}$ is the message space of $\mathcal{SIG}$. The verification algorithm $\mathsf{SIG.Verify}$ takes vk, msg, and σ, and outputs 1 or 0. We say that a signature scheme is correct if for all the messages $\mathsf{msg} \in \mathcal{M}_{\mathcal{SIG}}$, it holds that

$$\Pr[\ \mathsf{SIG.Verify}(\mathsf{vk}, (\mathsf{msg}, \sigma)) = 1 \mid (\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{SIG.Gen}(1^\lambda);\ \sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, \mathsf{msg})\ ] = 1.$$

In the following, we define existential unforgeability against chosen message attacks (EUF-CMA security) for signature schemes.

Definition 2.4.1 (EUF-CMA Security)**.** *Let $\mathcal{A}$ be an adversary for a signature scheme $\mathcal{SIG}$. We define an experiment $\mathsf{Exp}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{A}}(\lambda): \quad & \mathsf{ML} \leftarrow \emptyset;\ (\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{SIG.Gen}(1^\lambda);\ (\mathsf{msg}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})\\ & \text{Return 1 if } \mathsf{SIG.Verify}(\mathsf{vk}, (\mathsf{msg}^*, \sigma^*)) = 1 \ \wedge\ \mathsf{msg}^* \notin \mathsf{ML}\end{aligned}$$

else return 0

In this experiment, the oracle Sign *takes a message* msg*, computes* $\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, \mathsf{msg})$*, adds* msg *to the list* ML*, and returns* σ*.*

We say that $\mathcal{SIG}$ *satisfies EUF-CMA security if the advantage*

$$\mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$*.*

2.5 Symmetric Key Encryption

A symmetric key encryption (SKE) scheme $\mathcal{SKE}$ consists of three algorithms ($\mathsf{SKE.Gen}$, $\mathsf{SKE.Enc}, \mathsf{SKE.Dec}$). The key generation algorithm $\mathsf{SKE.Gen}$ takes 1^λ, and outputs a secret key k. The encryption algorithm $\mathsf{SKE.Enc}$ takes k and a message $\mathsf{msg} \in \mathcal{M}_{\mathcal{SKE}}$, and outputs a ciphertext ct where $\mathcal{M}_{\mathcal{SKE}}$ is the message space of $\mathcal{SKE}$. In this thesis, if necessary, we explicitly mention a randomness $r \in \mathcal{R}_{\mathcal{SKE}}$ used in the encryption and write $\mathsf{ct} \leftarrow \mathsf{SKE.Enc}(k, \mathsf{msg}; r)$ where $\mathcal{R}_{\mathcal{SKE}}$ is the randomness space of $\mathcal{SKE}$. The decryption algorithm $\mathsf{SKE.Dec}$ takes k and ct, and outputs a message msg or a special symbol $\perp$ that indicates failure of decryption. We say that a SKE scheme is correct if for all plaintexts msg and all the randomness r, it holds that

$$\Pr[\mathsf{msg} = \widetilde{\mathsf{msg}} \mid k \leftarrow \mathsf{SKE.Gen}(1^\lambda);\ \widetilde{\mathsf{msg}} \leftarrow \mathsf{SKE.Dec}(k, \mathsf{SKE.Enc}(k, \mathsf{msg}; r))] = 1.$$

In the following, we define ciphertext-pseudorandomness and key-robustness for SKE schemes which we use in this thesis.

Firstly, we define ciphertext-pseudorandomness. Intuitively, this security notion ensures that an adversary without a secret key cannot distinguish between a ciphertext generated by the $\mathsf{SKE.Enc}$ algorithm and a random value over the ciphertext space.

Definition 2.5.1 (Ciphertext-pseudorandomness)**.** *Let* $\mathcal{A}$ *be an adversary for a SKE scheme* $\mathcal{SKE}$*. We define experiments* $\mathsf{Exp}^{real}_{\mathcal{SKE},\mathcal{A}}(\lambda)$ *and* $\mathsf{Exp}^{simulate}_{\mathcal{SKE},\mathcal{A}}(\lambda)$ *as follows.*

$\mathsf{Exp}^{real}_{\mathcal{SKE},\mathcal{A}}(\lambda)$:	$\mathsf{Exp}^{simulate}_{\mathcal{SKE},\mathcal{A}}(\lambda)$:
$k \leftarrow \mathsf{SKE.Gen}(1^\lambda)$	$b \leftarrow \mathcal{A}^{\mathsf{CRand}(\cdot)}(\lambda)$
$b \leftarrow \mathcal{A}^{\mathsf{Enc}(k,\cdot)}(\lambda)$	Return b
Return b	

In this experiment, the oracle Enc *takes* msg*, computes* $\mathsf{ct} \leftarrow \mathsf{SKE.Enc}(k, \mathsf{msg})$*, and returns* ct*. The oracle* CRand *takes a message* msg*, samples a value* ct *from the ciphertext space at random, and outputs* ct *as a response.*

We say that $\mathcal{SKE}$ satisfies ciphertext-pseudorandomness if the advantage

$$\mathsf{Adv}^{random}_{\mathcal{SKE},\mathcal{A}}(\lambda) = \left| \Pr[\ \mathsf{Exp}^{real}_{\mathcal{SKE},\mathcal{A}}(\lambda) = 1\ ] - \Pr[\ \mathsf{Exp}^{simulate}_{\mathcal{SKE},\mathcal{A}}(\lambda) = 1\ ] \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

Next, we give the definition of key-robustness for SKE schemes. Intuitively, key-robustness ensures that for any two random keys it is hard to find a ciphertext that is not decrypted to $\perp$ under both keys. Our definition can be regarded as an analogue of robustness in the public-key setting defined by Abdalla, Bellare, and Neven [5]. We note that key-robustness we define is different from robustness for SKE schemes that was introduced by Hoang, Krovetz, and Rogaway [65].

Definition 2.5.2 (Key-robustness)**.** *Let $\mathcal{A}$ be an adversary for a SKE scheme $\mathcal{SKE}$. We define an experiment $\mathsf{Exp}^{key\text{-}robust}_{\mathcal{SKE},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{key\text{-}robust}_{\mathcal{SKE},\mathcal{A}}(\lambda):\quad & k, \widetilde{k} \leftarrow \mathsf{SKE.Gen}(1^\lambda);\ \mathsf{ct}^* \leftarrow \mathcal{A}(k, \widetilde{k}) \\ & \text{Return 1 if } \mathsf{SKE.Dec}(k, \mathsf{ct}^*) \neq \perp \wedge \mathsf{SKE.Dec}(\widetilde{k}, \mathsf{ct}^*) \neq \perp \\ & \text{else return 0}\end{aligned}$$

We say that $\mathcal{SKE}$ satisfies key-robustness if the advantage

$$\mathsf{Adv}^{key\text{-}robust}_{\mathcal{SKE},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{key\text{-}robust}_{\mathcal{SKE},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

2.6 Public Key Encryption

A public key encryption (PKE) scheme $\mathcal{PKE}$ consists of three algorithms ($\mathsf{PKE.Gen}$, $\mathsf{PKE.Enc}, \mathsf{PKE.Dec}$). The key generation algorithm $\mathsf{PKE.Gen}$ takes 1^λ and outputs an encryption/decryption key pair $(\mathsf{ek}, \mathsf{dk})$. The encryption algorithm $\mathsf{PKE.Enc}$ takes ek and a plaintext $\mathsf{msg} \in \mathcal{M}_{\mathcal{PKE}}$, and outputs a ciphertext ct where $\mathcal{M}_{\mathcal{PKE}}$ is the message space of $\mathcal{PKE}$. In this thesis, if necessary, we explicitly mention a randomness $r \in \mathcal{R}_{\mathcal{PKE}}$ used in the encryption and write $\mathsf{ct} \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}, \mathsf{msg}; r)$ where $\mathcal{R}_{\mathcal{PKE}}$ is the randomness space of $\mathcal{PKE}$. The decryption algorithm $\mathsf{PKE.Dec}$ takes dk and ct, and outputs msg or a special symbol $\perp$ that indicates failure of decryption. We say that a PKE scheme is correct if for all plaintexts msg and all the randomness r, it holds that

$$\Pr[\mathsf{msg} = \widetilde{\mathsf{msg}} \mid (\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda);\ \widetilde{\mathsf{msg}} \leftarrow \mathsf{PKE.Dec}(\mathsf{dk}, \mathsf{PKE.Enc}(\mathsf{ek}, \mathsf{msg}; r))] = 1.$$

In the following, we define indistinguishability against chosen plaintext attacks (IND-CPA security) and key privacy for PKE schemes.

Definition 2.6.1 (IND-CPA Security)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for a PKE scheme $\mathcal{PKE}$. We define an experiment $\mathsf{Exp}^{\textit{ind-cpa}}_{\mathcal{PKE},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{\textit{ind-cpa}}_{\mathcal{PKE},\mathcal{A}}(\lambda):\quad & (\mathsf{ek},\mathsf{dk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda);\ (\mathsf{st},\mathsf{msg}_0,\mathsf{msg}_1) \leftarrow \mathcal{A}_1(\mathsf{ek}) \\ & b \xleftarrow{\$} \{0,1\};\ \mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek},\mathsf{msg}_b);\ \widetilde{b} \leftarrow \mathcal{A}_2(\mathsf{st},\mathsf{ct}^*) \\ & \text{Return 1 if } b = \widetilde{b}, \text{ otherwise return 0}\end{aligned}$$

We say that $\mathcal{PKE}$ satisfies IND-CPA security if the advantage

$$\mathsf{Adv}^{\textit{ind-cpa}}_{\mathcal{PKE},\mathcal{A}}(\lambda) = \left|\ \Pr[\ \mathsf{Exp}^{\textit{ind-cpa}}_{\mathcal{PKE},\mathcal{A}}(\lambda) = 1\ ] - 1/2\ \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

Definition 2.6.2 (Key Privacy)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for a PKE scheme $\mathcal{PKE}$. We define an experiment $\mathsf{Exp}^{\textit{key-priv}}_{\mathcal{PKE},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{\textit{key-priv}}_{\mathcal{PKE},\mathcal{A}}(\lambda):\quad & (\mathsf{ek}_0,\mathsf{dk}_0) \leftarrow \mathsf{PKE.Gen}(1^\lambda);\ (\mathsf{ek}_1,\mathsf{dk}_1) \leftarrow \mathsf{PKE.Gen}(1^\lambda) \\ & (\mathsf{st},\mathsf{msg}^*) \leftarrow \mathcal{A}_1(\mathsf{ek}_0,\mathsf{ek}_1);\ b \xleftarrow{\$} \{0,1\} \\ & \mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_b,\mathsf{msg}^*);\ \widetilde{b} \leftarrow \mathcal{A}_2(\mathsf{st},\mathsf{ct}^*) \\ & \text{Return 1 if } b = \widetilde{b}, \text{ otherwise return 0}\end{aligned}$$

We say that $\mathcal{PKE}$ satisfies key privacy if the advantage

$$\mathsf{Adv}^{\textit{key-priv}}_{\mathcal{PKE},\mathcal{A}}(\lambda) = \left|\ \Pr[\ \mathsf{Exp}^{\textit{key-priv}}_{\mathcal{PKE},\mathcal{A}}(\lambda) = 1\ ] - 1/2\ \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

2.7 Identity-based Encryption

An identity-based encryption (IBE) scheme $\mathcal{IBE}$ consists of four algorithms ($\mathsf{IBE.Gen}$, $\mathsf{IBE.Ext}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec}$). The key generation algorithm $\mathsf{IBE.Gen}$ takes 1^λ and outputs system parameters params and a master secret key msk. The key extraction algorithm $\mathsf{IBE.Ext}$ takes params, msk, and an arbitrary string $\mathsf{ID} \in \{0,1\}^*$, and outputs a decryption key dk that is the corresponding decryption key with the public key ID. The encryption algorithm $\mathsf{IBE.Enc}$ takes params, ID, and a plaintext $\mathsf{msg} \in \mathcal{M}_{\mathcal{IBE}}$, and outputs a ciphertext ct where $\mathcal{M}_{\mathcal{IBE}}$ is the message space of $\mathcal{IBE}$. As the case of PKE schemes,

if necessary, we explicitly mention a randomness $r \in \mathcal{R}_{\mathcal{IBE}}$ used in the encryption and write $\mathsf{ct} \leftarrow \mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}, \mathsf{msg}; r)$ where $\mathcal{R}_{\mathcal{IBE}}$ is the randomness space of $\mathcal{IBE}$. The decryption algorithm $\mathsf{IBE.Dec}$ takes params, dk, and ct, and outputs msg. We say that an IBE scheme is correct if for all strings ID, all the plaintexts msg, and all the randomness r, it holds that

$$\Pr[\ \mathsf{msg} = \widetilde{\mathsf{msg}} \quad | \quad (\mathsf{params}, \mathsf{msk}) \leftarrow \mathsf{IBE.Gen}(1^\lambda);\ \mathsf{dk} \leftarrow \mathsf{IBE.Ext}(\mathsf{params}, \mathsf{msk}, \mathsf{ID});\ \widetilde{\mathsf{msg}} \leftarrow \mathsf{IBE.Dec}(\mathsf{dk}, \mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}, \mathsf{msg}; r))\ ] = 1.$$

Definition 2.7.1 (IND-ID-CPA Security)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for an IBE scheme $\mathcal{IBE}$. We define an experiment $\mathsf{Exp}^{\textit{ind-id-cpa}}_{\mathcal{IBE},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}
\mathsf{Exp}^{\textit{ind-id-cpa}}_{\mathcal{IBE},\mathcal{A}}(\lambda):\quad & \mathsf{IDSet} \leftarrow \emptyset;\ (\mathsf{params}, \mathsf{msk}) \leftarrow \mathsf{IBE.Gen}(1^\lambda) \\
& (\mathsf{st}, \mathsf{ID}^*, \mathsf{msg}_0, \mathsf{msg}_1) \leftarrow \mathcal{A}_1^{\mathsf{Extract}(\cdot)}(\mathsf{params}) \\
& \text{If } \mathsf{ID}^* \in \mathsf{IDSet},\ \text{return } 0 \\
& b \xleftarrow{\$} \{0,1\};\ \mathsf{ct}^* \leftarrow \mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}^*, \mathsf{msg}_b) \\
& \widetilde{b} \leftarrow \mathcal{A}_2^{\mathsf{Extract}(\cdot)}(\mathsf{st}, \mathsf{ct}^*) \\
& \text{Return 1 if } b = \widetilde{b},\ \text{otherwise return } 0
\end{aligned}$$

In this experiment, the oracle $\mathsf{Extract}$ *takes* ID*, computes* $\mathsf{dk} \leftarrow \mathsf{IBE.Ext}(\mathsf{params}, \mathsf{msk}, \mathsf{ID})$*, adds* ID *to the list* IDSet*, and returns* dk*. We note that it is not allowed to query the identity* ID^* *to the* $\mathsf{Extract}$ *oracle.*

We say that $\mathcal{IBE}$ satisfies IND-ID-CPA security if the advantage

$$\mathsf{Adv}^{\textit{ind-id-cpa}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = \left|\ \Pr[\ \mathsf{Exp}^{\textit{ind-id-cpa}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = 1\ ] - 1/2\ \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

Definition 2.7.2 (Key Privacy)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for an IBE scheme $\mathcal{IBE}$. We define an experiment $\mathsf{Exp}^{\textit{key-priv}}_{\mathcal{IBE},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}
\mathsf{Exp}^{\textit{key-priv}}_{\mathcal{IBE},\mathcal{A}}(\lambda):\quad & \mathsf{IDSet} \leftarrow \emptyset;\ (\mathsf{params}, \mathsf{msk}) \leftarrow \mathsf{IBE.Gen}(1^\lambda) \\
& (\mathsf{st}, \mathsf{ID}_0, \mathsf{ID}_1, \mathsf{msg}^*) \leftarrow \mathcal{A}_1^{\mathsf{Extract}(\cdot)}(\mathsf{params}) \\
& \text{If } \{\mathsf{ID}_0, \mathsf{ID}_1\} \cap \mathsf{IDSet} \neq \emptyset,\ \text{return } 0 \\
& b \xleftarrow{\$} \{0,1\};\ \mathsf{ct}^* \leftarrow \mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}_b, \mathsf{msg}^*) \\
& \widetilde{b} \leftarrow \mathcal{A}_2^{\mathsf{Extract}(\cdot)}(\mathsf{st}, \mathsf{ct}^*) \\
& \text{Return 1 if } b = \widetilde{b},\ \text{otherwise return } 0
\end{aligned}$$

In this experiment, the oracle $\mathsf{Extract}$ *takes* ID*, computes* $\mathsf{dk} \leftarrow \mathsf{IBE.Ext}(\mathsf{params}, \mathsf{msk}, \mathsf{ID})$*, adds* ID *to the list* IDSet*, and returns* dk*. We note that it is not allowed to query the identities* ID_0 *and* ID_1 *to the* $\mathsf{Extract}$ *oracle.*

We say that $\mathcal{IBE}$ *satisfies key privacy if the advantage*

$$\mathsf{Adv}^{\textit{key-priv}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = \left| \ \Pr[\ \mathsf{Exp}^{\textit{key-priv}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = 1\ ] - 1/2 \ \right|$$

is negligible for any PPT adversary $\mathcal{A}$*.*

2.8 Non-interactive Proof

Let R_L be an efficiently computable binary relation. For a pair $(x, w) \in R_L$, we call x a statement and w a witness. Let L be the language consisting of statements in R_L. A non-interactive proof system $\mathcal{P}_L$ for a language L consists of three algorithms ($\mathsf{ZK.Gen}$, $\mathsf{ZK.Prove}, \mathsf{ZK.Verify}$). The key generation algorithm $\mathsf{ZK.Gen}$ takes 1^λ and outputs a common reference string crs. The prove algorithm $\mathsf{ZK.Prove}$ takes crs, a statement x, and a witness w, and outputs a proof π. The verification algorithm $\mathsf{ZK.Verify}$ takes crs, x, and π, and outputs either 1 or 0. A non-interactive proof system is required the following two conditions:

Completeness: For all $(x, w) \in R_L$ and all $\mathsf{crs} \leftarrow \mathsf{ZK.Gen}(1^\lambda)$, it holds that $\Pr[\ \mathsf{ZK.Verify}(\mathsf{crs}, x, \pi) = 1 \mid \pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, x, w)\ ] = 1$.

Soundness: For any PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\textit{sound}}_{\mathcal{P}_L,\mathcal{A}}(\lambda) = \Pr[x^* \notin L \wedge \mathsf{ZK.Verify}(\mathsf{crs}, x^*, \pi^*) = 1 \mid \mathsf{crs} \leftarrow \mathsf{ZK.Gen}(1^\lambda); (x^*, \pi^*) \leftarrow \mathcal{A}(\mathsf{crs})]$ is negligible.

In the following, we define zero-knowledge for non-interactive proof systems.

Definition 2.8.1 (Zero-knowledge)**.** *Let* $\mathcal{A}$ *be an adversary and* $\mathcal{S} = (\mathtt{Sim}_1, \mathtt{Sim}_2)$ *be a simulator for a non-interactive proof system* $\mathcal{P}_L$*. We define experiments* $\mathsf{Exp}^{\textit{proof}}_{\mathcal{P}_L,\mathcal{A}}(\lambda)$ *and* $\mathsf{Exp}^{\textit{sim-proof}}_{\mathcal{P}_L,\mathcal{A}}(\lambda)$ *as follows.*

$\mathsf{Exp}^{\textit{proof}}_{\mathcal{P}_L,\mathcal{A}}(\lambda)$:	$\mathsf{Exp}^{\textit{sim-proof}}_{\mathcal{P}_L,\mathcal{A}}(\lambda)$:
$\mathsf{crs} \leftarrow \mathsf{ZK.Gen}(1^\lambda)$	$(\mathsf{crs}, \mathsf{td}) \leftarrow \mathtt{Sim}_1(1^\lambda)$
$b \leftarrow \mathcal{A}^{\mathsf{Prove}(\cdot,\cdot)}(\mathsf{crs})$	$b \leftarrow \mathcal{A}^{\mathsf{SimProve}(\cdot,\cdot)}(\mathsf{crs})$
Return b	Return b

In this experiment, the oracle Prove *takes* (x, w)*, computes* $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, x, w)$*, and returns* π*. The oracle* $\mathsf{SimProve}$ *takes* (x, w)*, computes* $\pi \leftarrow \mathtt{Sim}_2(\mathsf{crs}, \mathsf{td}, x)$*, and returns* π*. If* $(x, w) \notin R_L$*, then* $\mathsf{SimProve}$ *returns* $\perp$*.*

We say that $\mathcal{P}_L$ is zero-knowledge if for any PPT adversary $\mathcal{A}$ there exists a simulator $\mathcal{S} = (\mathtt{Sim}_1, \mathtt{Sim}_2)$ such that the advantage

$$\mathsf{Adv}^{zk}_{\mathcal{P}_L,\mathcal{A}}(\lambda) = \left| \Pr[\ \mathsf{Exp}^{proof}_{\mathcal{P}_L,\mathcal{A}}(\lambda) = 1\ ] - \Pr[\ \mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}_L,\mathcal{A}}(\lambda) = 1\ ] \right|$$

is negligible.

The Groth-Sahai Proof System. Groth and Sahai [63] introduced a framework for very efficient non-interactive proof for the satisfiability of relations in bilinear groups, including pairing product equations [63]. The proof system consists of algorithms (K_{NI}, $\mathsf{P}, \mathsf{V}, \mathsf{X}$). The algorithm K_{NI} takes a group parameter $\mathsf{gk} = (p, \mathbb{G}, \mathbb{G}_T, e, G)$ as input and outputs a common reference string $\mathsf{crs} = (F, H, U, V, W, U', V', W') \in \mathbb{G}^8$ with an extraction key xk, which can extract a witness from a proof. The algorithm P takes crs, an equation description x, and its witness w as input and outputs a proof π. This proof can be verified by running $\mathsf{V}(\mathsf{crs}, x, \pi)$. The algorithm $\mathsf{X}_{\mathsf{xk}}(\mathsf{crs}, x, \pi)$ extracts a witness w from the proof π.

There are two types of the Groth-Sahai proof systems, the algorithms ($\mathsf{K}_{\mathrm{NI}}, \mathsf{P}_{\mathrm{NIWI}}$, $\mathsf{V}_{\mathrm{NIWI}}, \mathsf{X}_{\mathrm{NIWI}}$) provides witness-indistinguishability and the algorithms ($\mathsf{K}_{\mathrm{NI}}, \mathsf{P}_{\mathrm{NIZK}}, \mathsf{V}_{\mathrm{NIZK}}$, $\mathsf{X}_{\mathrm{NIZK}}$) provides zero-knowledge. The two types of proof can share a single common reference string. (Thus, multiple systems can use a common K_{NI}.) There exists a simulator that outputs a simulated common reference string crs and a trapdoor key tk. These simulated common reference strings are computationally indistinguishable from the common reference strings produced by K. We say a proof system is perfect witness-indistinguishable, if, on a simulated common reference string, the proof π does not reveal anything about which witness was used by the prover when creating the proof. We say a proof system is perfect zero-knowledge, if there exists a simulator that produces a simulated proof and the simulated proof is perfectly indistinguishable from the proof which is produced by using a witness and a simulated common reference string.

In the Groth-Sahai proof system, to prove that committed variables satisfy a set of relations, the prover computes one commitment per variable and one proof element per relation. The non-interactive zero-knowledge (NIZK) proofs are available for pairing product equations, which are relations of the type

$$\prod_{i=1}^{n} e(\mathcal{A}_i, \mathcal{X}_i) \cdot \prod_{i=1}^{n} \prod_{j=1}^{n} e(\mathcal{X}_i, \mathcal{X}_j)^{a_{ij}} = t_T$$

with $t_T = 1$ for variables $\mathcal{X}_1, \ldots, \mathcal{X}_n \in \mathbb{G}$, and constants $\mathcal{A}_1, \ldots, \mathcal{A}_n \in \mathbb{G}$ and a_{ij} for $i, j \in \{1, \ldots n\}$. Even if $t_T \neq 1$, still we can construct NIZK proofs if t_T can be decomposed to known base group elements $\tilde{G}, \hat{G} \in \mathbb{G}$ such that $t_T = e(\tilde{G}, \hat{G})$. NIZK proofs also can be

constructed for multi-scalar multiplication equations, which are of the form

$$\prod_{i=1}^{m} \mathcal{A}_i^{y_i} \cdot \prod_{j=1}^{n} \mathcal{X}_j^{b_j} \cdot \prod_{i=1}^{m} \prod_{j=1}^{n} \mathcal{X}_j^{y_i \gamma_{ij}} = T$$

for variables $\mathcal{X}_1, \ldots, \mathcal{X}_n \in \mathbb{G}, y_1, \ldots, y_m \in \mathbb{Z}_p$ and constants $T, \mathcal{A}_1, \ldots, \mathcal{A}_m \in \mathbb{G}, b_1, \ldots, b_m \in \mathbb{Z}_p$, and $\gamma_{ij} \in \mathbb{G}$, for $i \in \{1, \ldots, m\}$ and $j \in \{1, \ldots, n\}$.

Chapter 3

Static Group Signature

In this chapter, we review the model of static group signature by Bellare, Micciancio, and Warinschi [14]. In the static setting, the number of members is fixed in the setup phase, and a member cannot join after the setup.

We firstly review the syntax of the Bellare-Micciancio-Warinschi (BMW) model in Section 3.1. Then, we define some security notions for static group signature in Section 3.2. Lastly, in Section 3.3, we introduce the construction by Bellare et al. [14], which shows that a static group signature scheme can be constructed from a trapdoor permutation.

3.1 Syntax

A group signature scheme $\mathcal{GS}$ consists of the following four algorithms ($\mathsf{GS.Gen}$, $\mathsf{GS.Sign}$, $\mathsf{GS.Verify}$, $\mathsf{GS.Open}$).

$\mathsf{GS.Gen}$**:** The key generation algorithm takes 1^λ and 1^n where n is the number of users, and outputs a group public key gpk, an opening key ok, and a collection of user signing keys $\mathbf{gsk} = \{\mathsf{gsk}_1, \ldots, \mathsf{gsk}_n\}$ where gsk_i indicates a signing key of user i.

$\mathsf{GS.Sign}$**:** The signing algorithm takes gsk_i and a message $\mathsf{msg} \in \mathcal{M}_{\mathcal{GS}}$ and outputs a group signature Σ where $\mathcal{M}_{\mathcal{GS}}$ is the message space of $\mathcal{GS}$.

$\mathsf{GS.Verify}$**:** The verification algorithm takes gpk, msg, and Σ, and outputs 1 or 0.

$\mathsf{GS.Open}$**:** The opening algorithm takes gpk, ok, msg, and Σ, and outputs a user ID i or $\emptyset$ or $\perp$. The symbol $\emptyset$ indicates that the group signature Σ is invalid and the symbol $\perp$ indicates failure of opening.

We say that a group signature scheme is correct if for all the user identities i and all the messages $\mathsf{msg} \in \mathcal{M}_{\mathcal{GS}}$, it holds that

$$\Pr[\ \mathsf{GS.Verify}(\mathsf{gpk}, (\mathsf{msg}, \Sigma)) = 0 \ \vee \ i \neq \widetilde{i} \ \mid \begin{array}{l} (\mathsf{gpk}, \mathsf{ok}, \mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda, 1^n) \\ \Sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_i, \mathsf{msg}) \\ \widetilde{i} \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, (\mathsf{msg}, \Sigma)) \end{array}]$$

is negligible.

3.2 Security

In this section, we define security requirements for static group signature schemes: full anonymity, selfless anonymity, and traceability. The definitions of full anonymity and traceability are identical to those in the BMW model (strictly speaking, traceability in this thesis corresponds to full traceability in Bellare et al.'s paper [14]). Selfless anonymity is a strictly weaker security notion than full anonymity.

Firstly, we give the definition of full anonymity. Intuitively, anonymity requires that no one can extract the signer's information from a signature except for the opener. Especially, full anonymity ensures that anonymity holds under the condition that an adversary possesses all the user signing keys. In the following definition, an adversary is also allowed to access the opening oracle, which returns the opening result for a signature.

Definition 3.2.1 (Full Anonymity)**.** *Let $A = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{\mathit{full\text{-}anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n)$ as follows.*

$$\begin{array}{ll} \mathsf{Exp}^{\mathit{full\text{-}anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n): & (\mathsf{gpk}, \mathsf{ok}, \mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda, 1^n) \\ & (\mathsf{st}, i_0, i_1, \mathsf{msg}^*) \leftarrow \mathcal{A}_1^{\mathsf{Open}(\cdot,\cdot)}(\mathsf{gpk}, \mathbf{gsk}) \\ & b \xleftarrow{\$} \{0,1\};\ \Sigma^* \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_{i_b}, \mathsf{msg}^*) \\ & \widetilde{b} \leftarrow \mathcal{A}_2^{\mathsf{Open}(\cdot)}(\mathsf{st}, \Sigma^*) \\ & \text{Return 1 if } b = \widetilde{b}, \text{ otherwise return 0} \end{array}$$

In this experiment, the oracle Open *takes a message* msg *and a signature Σ, computes $i \leftarrow \mathsf{GS.Open}(\mathsf{ok}, \mathsf{msg}, \Sigma)$, and returns i.*

We say that $\mathcal{GS}$ satisfies selfless anonymity if the advantage

$$\mathsf{Adv}^{\mathit{full\text{-}anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n) = \left| \ \Pr[\ \mathsf{Exp}^{\mathit{full\text{-}anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n) = 1\ ] - 1/2 \ \right|$$

is negligible for any polynomial $n = n(\lambda)$ and any PPT adversary $\mathcal{A}$.

Secondly, we define selfless anonymity. Selfless anonymity is originally introduced by Boneh and Shacham [28] as the security notion for group signature with verifier-local revocation. Selfless anonymity is a strictly weaker security notion than full anonymity, and ensures that the anonymity of a signature only against an adversary who does not possess the user signing key which was used in the generation of the corresponding signature. In the following definition, an adversary is not allowed to even access the opening oracle.

Definition 3.2.2 (Selfless Anonymity)**.** *Let $A = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{\textit{self-anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n)$ as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{\textit{self-anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n): \quad & (i_0, i_1, \mathsf{st}_1) \leftarrow \mathcal{A}_1(1^\lambda, 1^n) \\
& (\mathsf{gpk}, \mathsf{ok}, \mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda, 1^n);\ \mathbf{gsk}^- \leftarrow \{\mathsf{gsk}_i\}_{i \notin \{i_0, i_1\}} \\
& (\mathsf{st}_2, \mathsf{msg}^*) \leftarrow \mathcal{A}_2^{\mathsf{Sign}_0(\cdot), \mathsf{Sign}_1(\cdot)}(\mathsf{st}_1, \mathsf{gpk}, \mathbf{gsk}^-) \\
& b \xleftarrow{\$} \{0,1\};\ \Sigma^* \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_{i_b}, \mathsf{msg}^*); \\
& \widetilde{b} \leftarrow \mathcal{A}_3^{\mathsf{Sign}_0(\cdot), \mathsf{Sign}_1(\cdot)}(\mathsf{st}_2, \Sigma^*) \\
& \text{Return 1 if } b = \widetilde{b}, \text{ otherwise return 0}
\end{aligned}
$$

In this experiment, the oracle Sign_b takes a message msg, computes $\Sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_{i_b}, \mathsf{msg})$, and returns Σ.

We say that $\mathcal{GS}$ satisfies selfless anonymity if the advantage

$$
\mathsf{Adv}^{\textit{self-anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n) = \left| \ \Pr[\ \mathsf{Exp}^{\textit{self-anon}}_{\mathcal{GS},\mathcal{A}}(\lambda, n) = 1\ ] - 1/2 \ \right|
$$

is negligible for any polynomial $n = n(\lambda)$ and any PPT adversary $\mathcal{A}$.

Lastly, we define traceability. Traceability ensures that no colluding set of group members (even consisting of the entire group) can create a signature whose opening result is invalid, or a signature which is traced back to a member outside of the coalition. Formally, traceability is defined as follows.

Definition 3.2.3 (Traceability)**.** *Let $\mathcal{A}$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{\textit{trace}}_{\mathcal{GS},\mathcal{A}}(\lambda, n)$ as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{\textit{trace}}_{\mathcal{GS},\mathcal{A}}(\lambda, n): \quad & \mathsf{CU} \leftarrow \emptyset;\ \mathsf{QL} \leftarrow \emptyset;\ (\mathsf{gpk}, \mathsf{ok}, \mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda, 1^n) \\
& (\mathsf{msg}^*, \Sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\cdot,\cdot), \mathsf{Corrupt}(\cdot)}(\mathsf{gpk}, \mathsf{ok}) \\
& i^* \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, (\mathsf{msg}^*, \Sigma^*)) \\
& \text{If } \mathsf{GS.Verify}(\mathsf{gpk}, (\mathsf{msg}^*, \Sigma^*)) = 0, \text{ then return 0} \\
& \text{If } i^* = \perp, \text{ then return 1}
\end{aligned}
$$

Return 1 if $i^* \notin \mathsf{CU} \wedge (i^*, \mathsf{msg}^*) \notin \mathsf{QL}$, else return 0

In this experiment, the oracle Sign *takes* (i, msg)*, computes* $\Sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_i, \mathsf{msg})$*, adds* (i, msg) *to the list* QL*, and returns* Σ*. The oracle* $\mathsf{Corrupt}$ *takes* $i \in [1, n]$*, adds* i *to the list* CU*, and returns* $\mathsf{gsk}[i]$*.*

We say that $\mathcal{GS}$ *satisfies traceability if the advantage*

$$\mathsf{Adv}^{trace}_{\mathcal{GS},\mathcal{A}}(\lambda, n) = \Pr[\ \mathsf{Exp}^{trace}_{\mathcal{GS},\mathcal{A}}(\lambda, n) = 1\ ]$$

is negligible for any polynomial $n = n(\lambda)$ *and any PPT adversary* $\mathcal{A}$*.*

3.3 The Bellare-Micciancio-Warinschi Construction

In this section, we review the BMW construction [14], which allows to construct a group signature scheme from a PKE scheme, a signature scheme, and a non-interactive zero-knowledge (NIZK) proof system. The resulting group signature scheme satisfies full anonymity and traceability defined in Section 3.2.

In the BMW construction, the group manager possesses a key pair of a signature scheme $(\mathsf{vk}, \mathsf{sk})$ and a key pair of a PKE scheme $(\mathsf{PK}, \mathsf{SK})$. Each user i possesses a key pair of a signature scheme $(\mathsf{vk}_i, \mathsf{sk}_i)$. Each pair of user ID i and his verification key vk_i are certified by the group manager who computes a signature cert_i on the message $\langle i, \mathsf{vk}_i \rangle$ using sk as a certificate. To produce a group signature for a message m, a user i firstly computes a signature s on m under his signing key sk_i. Secondly, the user i encrypts the message $\langle i, \mathsf{vk}_i, \mathsf{cert}_i, s \rangle$ to a ciphertext C under PK. Also, the user i makes a proof π that the signature s and the ciphertext C are honestly generated, and outputs $\sigma = (C, \pi)$ as a group signature. More precisely, for a statement $x = \langle \mathsf{PK}, \mathsf{vk}, m, C \rangle$ and a witness $w = \langle i, \mathsf{vk}_i, \mathsf{cert}_i, s, R \rangle$, the user i proves the equations (a) $\mathsf{PKE.Enc}(\mathsf{PK}, \langle i, \mathsf{vk}_i, \mathsf{cert}_i, s \rangle; R) = C$, (b) $\mathsf{SIG.Verify}(\mathsf{vk}, (\langle i, \mathsf{vk}_i \rangle, \mathsf{cert}_i)) = 1$, and (c) $\mathsf{SIG.Verify}(\mathsf{vk}_i, (m, s)) = 1$ by the NIZK proof system. The group signature σ is accepted if the proof π is accepted. When the opener needs to identify the signer of the group signature $\sigma = (C, \pi)$, he can extract the signer's ID i by decrypting the ciphertext C using SK.

Intuitively, the scheme is anonymous because of the security of the PKE scheme and the zero-knowledge property of the NIZK proof system. Moreover, since a signing key $(i, \mathsf{sk}_i, \mathsf{cert}_i)$ is independent of the PKE scheme and the NIZK proof system, the scheme is still anonymous even if an adversary has all the user signing keys. Therefore, group signature schemes constructed by the BMW construction satisfy full anonymity. In addition, the traceability comes from the soundness of the NIZK proof system and the EUF-CMA security of the signature scheme.

Chapter 4

Dynamic Group Signature

Compared to the static setting, in the dynamic setting, neither the number nor the identities of members are fixed or known in the setup phase. Bellare, Shi, and Zhang [17] and Kiayias and Yung [74] independently developed security models for dynamic group signatures. After that, Sakai, Schuldt, Emura, Hanaoka, and Ohta [100] showed that there is a room for improving the Bellare-Shi-Zhang (BSZ) model and considered a new security notion called opening soundness to prevent a signature hijacking attack.

In this chapter, we review the model by Sakai et al. [100] (but we also call this "the Bellare-Shi-Zhang model" since this is based on the BSZ model [17]). This security model considers the strong security model. More precisely, the authority is separated into two roles: the opener, who identifies the signer of a signature and the issuer, who generates user signing keys. This separation provides the strong security level in the face of the possibility that authorities can be dishonest. Furthermore, in order to be protected against a fully corrupt opener, the opener is required to generate a publicly verifiable proof attached to an opening result. This proof ensures the validity of the opening result.

We firstly review the syntax of the BSZ model [17, 100] in Section 4.1. Secondly, we give the definitions of security notions in Section 4.2. Finally, in Section 4.3, we introduce the modified Groth scheme [100], which is one of efficient schemes secure in the BSZ model.

4.1 Syntax

A group signature scheme $\mathcal{GS}$ consists of the following algorithms ($\mathsf{GS.GGen}, \mathsf{GS.UGen}$, $\mathsf{GS.Join}/\mathsf{GS.Issue}, \mathsf{GS.Sign}, \mathsf{GS.Verify}, \mathsf{GS.Open}, \mathsf{GS.Judge}$).

$\mathsf{GS.GGen}$: The group key generation algorithm takes a security parameter 1^λ ($\lambda \in \mathbb{N}$), and outputs a group public key gpk, an issuing key ik, and an opening key ok.

GS.UGen: The user key generation algorithm, which is run by a user i, takes as input 1^λ and gpk, and returns a public and secret key pair $(\mathsf{upk}_i, \mathsf{usk}_i)$.

GS.Join/GS.Issue: The pair of (interactive) algorithms are run by a user i and the issuer, and takes gpk, upk_i, and usk_i from the user i, and gpk, upk_i, and ik from the issuer, respectively. If it is successful, the issuer stores the registration information of the user i in $\mathsf{reg}[i]$ and the user obtains the corresponding signing key gsk_i. We denote $\mathsf{reg} = \{\mathsf{reg}[i]\}_i$.

GS.Sign: The signing algorithm takes gpk, gsk_i, and a message msg, and returns a group signature Σ.

GS.Verify: The verification algorithm takes gpk, Σ, and msg, and returns either 1 or 0.

GS.Open: The opening algorithm takes gpk, ok, msg, Σ, and reg, and returns either (i, τ) or $\perp$ where i is a user identity and τ is a proof that the user i computed Σ. The symbol $\perp$ indicates that the opening procedure is failure.

GS.Judge: The judge algorithm takes as input gpk, i, upk_i, msg, Σ, and τ, and returns either 1 or 0.

4.2 Security

Firstly, we give the definitions of some oracles. The SndToI and SndToU oracle are interactive oracles. Also, HU and CU are the set of honest users and corrupted users, respectively.

AddU($\cdot$): The add-user oracle takes a user identity i, and runs the GS.UGen algorithm and GS.Join/GS.Issue protocol to add an honest user i to the group. The oracle returns upk_i and adds i to HU.

Corrupt($\cdot, \cdot$): The corrupt-user oracle takes as input a user identity i and upk. This oracle sets $\mathsf{upk}_i \leftarrow \mathsf{upk}$ and adds i to CU.

SndToI($\cdot$): The send-to-issuer oracle takes as input a user identity i and interacts with the adversary who corrupts the user i by running $\mathsf{GS.Issue}(\mathsf{gpk}, \mathsf{upk}_i, \mathsf{ik})$. The user i needs to be in the set CU. If $i \notin \mathsf{CU}$, the oracle outputs $\perp$.

SndToU($\cdot$): The send-to-user oracle takes as input a user identity i. At first, the oracle produces a user public and secret key pair $(\mathsf{upk}_i, \mathsf{usk}_i) \leftarrow \mathsf{GS.UGen}(1^\lambda, \mathsf{gpk})$ and adds i to HU. Then he interacts with the adversary who corrupts the issuer by running

$\mathsf{GS.Join}(\mathsf{gpk}, \mathsf{upk}_i, \mathsf{usk}_i)$. The user i needs to be neither in the set HU nor the set CU. If so, the oracle outputs $\perp$.

$\mathsf{USK}(\cdot)$**:** The user secret keys oracle takes as input i, and returns the secret keys usk_i and gsk_i if $i \in \mathsf{HU}$. If not, the oracle returns $\perp$.

$\mathsf{RReg}(\cdot)$**:** The read-registration-table oracle takes as input i, and returns $\mathsf{reg}[i]$.

$\mathsf{WReg}(\cdot, \cdot)$**:** The write-registration-table oracle takes as input i and a value $\widehat{\mathsf{reg}}$, and writes or modifies the contents of reg by setting $\mathsf{reg}[i] \leftarrow \widehat{\mathsf{reg}}$.

$\mathsf{Sign}(\cdot, \cdot)$**:** The signing oracle takes as input i and a message msg, and returns $\Sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, \mathsf{msg})$ if $i \in \mathsf{HU}$. Otherwise, the oracle returns $\perp$.

$\mathsf{Ch}(\cdot, \cdot, \cdot, \cdot)$**:** The challenge oracle takes as input a bit b, two identities i_0, i_1, and a message msg^*, and returns $\Sigma^* \leftarrow \mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_{i_b}, \mathsf{msg}^*)$ if both $i_0 \in \mathsf{HU}$ and $i_1 \in \mathsf{HU}$. If not, the oracle returns $\perp$. Here, we call b a challenge bit, m^* a challenge message, Σ^* a challenge signature, and i_0, i_1 challenge users.

$\mathsf{Open}(\cdot, \cdot)$**:** The opening oracle takes as input msg and Σ, and returns $(i, \tau) \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{msg}, \Sigma, \mathsf{reg})$ if $(\mathsf{msg}, \Sigma) \notin (\mathsf{msg}^*, \Sigma^*)$. If not, the oracle returns $\perp$.

Now, we define the security requirements for dynamic group signatures: correctness, anonymity, traceability, non-frameability, and opening soundness.

Firstly, we define correctness. Intuitively, correctness ensures that any honestly generated group signature is valid (i.e., it is accepted by the verification algorithm), and the opening algorithm correctly identifies its signer and the generated proof is accepted by the judge algorithm. Formally, correctness is defined as follows.

Definition 4.2.1 (Correctness)**.** *Let $\mathcal{A}$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{corr}_{\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{corr}_{\mathcal{GS},\mathcal{A}}(\lambda): \quad & (\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{HU} \leftarrow \emptyset;\ \mathsf{CU} \leftarrow \emptyset \\
& (i, \mathsf{msg}) \leftarrow \mathcal{A}^{\mathsf{AddU}(\cdot),\mathsf{RReg}(\cdot)}(\mathsf{gpk}) \\
& \text{If } i \notin \mathsf{HU}, \text{ return } 0 \\
& \Sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, \mathsf{msg}) \\
& (\widetilde{i}, \tau) \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{msg}, \Sigma, \mathsf{reg}) \\
& \text{Output 1 if the following holds :} \\
& \qquad \mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{msg}, \Sigma) = 0 \ \vee \ i \neq \widetilde{i} \\
& \qquad\qquad \vee \ \mathsf{GS.Judge}(\mathsf{gpk}, i, \mathsf{upk}_i, \mathsf{msg}, \Sigma, \tau) = 0
\end{aligned}
$$

Otherwise return 0

We say that $\mathcal{GS}$ is correct if the advantage

$$\mathsf{Adv}^{corr}_{\mathcal{GS},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{corr}_{\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

Secondly, we define anonymity. Intuitively, anonymity ensures that an adversary who can corrupt the issuer and malicious users cannot extract any user information from a signature in the case when the adversary is able to access the opening oracle.

Definition 4.2.2 (Anonymity)**.** *Let $\mathcal{A}$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{anon}_{\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{anon}_{\mathcal{GS},\mathcal{A}}(\lambda):\quad & b \xleftarrow{\$} \{0,1\};\ (\mathsf{gpk},\mathsf{ik},\mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{CU} \leftarrow \emptyset;\ \mathsf{HU} \leftarrow \emptyset \\ & \widetilde{b} \leftarrow \mathcal{A}^{\mathsf{Corrupt}(\cdot,\cdot),\mathsf{SndToU}(\cdot),\mathsf{WReg}(\cdot,\cdot),\mathsf{USK}(\cdot),\mathsf{Open}(\cdot,\cdot),\mathsf{Ch}(b,\cdot,\cdot,\cdot)}(\mathsf{gpk},\mathsf{ik}) \\ & \text{Return 1 if } b = \widetilde{b},\ \text{otherwise return 0}\end{aligned}$$

We say that $\mathcal{GS}$ satisfies anonymity if the advantage

$$\mathsf{Adv}^{anon}_{\mathcal{GS},\mathcal{A}} = \left|\ \Pr[\ \mathsf{Exp}^{anon}_{\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ] - 1/2\ \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

Thirdly, we define traceability. Intuitively, this security notion ensures that an adversary who can corrupt the opener and malicious users cannot produce a valid group signature whose opening result is not valid (i.e., an invalid identity) or opening proof is not accepted by the judge algorithm. In the following, we give the formal definition of traceability.

Definition 4.2.3 (Traceability)**.** *Let $\mathcal{A}$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{trace}_{\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{trace}_{\mathcal{GS},\mathcal{A}}(\lambda):\quad & (\mathsf{gpk},\mathsf{ik},\mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{CU} \leftarrow \emptyset;\ \mathsf{HU} \leftarrow \emptyset \\ & (\mathsf{msg},\Sigma) \leftarrow \mathcal{A}^{\mathsf{Corrupt}(\cdot,\cdot),\mathsf{SndToI}(\cdot),\mathsf{AddU}(\cdot),\mathsf{RReg}(\cdot),\mathsf{USK}(\cdot)}(\mathsf{gpk},\mathsf{ok}) \\ & (i,\tau) \leftarrow \mathsf{GS.Open}(\mathsf{gpk},\mathsf{ok},\mathsf{msg},\Sigma,\mathsf{reg}) \\ & \text{Return 1 if the following two conditions hold :} \\ & \quad \mathsf{GS.Verify}(\mathsf{gpk},\mathsf{msg},\Sigma) = 1 \\ & \quad i = 0\ \vee\ \mathsf{GS.Judge}(\mathsf{gpk},i,\mathsf{upk}_i,\mathsf{msg},\Sigma,\tau) = 0\end{aligned}$$

Otherwise return 0

We say that $\mathcal{GS}$ satisfies traceability if the advantage

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

is negligible for any PPT adversary $\mathcal{A}$.

Next, we give the definition of non-frameability. Non-frameability ensures that an adversary who can corrupt the issuer, the opener, and malicious users except one honest user cannot produce a valid signature of this honest user and the corresponding opening proof, which is accepted by the judge algorithm.

Definition 4.2.4 (Non-Frameability)**.** *Let $\mathcal{A}$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{non\text{-}frame}_{\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$\mathsf{Exp}^{non\text{-}frame}_{\mathcal{GS},\mathcal{A}}(\lambda)$: $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{CU} \leftarrow \emptyset;\ \mathsf{HU} \leftarrow \emptyset$

$(\mathsf{msg}, \Sigma, j, \tau) \leftarrow \mathcal{A}^{\mathsf{SndToU}(\cdot),\mathsf{Sign}(\cdot,\cdot),\mathsf{WReg}(\cdot,\cdot),\mathsf{USK}(\cdot),\mathsf{Corrupt}(\cdot,\cdot)}(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$

Return 1 if all of the following hold :

- $j \in \mathsf{HU}$
- $\mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{msg}, \Sigma) = 1$
- $\mathsf{GS.Judge}(\mathsf{gpk}, j, \mathsf{upk}_j, m, \Sigma, \tau) = 1$
- $\mathcal{A}$ did not query $\mathsf{USK}(j)$ and $\mathsf{Sign}(j, \mathsf{msg})$

Otherwise return 0

We say that $\mathcal{GS}$ satisfies non-frameability if the advantage

$$\mathsf{Adv}^{non\text{-}frame}_{\mathcal{GS},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{non\text{-}frame}_{\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

Finally, we define opening soundness. Intuitively, opening soundness ensures that an adversary who can corrupt the issuer, the opener, and malicious users cannot produce a valid group signature and the corresponding opening proofs for i and j which are both accepted by the judge algorithm. Formal definition is as follows.

Definition 4.2.5 (Opening Soundness)**.** *Let $\mathcal{A}$ be an adversary for a group signature scheme $\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{open\text{-}sound}_{\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$\mathsf{Exp}^{open\text{-}sound}_{\mathcal{GS},\mathcal{A}}(\lambda)$: $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda)$

$(i, j, \tau_i, \tau_j, \mathsf{msg}, \Sigma) \leftarrow \mathcal{A}^{\mathsf{Corrupt}(\cdot),\mathsf{WReg}(\cdot,\cdot)}(\mathsf{gpk}, \mathsf{ok}, \mathsf{ik})$

Return 1 if all of the following hold :

$\mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{msg}, \Sigma) = 1$

$\mathsf{GS.Judge}(\mathsf{gpk}, i, \mathsf{upk}_i, \mathsf{msg}, \Sigma, \tau_i) = 1$

$i \neq j \wedge \mathsf{GS.Judge}(\mathsf{gpk}, j, \mathsf{upk}_j, \mathsf{msg}, \Sigma, \tau_j) = 1$

Otherwise return 0

We say that $\mathcal{GS}$ satisfies opening soundness if the advantage

$$\mathsf{Adv}^{open\text{-}sound}_{\mathcal{GS},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{open\text{-}sound}_{\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

4.3 The Modified Groth Scheme

In this section, we introduce the modified Groth scheme [100], which is one of efficient schemes with opening soundness. The original scheme is proposed by Groth [61]. Firstly, we give building blocks using in the modified Groth scheme in Section 4.3.1. Then, description of the modified Groth scheme is provided in Section 4.3.2.

4.3.1 Building Blocks

Here, we review the Kiltz tag-based encryption scheme [76] and the Groth certified signature scheme [61], which are used in the modified Groth scheme as building blocks.

The Kiltz Tag-based Encryption Scheme [76]. Let $\mathsf{gk} = (p, \mathbb{G}, \mathbb{G}_T, e, G)$ be a group description. The key generation algorithm $\mathsf{PKE.Gen}(1^\lambda)$ chooses random integers $\zeta, \eta \leftarrow \mathbb{Z}_p$ and random elements $K, L \leftarrow \mathbb{G}$, and sets a public key $\mathsf{ek} = (F, H, K, L)$ where $F = G^\zeta$ and $H = G^\eta$ and a decryption key $\mathsf{dk} = (\zeta, \eta)$. The encryption algorithm $\mathsf{PKE.Enc}(\mathsf{ek}, t, m)$ outputs $y = (y_1, y_2, y_3, y_4, y_5) = (F^r, H^s, mG^{r+s}, (G^tK)^r, (G^tL)^s)$ where m is a plaintext, t is a tag, and r, s are randomness. The validity of a ciphertext y for a tag t is publicly verifiable by checking the two equations $e(F, y_4) = e(y_1, G^tK)$ and $e(H, y_5) = e(y_2, G^tL)$. Let $\mathsf{ValidCiphertext}_{\mathsf{ek}}$ be an algorithm verifying the validity of a ciphertext indexed a public key ek. The decryption algorithm $\mathsf{PKE.Dec}(\mathsf{dk}, t, y)$ outputs $m = y_3/(y_1^{-\zeta}y_2^{-\eta})$ if the above two equations hold, otherwise outputs $\perp$.

The Kiltz tag-based encryption scheme is secure against selective-tag weak chosen ciphertext attack under the DLIN assumption. In the modified Groth scheme, the same F, H are used as in the common reference string of non-interactive proofs.

The Groth Certified Signature Scheme [61]. A certified signature scheme [23] is a combined scheme for signing messages and producing certificates of verification keys. In the following, we give a formal definition given by Groth [61]. The key generation algorithm CertKeyGen takes a group parameter gk as input and outputs a public authority key ak and a certification key ck. CertJoin/CertIss is the pair of (interactive) algorithms. CertJoin takes gk and ak as input whereas CertIss takes gk and ck as input. If successful CertJoin outputs a tuple $(\mathsf{vk}, \mathsf{sk}, \mathsf{cert})$ whereas CertIss outputs $(\mathsf{vk}, \mathsf{cert})$. The signing algorithm CertSign takes sk and a message m as input and outputs a signature σ. The verification algorithm CertVer takes $\mathsf{gk}, \mathsf{ak}, \mathsf{vk}, \mathsf{cert}, m$, and σ as input and outputs 1 or 0. We say that a certified signature scheme is secure if it satisfies unfakeability and existential unforgeability against weak chosen message attack. See the paper [61] for definitions of unfakeability and EUF-wCMA.

Groth [61] constructed an efficient certified signature scheme secure under the q-SDH assumption and q-U assumption, and this scheme is used in the modified Groth group signature scheme. For certification, there are two steps in the Groth certified signature scheme. In the first step, the protocol [61] generates a random $v = G^x$ such that the issuer learns v but only the user learns x. In the second step, a variant of the signature scheme of Zhou and Lin [106] is used to certify a verification key v. To set up the certified signature scheme, the authority picks random group elements $f, h, z \in \mathbb{G}$, and sets the authority key (f, h, T) and the secret certification key z where $T = e(f, z)$. When the authority certifies a key v, he picks random $r \in \mathbb{Z}_p$ and sets $(a, b) = (f^{-r}, (hv)^r z)$ as the certificate. The certificate is verified by checking $e(a, hv) \cdot e(f, b) = T$.

4.3.2 Description

We give a description of the modified Groth scheme in Figure 4.1. Before giving the intuition of the scheme, we note that the modified Groth scheme slightly diverge from the BSZ model described in Section 4.1. More precisely, in the BSZ model, it is assumed that a user generate a public and secret key pair $(\mathsf{upk}_i, \mathsf{usk}_i)$, and then afterwards obtain a signing key gsk_i by interacting with the issuer in the GS.Join/GS.Issue protocol. On the other hand, in the modified Groth scheme, a user generates a public/secret key pair jointly with the issuer in the GS.Join/GS.Issue protocol. This intuitively indicates that the user key generation algorithm GS.UGen is merged with the GS.Join/GS.Issue protocol in the modified Groth scheme.

Now, we give the intuition of the modified Groth scheme. The core of the modified Groth scheme is the certified signature scheme described in Section 4.3.1. In the GS.Join/GS.Issue protocol, for which users enroll the group, the issuer plays a role as a certification authority and a user i obtains a signing key for the Boneh-Boyen signa-

ture [24] x_i and a certificate $(a_i, b_i) = (f^{-r}, (v_i h)^r z)$, which certifies x_i's verification key $v_i = G^{x_i}$. When generating a group signature, a user generates a key pair of the one-time signature scheme $(\mathsf{vk}_{\mathrm{sots}}, \mathsf{sk}_{\mathrm{sots}})$. Then, the user generates a signature σ on the verification key $\mathsf{vk}_{\mathrm{sots}}$ by his signing key x_i, and signs a message m using $\mathsf{sk}_{\mathrm{sots}}$.

To make signatures be anonymous, the user encrypts the signature σ by the Kiltz tag-based encryption scheme and proves that the procedure is honestly done by the NIZK proof system, and he also proves that he knows the valid certificate by the non-interactive witness-indistinguishable (NIWI) proof system. More precisely, in the $\mathsf{GS.Sign}$ algorithm, a signer constructs two Groth-Sahai proofs [63]. The first proof π, constructed via $\mathsf{P}_{\mathrm{NIWI}}$ shows the knowledge of a signature σ, a verification key v, and a part b of a certificate (a, b) that satisfies

$$e(a, hv) \cdot e(f, b) = T \ \wedge \ e(\sigma, vG^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) = e(G, G)$$

where the first part a is revealed in the signature. The second proof ψ, constructed via $\mathsf{P}_{\mathrm{NIZK}}$ demonstrates that the plaintext of y is the same as the witness σ used in π. That is, for a commitment $c = (c_1, c_2, c_3) = (F^{r_c} U^t, H^{s_c} V^t, G^{r_c + s_c} W^t \sigma)$ contained in π, there exists (r, s, t) such that

$$(c_1 y_1^{-1}, c_2 y_2^{-1}, c_3 y_3^{-1}) = (F^r U^t, H^s V^t, G^{r+s} W^t).$$

When opening a signature, the opener convinces other parties that the opening procedure is honestly done. More precisely, the opener reveals G^r and G^s where $y = (y_1, y_2, y_3, y_4, y_5) = (F^r, H^s, mG^{r+s}, (G^t K)^r, (G^t L)^s)$ is the ciphertext in a signature. Then, a verifier checks the validity by checking the equations $e(F, \tau_F) = e(y_1, G)$, $e(H, \tau_H) = e(y_2, G)$, $\sigma \tau_F \tau_H = y_3$, and $e(\sigma, v_i G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) = e(G, G)$.

The modified Groth scheme uses a universal one-way hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p$, the Groth-Sahai proof systems $(\mathsf{K}_{\mathrm{NI}}, \mathsf{P}_{\mathrm{NIWI}}, \mathsf{V}_{\mathrm{NIWI}}, \mathsf{X}_{\mathrm{NIWI}})$ and $(\mathsf{K}_{\mathrm{NI}}, \mathsf{P}_{\mathrm{NIZK}}, \mathsf{V}_{\mathrm{NIZK}}, \mathsf{X}_{\mathrm{NIZK}})$, a strong one-time signature $(\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify})$, the Kiltz tag-based encryption scheme, and the Groth certified signature scheme as building blocks. Note that in the modified Groth scheme, we use a common K_{NI} for both systems and $\mathsf{X}_{\mathrm{NIZK}}$ is not used.

$\mathsf{GS.GGen}(1^\lambda)$:
$\mathsf{gk} = (p, \mathbb{G}, \mathbb{G}_T, e, G) \leftarrow \mathcal{G}(1^\lambda)$; $\mathcal{H} \leftarrow \mathrm{HashGen}(1^\lambda)$; $f, h, z \leftarrow \mathbb{G}$; $T \leftarrow e(f, z)$
$(\mathsf{crs}, \mathsf{xk}) \leftarrow \mathsf{K}_{\mathrm{NI}}(\mathsf{gk})$; $(F, H, U, V, W, U', V', W') \leftarrow \mathsf{crs}$; $K, L \leftarrow \mathbb{G}$; $\mathsf{ek} \leftarrow (F, H, K, L)$
$(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow ((\mathsf{gk}, \mathcal{H}, f, h, T, \mathsf{crs}, \mathsf{ek}), z, \mathsf{xk})$
Return $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$

$\mathsf{GS.Join}/\mathsf{GS.Issue}$(User i: gpk; Issuer: $\mathsf{gpk}, \mathsf{ik}$):
Run the coin-flipping protocol in [61]
The user obtains $v_i = g^{x_i}$ and x_i and the issuer obtains v_i
(Repeat until $v_i \neq \mathsf{reg}[j]$ for all j)
Issuer:
$r \leftarrow \mathbb{Z}_p$; $(a_i, b_i) \leftarrow (f^{-r}, (v_i h)^r z)$
Set $\mathsf{reg}[i] \leftarrow v_i$ and send (a_i, b_i) to the user
User:
Set $\mathsf{gsk}_i \leftarrow (x_i, a_i, b_i)$ if $e(a_i, h v_i) \cdot e(f, b_i) = T$

$\mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m)$:
$(\mathsf{vk}_{\mathrm{sots}}, \mathsf{sk}_{\mathrm{sots}}) \leftarrow \mathsf{KeyGen}_{\mathrm{sots}}(1^k)$ (Repeat until $\mathcal{H}(\mathsf{vk}_{\mathrm{sots}}) \neq -x_i$)
$\rho \leftarrow \mathbb{Z}_p$; $a \leftarrow a_i f^{-\rho}$; $b \leftarrow b_i (h v_i)^\rho$; $\sigma \leftarrow G^{1/(x_i + \mathcal{H}(\mathsf{vk}_{\mathrm{sots}}))}$
$\pi \leftarrow \mathsf{P}_{\mathrm{NIWI}}(\mathsf{crs}, (\mathsf{gpk}, a, \mathcal{H}(\mathsf{vk}_{\mathrm{sots}})), (b, v_i, \sigma))$; $y \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}, \mathcal{H}(\mathsf{vk}_{\mathrm{sots}}), \sigma)$
$\psi \leftarrow \mathsf{P}_{\mathrm{NIZK}}(\mathsf{crs}, (\mathsf{gpk}, y, \pi), (r, s, t))$; $\sigma_{\mathrm{sots}} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{\mathrm{sots}}, m, a, \pi, y, \psi)$
Return $\Sigma = (\mathsf{vk}_{\mathrm{sots}}, a, \pi, y, \psi, \sigma_{\mathrm{sots}})$

$\mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{reg}, m, \Sigma)$:
Return 1 if the following holds:
$\mathsf{SIG.Verify}(\mathsf{vk}_{\mathrm{sots}}, (\mathsf{vk}_{\mathrm{sots}}, m, a, \pi, y, \psi), \sigma_{\mathrm{sots}}) = 1$
$\mathsf{V}_{\mathrm{NIWI}}(\mathsf{crs}, (\mathsf{gpk}, a, \mathcal{H}(\mathsf{vk}_{\mathrm{sots}})), \pi) = 1$
$\mathsf{V}_{\mathrm{NIZK}}(\mathsf{crs}, (\mathsf{gpk}, y, \pi), \psi) = 1$
$\mathsf{ValidCiphertext}_{\mathsf{ek}}(\mathcal{H}(\mathsf{vk}_{\mathrm{sots}}), y) = 1$
$\mathsf{reg}[i] \neq \mathsf{reg}[j]$ for all $i \neq j$
else return 0

$\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \Sigma)$:
If $\mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{reg}, m, \Sigma) = 0$, return $(0, \perp)$
$(b, v, \sigma) \leftarrow \mathsf{X}_{\mathsf{xk}}(\mathsf{crs}, (\mathsf{gpk}, a, \mathcal{H}(\mathsf{vk}_{\mathrm{sots}})), \pi)$; $(d_F, d_H) \leftarrow \mathsf{xk}$
$(y_1, y_2, \ldots, y_5) \leftarrow y$; $\tau_F \leftarrow y_1^{1/d_F}, \tau_H \leftarrow y_2^{1/d_H}$
Return $(i, (\sigma, \tau_F, \tau_H))$ if there is i s.t. $v = \mathsf{reg}[i]$, else $(0, \perp)$

$\mathsf{GS.Judge}(\mathsf{gpk}, i, \mathsf{reg}, m, \Sigma, (\sigma, \tau_F, \tau_H))$:
$v_i \leftarrow \mathsf{reg}[i]$
Return 1 if the following hold:
$\mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{reg}, m, \Sigma) = 1$
$i \neq 0 \ \wedge \ e(\sigma, v_i G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) = e(G, G)$
$e(F, \tau_F) = e(y_1, G) \ \wedge \ e(H, \tau_H) = e(y_2, G)$
$\sigma \tau_F \tau_H = y_3$
else return 0

Figure 4.1: The Modifie Groth Scheme

Chapter 5

The Minimal Assumptions of Group Signature

It is important to investigate the minimum assumptions in order for a cryptographic primitive to exist. Specifically, it is essential to investigate that the cryptographic primitive can be constructed from a OWF since the existence of a OWF is the most basic and weakest assumption in cryptography. For example, a pseudorandom function (PRF), a digital signature scheme, a SKE scheme, and a commitment scheme can be constructed from a OWF [64, 58, 98, 84, 91]. On the other hand, a PKE scheme cannot be constructed from a OWF in a black-box manner even in the random oracle model [67].

In this chapter, we focus on group signature and investigate the minimum assumptions for the existence of it. Especially, it is pointed out that the minimal assumptions depend on whether the target group signature is fully anonymous or selfless anonymous.

5.1 The Minimal Assumptions of Fully Anonymous Group Signature

The previous works [6, 95, 48, 49] showed that a PKE scheme (and its extended schemes [44, 57, 26]) can be constructed from a group signature scheme that satisfies full anonymity.

Intuitively, in constructions of a PKE scheme from a group signature scheme, the indistinguishability of the PKE scheme comes from the anonymity of the underlying group signature scheme. Concretely, the underlying group signature scheme is set up for two users i_0 and i_1 in the key generation of a PKE scheme, and a signature for the user i_b is considered as an encryption of a message $b \in \{0, 1\}$. From the anonymity of the group signature scheme, the user ID i_b (the message b) cannot be extracted from the signature (the ciphertext). However, by using the opening key (the decryption key), the

user ID i_b (the message b) can be recovered. Here, the users' signing keys correspond to the encryption key of the PKE scheme. Therefore, for achieving the indistinguishability of the PKE scheme, the anonymity of the group signature scheme should be guaranteed even if the users' signing keys are published. Thus, the underlying group signature scheme need to be fully anonymous to construct a PKE scheme in the construction.

Formally, the following theorems hold. Recall that the BMW model and the BSZ model were appeared in Chapter 3 and 4, respectively. Also, public key encryption with non-interactive opening [44, 57] and threshold public key encryption [26] are strictly stronger primitives than public key encryption.

Theorem 5.1.1 ([6])**.** *If fully anonymous group signature (secure in the sense of the BMW model [14]) exists, public key encryption which is IND-CPA secure exists.*

Theorem 5.1.2 ([95])**.** *If fully anonymous group signature (secure in the sense of the BMW model [14]) exists, public key encryption which is IND-CCA secure exists.*

Theorem 5.1.3 ([48, 49])**.** *If fully anonymous group signature (secure in the sense of the BSZ model [17]) exists, public key encryption with non-interactive opening exists.*

Theorem 5.1.4 ([48])**.** *If fully anonymous group signature (secure in the sense of the BSZ model [17]) exists, threshold public key encryption exists.*

From these theorems, it is unlikely to construct a fully anonymous group signature scheme only from a OWF. This is because, if a group signature scheme can be constructed from a OWF, this also derive the fact that a PKE scheme can be constructed from a OWF. However, this contradicts to the impossibility result by Impagliazzo and Rudich [67].

On the other hand, there is a possibility that a group signature scheme which satisfies selfless anonymity [28] can be constructed from a OWF since a conversion from a selfless anonymous group signature scheme to a PKE scheme is not known. In the next section, we discuss the minimal assumptions of selfless anonymous group signature.

5.2 A Construction of a Selfless Anonymous Group Signature Scheme without a Public Key Encryption Scheme

In this section, we give a construction of a selfless anonymous group signature scheme without any PKE scheme. Concretely, we construct a selfless anonymous group signature scheme from a SKE scheme, a commitment scheme, a digital signature scheme, and a NIZK proof system. Here, we require that the underlying SKE scheme satisfies

the following properties: ciphertext-pseudorandomness and key-robustness. Intuitively, ciphertext-pseudorandomness ensures that it is hard to distinguish a ciphertext from a random value. In addition, key-robustness ensures that for any two random keys it is hard to find a ciphertext that is not decrypted to $\perp$ under both keys.

Moreover, we give a construction of the underlying SKE scheme from a PRF. This means that a SKE scheme with ciphertext-pseudorandomness and key-robustness can be constructed from a OWF since a PRF can be constructed from a OWF.

Camenisch and Groth [34] showed a construction of selfless anonymous group signature schemes. Although our construction is different from that of Camenisch and Groth, these claimed results are same (that is, a selfless anonymous group signature scheme can be constructed from a OWF and a NIZK proof system). We here give full proofs of our scheme while Camenisch and Groth provided only proof sketches.

5.2.1 High Level Idea

In this section, we firstly consider the construction in which a SKE scheme is simply used instead of a PKE scheme for the BMW construction, and point out some issues of this simple construction. Then, we give a high level idea to overcome these issues.

The Problems in the Simple SKE-based BMW Construction. Here, we simply replace a PKE scheme with a SKE scheme in the BMW construction. In this construction, the encryption key used in signing and the opening key are the secret key of the SKE scheme. Since the encryption key is necessary to generate a ciphertext C that is a part of a signature, a signer needs to possess the SKE secret key as a part of the signing key.[1] This means that an adversary who has the signing key used in generating a group signature $\sigma = (C, \pi)$ can decrypt C and easily identify the signer. Therefore, the group signature in the SKE-based construction does not satisfy full anonymity. On the other hand, one might think that this modified group signature scheme satisfies selfless anonymity. However in fact, the scheme does not satisfy selfless anonymity. In the following, we point out three problems of this construction.

The first problem is that there is a possibility of leaking the signer identity from the ciphertext C. Since the SKE secret key used in signing is different by users in the SKE-based construction, this problem might occur. Strictly speaking, it does not happen that the SKE secret key is leaked from a ciphertext since it contradicts the CPA security of the SKE scheme. However, it may occur that for two ciphertexts an adversary can distinguish

[1] If all the users use the same SKE secret key k, the group signature does not satisfy anonymity since they can open group signatures generated by other users using the key k. Therefore, we need to provide n secret keys $k_1, \ldots, k_n$ of the SKE scheme where n is the number of users, and each user i possesses a key k_i.

whether they are encrypted under the same key. If the adversary can distinguish it, he also can break the anonymity of the group signature scheme by comparing the challenge signature and the signature that is given from the signing oracle. Therefore, we cannot prevent such attacks only by requiring the CPA security for the underlying SKE scheme.

Second, there is a possibility that the opener cannot open signatures correctly. Let k_i be the SKE secret key for the user i. The opener has n secret keys $k_1, \ldots, k_n$ as the opening key. For opening a group signature $\sigma = (C, \pi)$, the opener tries to decrypt C by all the secret keys $k_1, \ldots, k_n$ and outputs the decryption result under the key which does not induce $\perp$ since he does not know which secret key is used in generating a ciphertext C. However, if there exist more than two such keys, the opener cannot decide which decryption result is the real one. That is, he cannot determine the opening result uniquely.

Lastly, there exist attacks that break the traceability. We show the way to construct a forgery in the following. First, the user i in the group generates a signature s under his signing key sk_i. Second, he computes a ciphertext C^* that is an encryption of the message $\langle i, \mathsf{vk}_i, \mathsf{cert}_i, s\rangle$ by using the key $\widehat{k} \notin \{k_1, \ldots, k_n\}$ generated by himself, and makes a proof π^* that s and C^* are honestly generated. More precisely, the user generates a proof π^* that proves the equations (a) $\mathsf{SKE.Enc}(\widehat{k}, \langle i, \mathsf{vk}_i, \mathsf{cert}_i, s\rangle; R) = C^*$, (b) $\mathsf{SIG.Verify}(\mathsf{vk}, (\langle i, \mathsf{vk}_i\rangle, \mathsf{cert}_i)) = 1$, and (c) $\mathsf{SIG.Verify}(\mathsf{vk}_i, (m, s)) = 1$ for the statement $x = \langle \mathsf{vk}, m, C\rangle$ and the witness $w = \langle i, \mathsf{vk}_i, \mathsf{cert}_i, s, R, \widehat{k}\rangle$. Then, he outputs the group signature $\sigma^* = (C^*, \pi^*)$ as a forgery. The group signature is valid since the proof π^* is accepted by the verification algorithm. However, it is untraceable since the opener does not have the key $\widehat{k}$ and cannot extract the signer's identity. Therefore, the group signature is a forgery that breaks the traceability.

From these problems, the simple construction in which we replace a PKE scheme with the SKE scheme for the BMW construction does not achieve selfless anonymity.

The High Level Idea to Overcome the Problems. To settle the first problem, we additionally require the underlying SKE scheme to be key-anonymous. Intuitively, key-anonymity ensures that for two ciphertexts it is difficult to distinguish whether they are encrypted under the same key. By this new security requirement, we can prevent attacks to compare the challenge signature and the signature that is given from the signing oracle.

We solve the second problem by additionally requiring key-robustness for the underlying SKE scheme. Intuitively, key-robustness ensures that for any two random keys it is hard to find a ciphertext that is not decrypted to $\perp$ under both keys. If the SKE scheme satisfies key-robustness and the ciphertext C is encrypted under one of the keys $k_1, \ldots, k_n$, there exists only one index $i \in [1, n]$ such that $\mathsf{SKE.Dec}(k_i, C) \neq \perp$ with high probability. Therefore by introducing key-robustness to the SKE scheme, the opener can

determine the opening result uniquely with high probability.

For the third problem, we add a commitment scheme as a building block for ensuring that the SKE key k_i used in generating group signatures is issued by the group manager. Then in our construction, the group manager computes the commitment c_i of the key k_i for $i \in [1, n]$, and publishes the commitments $c_1, \ldots, c_n$ as a part of the group public key. Each user i possesses the randomness r_i used in committing k_i as a part of his signing key. Then, he additionally proves that at least one commitment $c^* \in \{c_1, \ldots, c_n\}$ contains the key k_i by the NIZK proof system. By doing so, if the uncommitted key $\widehat{k}$ is used to generate a group signature, it will be rejected by the verification algorithm.

5.2.2 Description

In this section, we describe the construction of a selfless anonymous group signature scheme. At first, we explain the construction methodology based on the idea presented in the previous section. After that, we slightly change the construction for simplicity. We explain this change in the paragraph "The Simplification of the Construction" in this section. In the construction, we use a SKE scheme $\mathcal{SKE} = (\mathsf{SKE.Gen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$, a commitment scheme $\mathcal{COM} = (\mathsf{COM.Setup}, \mathsf{Commit})$, a signature scheme $\mathcal{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify})$, and a NIZK proof system $\mathcal{P}_L = (\mathsf{ZK.Gen}, \mathsf{ZK.Prove}, \mathsf{ZK.Verify})$ as building blocks. We mention the language L for which the proof system proves the membership in the last of this section.

For the BMW construction, we replace a PKE scheme with a SKE scheme, and additionally require key-anonymity and key-robustness to the SKE scheme. Moreover, to certify the SKE secret keys we introduce a commitment scheme as an additional building block. Specifically, the group manager computes $c_i \leftarrow \mathsf{Commit}_{\mathsf{ck}}(k_i; r_i)$ for each key k_i and publishes the list $T = \{c_1, \ldots, c_n\}$ in a group public key gpk. Each user i has the randomness r_i as a part of his signing key. When the user i generates a group signature, for the statement $x = \langle \mathsf{vk}, m, C, T, \mathsf{ck} \rangle$ and the witness $w = \langle i, \mathsf{vk}_i, \mathsf{cert}_i, s, R, k_i, r_i \rangle$ he proves the equation

$$(*)\ \mathsf{Commit}_{\mathsf{ck}}(k_i; r_i) = c_1 \vee \ldots \vee \mathsf{Commit}_{\mathsf{ck}}(k_i; r_i) = c_n$$

in addition to the equations (a) $\mathsf{SKE.Enc}(k_i, \langle i, \mathsf{vk}_i, \mathsf{cert}_i, s \rangle; R) = C$, (b) $\mathsf{SIG.Verify}(\mathsf{vk}, (\langle i, \mathsf{vk}_i \rangle, \mathsf{cert}_i)) = 1$, and (c) $\mathsf{SIG.Verify}(\mathsf{vk}_i, (m, s)) = 1$. By doing this, it can be ensured that the key k_i used to generate C is certified by the group manager. Intuitively, if an adversary generates a group signature by using an uncommitted key $\widehat{k} \notin \{k_1, \ldots, k_n\}$ and a randomness $\widehat{r}$, and the group signature is valid, there exists at least one index $i \in [1, n]$ such that $\mathsf{Commit}_{\mathsf{ck}}(\widehat{k}; \widehat{r}) = c_i$ from the equation $(*)$. However, the situation happens with negligible probability by the binding property of the underlying commitment scheme.

Also, the information of the key k_i is not leaked from the commitment c_i by the hiding property.[2]

The Simplification of the Construction. We give further two modifications for the construction explained above.

First, for simplicity of proofs, we require ciphertext-pseudorandomness instead of CPA security and key-anonymity to be satisfied by the SKE scheme. Intuitively, ciphertext-pseudorandomness ensures that an adversary who does not possess a secret key cannot distinguish a ciphertext generated by the encryption algorithm and a random value in the ciphertext space. Thus, ciphertext-pseudorandomness implies CPA security and key-anonymity.

Second, a pair of the user ID and his verification key $\langle i, \mathsf{vk}_i \rangle$ is committed with his SKE key k_i, and then the group manager omits to issue the certificate $\mathsf{cert}_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, \langle i, \mathsf{vk}_i \rangle)$. More precisely for each user i, the group manager computes a commitment $c_i \leftarrow \mathsf{Commit}_{\mathsf{ck}}(\langle i, \mathsf{vk}_i, k_i \rangle; r_i)$, and publishes the list $T = \{c_1, \ldots, c_n\}$ where r_i is a randomness. Each user i possesses the randomness r_i as a part of the signing key gsk_i. When the user i generates a group signature, he proves the equations

$(\tilde{a})$ $\mathsf{SKE.Enc}(k_i, \langle i, \mathsf{vk}_i, s \rangle; R) = C$,

$(\tilde{b})$ $\mathsf{SIG.Verify}(\mathsf{vk}_i, (m, s)) = 1$

$(\tilde{c})$ $\mathsf{Commit}_{\mathsf{ck}}(\langle i, \mathsf{vk}_i, k_i \rangle; r_i) = c_1 \vee \ldots \vee \mathsf{Commit}_{\mathsf{ck}}(\langle i, \mathsf{vk}_i, k_i \rangle; r_i) = c_n$

for the statement $x = \langle m, C, T, \mathsf{ck} \rangle$ and the witness $w = \langle i, \mathsf{vk}_i, s, R, k_i, r_i \rangle$. In the equation $(\tilde{c})$, it can be ensured that the verification key vk_i is certified by the group manager by the binding of the commitment scheme.

We give our selfless anonymous group signature scheme in Figure 5.1. In our construction, when for a statement $x = \langle m, C, T, \mathsf{ck} \rangle$ and a witness $w = \langle i, \mathsf{vk}_i, s, R, k_i, r_i \rangle$, the equations $(\tilde{a})$, $(\tilde{b})$, and $(\tilde{c})$ hold, we say that x and w satisfy the relation R_L. Moreover for a statement $x = \langle m, C, T, \mathsf{ck} \rangle$, if there exists a witness $\langle i, \mathsf{vk}_i, s, R, k_i, r_i \rangle$ which satisfies the equations, we say that the statement belongs to the language L and denote $x \in L$.

The Correctness of the Scheme. We prove the correctness of the scheme in the following theorem.

[2]Some readers may think that it is natural to certify the SKE key k_i by generating a signature of k_i under the group manager's signing key sk as with the user's verification key vk_i. However, if do so, the signature of k_i will be encrypted under k_i in signing procedure for group signatures. That is, the circularity happens, and we need to require an additional security of the SKE scheme. Since it is not clear that such a SKE scheme can be constructed from a OWF, we use a commitment scheme.

$\mathsf{GS.Gen}(1^\lambda, 1^n)$:
  $\mathsf{crs} \leftarrow \mathsf{ZK.Gen}(1^\lambda)$; $\mathsf{ck} \leftarrow \mathsf{COM.Setup}(1^\lambda)$
  For $1 \le i \le n$:
    $k_i \leftarrow \mathsf{SKE.Gen}(1^\lambda)$; $(\mathsf{vk}_i, \mathsf{sk}_i) \leftarrow \mathsf{SIG.Verify}(1^\lambda)$
    $r_i \xleftarrow{\$} \{0,1\}^\lambda$; $c_i \leftarrow \mathsf{Commit}_{\mathsf{ck}}(\langle i, \mathsf{vk}_i, k_i\rangle; r_i)$
  $T \leftarrow \{c_1, \ldots, c_n\}$; $\mathsf{gpk} \leftarrow (\mathsf{crs}, \mathsf{ck}, T)$; $\mathsf{ok} \leftarrow (k_1, \ldots, k_n)$
  $\mathsf{gsk}_i \leftarrow (i, \mathsf{vk}_i, \mathsf{sk}_i, k_i, r_i)$; $\mathbf{gsk} \leftarrow \{\mathsf{gsk}_1, \ldots, \mathsf{gsk}_n\}$
 Return $(\mathsf{gpk}, \mathsf{ok}, \mathbf{gsk})$

$\mathsf{GS.Sign}(\mathsf{gsk}_i, m)$:
  $(i, \mathsf{vk}_i, \mathsf{sk}_i, k_i, r_i) \leftarrow \mathsf{gsk}_i$; $s \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_i, m)$
  $R \xleftarrow{\$} \mathcal{R}$; $C \leftarrow \mathsf{SKE.Enc}(k_i, \langle i, \mathsf{vk}_i, s\rangle; R)$
  $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, \langle m, C, T, \mathsf{ck}\rangle, \langle i, \mathsf{vk}_i, s, R, k_i, r_i\rangle)$; $\sigma \leftarrow (C, \pi)$
 Return σ

$\mathsf{GS.Verify}(\mathsf{gpk}, (m, \sigma))$:
  $(\mathsf{crs}, \mathsf{ck}, T) \leftarrow \mathsf{gpk}$; $(C, \pi) \leftarrow \sigma$
 Return $\mathsf{ZK.Verify}(\mathsf{crs}, \langle m, C, T, \mathsf{ck}\rangle, \pi)$

$\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, (m, \sigma))$:
  $(\mathsf{crs}, \mathsf{ck}, T) \leftarrow \mathsf{gpk}$; $(k_1, \ldots, k_n) \leftarrow \mathsf{ok}$; $(C, \pi) \leftarrow \sigma$
  If $\mathsf{GS.Verify}(\mathsf{gpk}, (m, \sigma)) = 0$, return $\emptyset$
  If $\forall i \in [1, n]$, $\mathsf{SKE.Dec}(k_i, C) = \perp$ then return $\perp$
  If $\exists i \neq j$, $\mathsf{SKE.Dec}(k_i, C) \neq \perp \wedge \mathsf{SKE.Dec}(k_j, C) \neq \perp$ then return $\perp$
  $\langle i, \mathsf{vk}_i, s\rangle \leftarrow \mathsf{SKE.Dec}(k_i, C)$
  If $i \notin [1, n]$, return $\perp$
 Otherwise return i

Figure 5.1: A Construction of Selfless Anonymous Group Signature Scheme without a Public Key Encryption Scheme

$\mathcal{SKE} = (\mathsf{SKE.Gen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ is a SKE scheme, $\mathcal{COM} = (\mathsf{COM.Setup}, \mathsf{Commit})$ is a commitment scheme, $\mathcal{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify})$ is a signature scheme, and $\mathcal{P}_L = (\mathsf{ZK.Gen}, \mathsf{ZK.Prove}, \mathsf{ZK.Verify})$ is a NIZK proof system. The language L is a collection of statements for which there exists a witness that satisfies the equations $(\tilde{a})$, $(\tilde{b})$, and $(\tilde{c})$ defined in Section 5.2.2.

Theorem 5.2.1. *The group signature scheme* Π_{GS} *is correct if the underlying SKE scheme satisfies key-robustness and correctness, the underlying signature scheme satisfies correctness, and the underlying NIZK proof system satisfies completeness.*

Proof. We prove that for all the user IDs i and all the messages $m \in \mathcal{M}$, the probability

$$\Pr[\ \mathsf{GS.Verify}(\mathsf{gpk}, (m, \sigma)) = 0 \vee i \neq i' \mid \begin{array}{l} (\mathsf{gpk}, \mathsf{ok}, \mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda, 1^n); \\ \sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_i, m); \\ i' \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, (m, \sigma))\ ] \end{array} \tag{5.1}$$

is negligible. When we omit the conditional part of the above probability, it holds that

$$\Pr[\ \mathsf{GS.Verify}(\mathsf{gpk},(m,\sigma))=0 \vee i \neq i'\ ] \leq \Pr[\ \mathsf{GS.Verify}(\mathsf{gpk},(m,\sigma))=0\ ] + \Pr[\ i \neq i'\ ].$$

From the correctness of the underlying SKE scheme, the correctness of the underlying signature scheme, and the completeness of the underlying NIZK proof system, it holds that $\Pr[\mathsf{GS.Verify}(\mathsf{gpk},(m,\sigma))=1]=1$. Therefore, $\Pr[\mathsf{GS.Verify}(\mathsf{gpk},(m,\sigma))=0] = 1-\Pr[\mathsf{GS.Verify}(\mathsf{gpk},(m,\sigma))=1]=0$. Then, we prove that $\Pr[i \neq i']$ is negligible. The probability $\Pr[i \neq i']$ can be converted and we explain the conversion in the following.

$$\begin{aligned}
\Pr[\ i \neq i'\ ] &\overset{\text{(A)}}{=} \Pr[\ \exists j \neq i,\ \mathsf{SKE.Dec}(k_j, C) \neq \bot \\
&\qquad\qquad \mid\ (\mathsf{gpk},\mathsf{ok},\mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda,1^n);\ \sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_i,m)\ ] \\
&\overset{\text{(B)}}{\leq} \sum_{j\neq i} \Pr[\ \mathsf{SKE.Dec}(k_j,C)\neq\bot \\
&\qquad\qquad\qquad \mid\ (\mathsf{gpk},\mathsf{ok},\mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda,1^n);\ \sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_i,m)\ ] \\
&\overset{\text{(C)}}{\leq} n \cdot \Pr[\ \mathsf{SKE.Dec}(k_1,C)\neq\bot \\
&\qquad\qquad\qquad \mid\ (\mathsf{gpk},\mathsf{ok},\mathbf{gsk}) \leftarrow \mathsf{GS.Gen}(1^\lambda,1^n);\ \sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gsk}_i,m)\ ] \\
&\overset{\text{(D)}}{\leq} n \cdot \mathsf{Adv}^{\textit{key-robust}}_{\mathcal{SKE},\mathcal{A}}(\lambda)
\end{aligned}$$

Here, k_i (resp. k_j) is a secret key of the user i (resp. j) which is generated in the key generation, and C is the ciphertext included in the signature σ. When a group signature $\sigma=(C,\pi)$ is honestly produced under the user i's signing key $\mathsf{gsk}_i=(i, \mathsf{vk}_i, \mathsf{sk}_i, k_i, r_i)$, it holds that $\mathsf{GS.Verify}(\mathsf{gpk},(m,\sigma))=1$, that is, the $\mathsf{GS.Open}$ algorithm never outputs $\emptyset$. Since the ciphertext C is decrypted to the message $\langle i, \mathsf{vk}_i, s\rangle$ under k_i, it holds that $\mathsf{SKE.Dec}(k_i,C)\neq\bot$. Therefore if it holds that $\mathsf{SKE.Dec}(k_j,C)=\bot$ for all the other indices $j\in[1,n]$, the $\mathsf{GS.Open}$ algorithm outputs a user ID i (i.e., $\mathsf{GS.Open}(\mathsf{gpk},\mathsf{ok},(m,\sigma))=i$). In the case that the opening result i' is different from the value i, it holds that $\mathsf{SKE.Dec}(k_j,C)\neq\bot$ for some index $j\in[1,n]$ which is different from i. Therefore, we can get the equality (A). Also, we obtain the inequality (B) by the union bound and the inequality (C) from the fact that the probability $\Pr[\mathsf{SKE.Dec}(k_j,C)\neq\bot]$ is symmetrical with respect to the index j. Furthermore, we obtain the inequality (D) by constructing the following adversary $\mathcal{A}$ for the key-robustness of the underlying SKE scheme.

$$\begin{aligned}
\mathcal{A}(k,\widetilde{k}):\ &\mathsf{ck} \leftarrow \mathsf{COM.Setup}(1^\lambda);\ (\mathsf{vk}_i,\mathsf{sk}_i) \leftarrow \mathsf{SIG.Gen}(1^\lambda);\ k_1 \leftarrow \widetilde{k} \\
&r_i \xleftarrow{\$} \{0,1\}^\lambda;\ c_i \leftarrow \mathsf{Commit}_{\mathsf{ck}}(\langle i,\mathsf{vk}_i,k\rangle; r_i);\ s \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_i,m) \\
&\text{Return } C \leftarrow \mathsf{SKE.Enc}(k,\langle i,\mathsf{vk}_i,s\rangle)
\end{aligned}$$

In the above, the distribution of C is the same as that in Π_{GS}. If $\mathsf{SKE.Dec}(k_1, C) \neq \perp$, $\mathcal{A}$ breaks the key-robustness for the SKE scheme, and then we obtain the inequality (D). Since the underlying SKE scheme is key-robust and n is polynomial in λ, the probability $n \cdot \mathsf{Adv}^{\mathit{key\text{-}robust}}_{\mathcal{SKE},\mathcal{A}}(\lambda)$ is negligible. Thus, since the probability (5.1) is negligible, our scheme Π_{GS} satisfies correctness. □

5.2.3 Security Proof

In this section, we prove that the scheme given in Section 5.2.2 satisfies selfless anonymity and traceability. First, we give a proof of the selfless anonymity.

Theorem 5.2.2. *The scheme Π_{GS} is selfless anonymous if the underlying NIZK proof system is zero-knowledge, the underlying commitment scheme is computational hiding, and the underlying SKE scheme is ciphertext-pseudorandom.*

Proof. Let $\mathcal{A}$ be an adversary that attacks selfless anonymity of Π_{GS}. We consider the following sequence of games. Let Suc_i denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit in Game i.

[Game 0]: This is the experiment $\mathsf{Exp}^{\mathit{anon}}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda, n)$ itself. For the sake of convenience, we choose a challenge bit $b \in \{0, 1\}$ at the beginning of the game. This change does not have an effect on the behavior of the adversary $\mathcal{A}$.

[Game 1]: Same as Game 0, except that a common reference string and proofs for the NIZK proof system are produced by using the simulator $\mathcal{S} = (\mathsf{S}_1, \mathsf{S}_2)$. Specifically, a common reference string crs is generated using the algorithm S_1 instead of the algorithm $\mathbb{G}$, and proofs π are generated using the algorithm S_2 instead of the algorithm P. The proof π^* in the challenge signature $\sigma^* = (C^*, \pi^*)$ is also generated by the algorithm S_2.

[Game 2]: In this game, we change the message committed in the commitment c_{i_b} where b is a challenge bit. More precisely, the message $0^{|\langle i_b, \mathsf{vk}_{i_b}, k_{i_b}\rangle|}$ is committed to c_{i_b} instead of $\langle i_b, \mathsf{vk}_{i_b}, k_{i_b}\rangle$. That is, $c_{i_b} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_b, \mathsf{vk}_{i_b}, k_{i_b}\rangle|}; r_{i_b})$ where r_{i_b} is a randomness.

[Game 3]: In Game 3, the message $0^{|\langle i_{1-b}, \mathsf{vk}_{i_{1-b}}, k_{i_{1-b}}\rangle|}$ is committed to the commitment $c_{i_{1-b}}$ instead of the message $\langle i_{1-b}, \mathsf{vk}_{i_{1-b}}, k_{i_{1-b}}\rangle$. That is, $c_{i_{1-b}} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_{1-b}, \mathsf{vk}_{i_{1-b}}, k_{i_{1-b}}\rangle|}; r_{i_{1-b}})$ where $r_{i_{1-b}}$ is a randomness.

[Game 4]: Throughout this game, a ciphertext C is chosen uniformly at random from the ciphertext space $\mathcal{C}$ when group signatures $\sigma = (C, \pi)$ for the user i_b are generated. This means that the ciphertext C^* in the challenge signature is also an uniformly random value.

[Game 5]: In Game 5, a ciphertext C is chosen uniformly at random from the space $\mathcal{C}$ when generating group signatures $\sigma = (C, \pi)$ for the user i_{1-b}.

For $\mathcal{A}$'s advantage $\mathsf{Adv}^{anon}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda)$,

$$\begin{aligned}\mathsf{Adv}^{anon}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda) =& \left|\Pr[\mathtt{Suc}_0] - 1/2\right| \\ &\leq \sum_{i=0}^{4}\left|\Pr[\mathtt{Suc}_i] - \Pr[\mathtt{Suc}_{i+1}]\right| + \left|\Pr[\mathtt{Suc}_5] - 1/2\right|\end{aligned}$$

holds. Moreover, the following lemmas hold.

Lemma 5.2.1. *There exists a PPT algorithm $\mathcal{B}_1$ such that $\left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right| = \mathsf{Adv}^{zk}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda)$.*

Proof. Let $\mathcal{B}_1$ be an adversary that attacks the zero-knowledge of the NIZK proof system. At the beginning of the game with $\mathcal{A}$, the algorithm $\mathcal{B}_1$ chooses a bit β uniformly at random. Then, $\mathcal{B}_1$ generates an instance of the selfless anonymity game except for a common reference string, and uses the common reference string crs which is given by the challenger of the zero-knowledge game. When $\mathcal{A}$ queries a message m to the Sign_B ($B \in \{0,1\}$) oracle, $\mathcal{B}_1$ operates by using the key $\mathsf{gsk}_{i_B} = (i_B, \mathsf{vk}_{i_B}, \mathsf{sk}_{i_B}, k_{i_B}, r_{i_B})$ he possesses as follows.

1. Generate a signature $s \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_B}, m)$.
2. Sample a randomness R and compute $C \leftarrow \mathsf{SKE.Enc}(k_{i_B}, \langle i_B, \mathsf{vk}_{i_B}, s\rangle; R)$.
3. Query (x, w) to the oracle of the zero-knowledge game and obtain a proof π where $x = \langle m, C, T, \mathsf{ck}\rangle$ and $w = \langle i_B, \mathsf{vk}_{i_B}, s, R, k_{i_B}, r_{i_B}\rangle$.
4. Return $\sigma = (C, \pi)$ to the adversary $\mathcal{A}$.

When getting a challenge query, $\mathcal{B}_1$ answers the query as with signing queries by using his proof oracle of the zero-knowledge game. Finally, when $\mathcal{A}$ terminates with a bit β', the algorithm $\mathcal{B}_1$ outputs $b' = 1$ if $\beta' = \beta$ otherwise outputs $b' = 0$.

If the common reference string that $\mathcal{B}_1$ is given by the challenger is generated by the algorithm $\mathsf{ZK.Gen}$, and the oracle which $\mathcal{B}_1$ accesses is the oracle $\mathsf{ZK.Prove}(\mathsf{crs}, \cdot, \cdot)$, $\mathcal{B}_1$ perfectly simulates Game 0 for $\mathcal{A}$. On the other hand, if the common reference string given by the challenger is generated by the simulator $\mathcal{S}$, and the oracle which $\mathcal{B}_1$ accesses is the oracle constructed from $\mathcal{S}$, $\mathcal{B}_1$ perfectly simulates Game 1 for $\mathcal{A}$. Therefore, it holds that $\mathsf{Adv}^{zk}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) = \left|\Pr[\mathsf{Exp}^{proof}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) = 1] - \Pr[\mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) = 1]\right| = \left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right|$. □

Lemma 5.2.2. *There exists a PPT algorithm $\mathcal{B}_2$ such that $\left|\Pr[\mathtt{Suc}_1] - \Pr[\mathtt{Suc}_2]\right| = 2 \cdot \mathsf{Adv}^{hiding}_{\mathcal{COM},\mathcal{B}_2}(\lambda)$.*

Proof. Let $\mathcal{B}_2$ be an adversary that attacks the hiding of the commitment scheme. At first, $\mathcal{B}_2$ chooses a challenge bit $\beta \in \{0,1\}$ uniformly at random. $\mathcal{B}_2$ is given a commitment key ck from the challenger, and two user IDs i_0 and i_1 from the adversary $\mathcal{A}$. Also, $\mathcal{B}_2$ generates a common reference string crs' by using the algorithm S_1. For the user i except for i_β, $\mathcal{B}_2$ generates signature keys gsk_i and commitments c_i by following algorithms of Π_{GS}. In terms of the user i_β, a key pair $(\mathsf{vk}_{i_\beta}, \mathsf{sk}_{i_\beta})$ is generated following the scheme Π_{GS}, and a commitment c_{i_β} is produced as follows.

1. Generate $k_{i_\beta} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$.
2. Send $(0^{|\langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle|}, \langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle)$ to the challenger as a challenge and obtain a commitment c^*.
3. Set $c_{i_\beta} \leftarrow c^*$.

We note that $\mathcal{B}_2$ cannot obtain the i_β's whole signing key gsk_{i_β} since he does not know the randomness used in generating the commitment c^*. However, since the user i_β is one of challenge users, $\mathcal{B}_2$ does not have to give the key gsk_{i_β} to $\mathcal{A}$. Moreover for signing queries of i_β from $\mathcal{A}$, the algorithm $\mathcal{B}_2$ can produce a signature $\sigma = (C, \pi)$ without the randomness by using the trapdoor td obtained in generating crs'. Specifically, when $\mathcal{A}$ queries a message m to the Sign_B $(B \in \{0,1\})$ oracle, $\mathcal{B}_2$ answers as follows.

1. Generate a signature $s \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_B}, m)$.
2. Sample a randomness R and compute $C \leftarrow \mathsf{SKE.Enc}(k_{i_B}, \langle i_B, \mathsf{vk}_{i_B}, s\rangle; R)$.
3. Compute a proof $\pi \leftarrow \mathsf{S}_2(\mathsf{crs}', \mathsf{td}, x)$ where $x = \langle m, C, T, \mathsf{ck}\rangle$.
4. Return $\sigma = (C, \pi)$ to the adversary $\mathcal{A}$.

$\mathcal{B}_2$ can generate a challenge signature in the same way. For a conclusive output β' of $\mathcal{A}$, $\mathcal{B}_2$ returns $b' = 1$ if β' is same as the bit β that he chose at the beginning of the game, and $b' = 0$ otherwise.

In this situation, if $b = 0$, the commitment c_{i_β} for the challenge user i_β is $c_{i_\beta} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle|}; r_{i_\beta})$. That is, $\mathcal{B}_2$ perfectly simulates Game 2 for $\mathcal{A}$. On the other hand, if $b = 1$, the commitment c_{i_β} is $c_{i_\beta} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(\langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle; r_{i_\beta})$, and then $\mathcal{B}_2$ perfectly simulates Game 1 for $\mathcal{A}$. Therefore, $\mathsf{Adv}^{hiding}_{\mathcal{COM},\mathcal{B}_2}(\lambda) = \left|\Pr[b = b'] - 1/2\right| = 1/2 \cdot \left|\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]\right| = 1/2 \cdot \left|\Pr[\mathtt{Suc}_1] - \Pr[\mathtt{Suc}_2]\right|$ holds. □

Lemma 5.2.3. *There exists a PPT algorithm $\mathcal{B}_3$ such that* $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| = 2 \cdot \mathsf{Adv}^{hiding}_{\mathcal{COM},\mathcal{B}_3}(\lambda)$.

Proof. Let $\mathcal{B}_3$ be an adversary for the hiding of the underlying commitment scheme. $\mathcal{B}_3$ operates as with $\mathcal{B}_2$ except for the way to generating the commitments for i_β and $i_{1-\beta}$. Roughly speaking, $\mathcal{B}_3$ commits a message $0^{|\langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle|}$ to the commitment c_{i_β}. Also, $c_{i_{1-\beta}}$ is computed in the same way to generate c_{i_β} in Lemma 8.3.2, that is, $\mathcal{B}_3$ generates $c_{i_{1-\beta}}$ by accessing the challenge oracle of the hiding game. The detailed $\mathcal{B}_3$'s operation is described as follows.

In the beginning of the game, $\mathcal{B}_3$ chooses a challenge bit $\beta \in \{0,1\}$ uniformly at random. He obtains a commitment key from the challenger, and user IDs i_0 and i_1 from the adversary $\mathcal{A}$. The algorithm $\mathcal{B}_3$ generates a common reference string crs' by using S_1. Also he produces signing keys gsk_i for the user i except for users i_β and $i_{1-\beta}$, and commitments c_i by following algorithms of the scheme Π_{GS}. In terms of the user i_β, $\mathcal{B}_3$ generates a key pair $(\mathsf{vk}_{i_\beta}, \mathsf{sk}_{i_\beta})$ by following the scheme Π_{GS} and a key $k_{i_\beta} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$, and computes a commitment $c_{i_\beta} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle|}; r_{i_\beta})$. In terms of the user $i_{1-\beta}$, a key pair $(\mathsf{vk}_{i_{1-\beta}}, \mathsf{sk}_{i_{1-\beta}})$ is generated following the scheme Π_{GS}. In the same way to generate c_{i_β} in Lemma 8.3.2, a commitment $c_{i_{1-\beta}}$ is produced as follows. $\mathcal{B}_3$ generates $k_{i_{1-\beta}} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$, sends $(0^{|\langle i_{1-\beta}, \mathsf{vk}_{i_{1-\beta}}, k_{i_{1-\beta}}\rangle|}, \langle i_{1-\beta}, \mathsf{vk}_{i_{1-\beta}}, k_{i_{1-\beta}}\rangle)$ to the challenger as a challenge, and obtains a commitment c^*. Then, he sets $c_{i_{1-\beta}} \leftarrow c^*$. With the same discussion in Lemma 8.3.2, $\mathcal{B}_3$ cannot obtain the $i_{1-\beta}$'s whole signing key $\mathsf{gsk}_{i_{1-\beta}}$. However, $\mathcal{B}_3$ can simulate the game since he does not have to give the key $\mathsf{gsk}_{i_{1-\beta}}$ to $\mathcal{A}$ and can produce a signature $\sigma = (C, \pi)$ without the randomness by using the trapdoor td. For a conclusive output β' of $\mathcal{A}$, $\mathcal{B}_3$ returns $b' = 1$ if β' is same as the bit β that he chose at the beginning of the game, and $b' = 0$ otherwise.

In the case $b = 0$, the commitment $c_{i_{1-\beta}}$ for the user $i_{1-\beta}$ is $c_{i_{1-\beta}} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_{1-\beta}, \mathsf{vk}_{i_{1-\beta}}, k_{i_{1-\beta}}\rangle|}; r_{i_{1-\beta}})$, and then $\mathcal{B}_3$ perfectly simulates Game 3 for $\mathcal{A}$. On the other hand, if $b = 1$, the commitment $c_{i_{1-\beta}}$ is $c_{i_{1-\beta}} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(\langle i_{1-\beta}, \mathsf{vk}_{i_{1-\beta}}, k_{i_{1-\beta}}\rangle; r_{i_{1-\beta}})$, and $\mathcal{B}_3$ perfectly simulates Game 2 for $\mathcal{A}$. Thus as with Lemma 8.3.2, we get $\mathsf{Adv}^{hiding}_{\mathcal{COM},\mathcal{B}_3}(\lambda) = 1/2 \cdot \left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right|$. □

Lemma 5.2.4. *There exists a PPT algorithm $\mathcal{B}_4$ such that $\left|\Pr[\mathtt{Suc}_3] - \Pr[\mathtt{Suc}_4]\right| = \mathsf{Adv}^{random}_{\mathcal{SKE},\mathcal{B}_4}(\lambda)$.*

Proof. Let $\mathcal{B}_4$ be an adversary for the ciphertext-pseudorandomness of the underlying SKE scheme. At the beginning of the game with the adversary $\mathcal{A}$, $\mathcal{B}_4$ chooses a challenge bit $\beta \in \{0,1\}$. $\mathcal{B}_4$ generates a common reference string crs' by the algorithm S_1, and computes signing keys gsk_i and commitments $c_i$by following the scheme Π_{GS} except for i_β and $i_{1-\beta}$. With respect to the user i_β, $\mathcal{B}_4$ computes elements in a signing key gsk_{i_β} except for a SKE key k_{i_β}, and lets a commitment c_{i_β} be $c_{i_\beta} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle|}; r_{i_\beta})$. For the user $i_{1-\beta}$, the algorithm $\mathcal{B}_4$ generates a SKE key $k_{i_{1-\beta}}$ and computes $c_{i_{1-\beta}} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_{1-\beta}, \mathsf{vk}_{i_{1-\beta}}, k_{i_{1-\beta}}\rangle|}; r_{i_{1-\beta}})$. $\mathcal{B}_4$ answers signing queries and the challenge query

from $\mathcal{A}$ as follows. First, we consider the case that $\mathcal{A}$ queries a message m to the oracle Sign_B ($B \in \{0,1\}$). If $B = 1-\beta$, it is easy to answer the query since $\mathcal{B}_4$ has the i_B's whole signing key gsk_{i_B} including the key k_{i_B}. On the other hand, if $B = \beta$, $\mathcal{B}_4$ replies to the signing query using the oracle of the ciphertext-pseudorandomness game. Specifically, he computes $M = \langle i_\beta, \mathsf{vk}_{i_\beta}, \mathsf{SIG.Sign}(\mathsf{sk}_{i_\beta}, m)\rangle$ and queries M to the encryption oracle. Then, $\mathcal{B}_4$ computes $\pi \leftarrow \mathsf{S}_2(\mathsf{crs}', \mathsf{td}, \langle m, C, T, \mathsf{ck}\rangle)$ and returns a group signature $\sigma = (C, \pi)$ to $\mathcal{A}$ where C is the ciphertext that $\mathcal{B}_4$ obtains from the oracle. In the same way, $\mathcal{B}_4$ answers the challenge query on a message m^* by using his oracle. For a conclusive output β' of $\mathcal{A}$, $\mathcal{B}_4$ returns $b' = 1$ if β' is same as the bit β, and $b' = 0$ otherwise.

In the case that the oracle $\mathcal{B}_4$ accesses is the $\mathsf{SKE.Enc}(k, \cdot)$ oracle, ciphertexts in the group signatures of the user i_β are generated by the $\mathsf{SKE.Enc}$ algorithm. Therefore, $\mathcal{B}_4$ perfectly simulates Game 3 for $\mathcal{A}$. On the other hand in the case that the oracle $\mathcal{B}_4$ accesses is the CRand oracle, ciphertexts in the group signatures of the user i_β are uniformly random values sampled from the ciphertext space $\mathcal{C}$. Then, $\mathcal{B}_4$ perfectly simulates Game 4 for $\mathcal{A}$. Thus, it holds that $\mathsf{Adv}^{random}_{\mathcal{SKE},\mathcal{B}_4}(\lambda) = \big|\Pr[\mathcal{A}^{\mathsf{S.Enc}(k,\cdot)}(\lambda) = 1 \mid k \leftarrow \mathsf{S.Gen}(1^\lambda)] - \Pr[\mathcal{A}^{\mathsf{CRand}(\cdot)}(\lambda) = 1]\big| = \big|\Pr[\beta' = \beta \text{ in Game 3}] - \Pr[\beta' = \beta \text{ in Game 4}]\big| = \big|\Pr[\mathtt{Suc}_3] - \Pr[\mathtt{Suc}_4]\big|$. □

Lemma 5.2.5. *There exists a PPT algorithm $\mathcal{B}_5$ such that $\big|\Pr[\mathtt{Suc}_4] - \Pr[\mathtt{Suc}_5]\big| = \mathsf{Adv}^{random}_{\mathcal{SKE},\mathcal{B}_5}(\lambda)$.*

Proof. Let $\mathcal{B}_5$ be an adversary which attacks the ciphertext-pseudorandomness of the SKE scheme. $\mathcal{B}_5$ operates as with $\mathcal{B}_4$ except for the way to answer queries for the oracle Sign_B ($B \in \{0,1\}$) and the challenge oracle. Roughly speaking, if $B = \beta$ for signing queries, $\mathcal{B}_5$ samples a ciphertext C uniformly at random from the ciphertext space $\mathcal{C}$, simulates π by using the simulator S_2, and returns $\sigma = (C, \pi)$. For the challenge query, $\mathcal{B}_5$ replies in the same way. If $B = 1 - \beta$, $\mathcal{B}_5$ answers a query by using his encryption oracle as with the case of $B = \beta$ for the algorithm $\mathcal{B}_4$. The detailed $\mathcal{B}_5$'s operation is described as follows.

At the beginning of the game, $\mathcal{B}_5$ chooses a challenge bit $\beta \in \{0,1\}$. $\mathcal{B}_5$ generates a common reference string crs' by the algorithm S_1, and computes signing keys gsk_i and commitments c_i except for i_β and $i_{1-\beta}$ following the scheme Π_{GS}. For the user i_β, $\mathcal{B}_5$ generates the signing key gsk_β by following the algorithm of Π_{GS}, and lets a commitment c_{i_β} be $c_{i_\beta} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_\beta, \mathsf{vk}_{i_\beta}, k_{i_\beta}\rangle|}; r_{i_\beta})$. In terms of the user $i_{1-\beta}$, $\mathcal{B}_5$ computes all the elements in a signing key $\mathsf{gsk}_{i_{1-\beta}}$ except for a SKE key $k_{i_{1-\beta}}$, and lets a commitment $c_{i_{1-\beta}}$ be $c_{i_{1-\beta}} \leftarrow \mathsf{Commit}_{\mathsf{ck}}(0^{|\langle i_{1-\beta}, \mathsf{vk}_{i_{1-\beta}}, k_{i_{1-\beta}}\rangle|}; r_{i_{1-\beta}})$. The algorithm $\mathcal{B}_5$ answers signing queries and the challenge query from $\mathcal{A}$ as follows. First, we explain the case that $\mathcal{A}$ queries a message m to the oracle Sign_B ($B \in \{0,1\}$). If $B = \beta$, $\mathcal{B}_5$ samples a ciphertext C uniformly at random from the ciphertext space $\mathcal{C}$ and computes $\pi \leftarrow \mathsf{S}_2(\mathsf{crs}', \mathsf{td}, \langle m, C, T, \mathsf{ck}\rangle)$. Then

he returns $\sigma = (C, \pi)$ as a signature of a message m to $\mathcal{A}$. In the same way for the challenge query, $\mathcal{B}_5$ samples C^* from the space $\mathcal{C}$ uniformly at random, simulates a proof π^* using the algorithm S_2, and returns the challenge signature $\sigma^* = (C^*, \pi^*)$. In the case that $\mathcal{A}$ queries the oracle $\mathsf{Sign}_{1-\beta}$ on a message m, $\mathcal{B}_5$ answers the query by using the encryption oracle of the ciphertext-pseudorandomness game. More precisely, $\mathcal{B}_5$ computes $M = \langle i_{1-\beta}, \mathsf{vk}_{i_{1-\beta}}, \mathsf{SIG.Sign}(\mathsf{sk}_{i_{1-\beta}}, m)\rangle$, queries M to the encryption oracle, and obtains a ciphertext C. Then, he computes a proof $\pi \leftarrow \mathsf{S}_2(\mathsf{crs}', \mathsf{td}, \langle m, C, T, \mathsf{ck}\rangle)$ and returns a group signature $\sigma = (C, \pi)$ to the adversary $\mathcal{A}$. For a conclusive output β' of $\mathcal{A}$, $\mathcal{B}_5$ returns $b' = 1$ if β' is same as the bit β, and $b' = 0$ otherwise.

If $\mathcal{B}_5$ accesses the $\mathsf{S.Enc}(k, \cdot)$ oracle, the ciphertexts in group signatures for the user $i_{1-\beta}$ are generated by the encryption algorithm $\mathsf{SKE.Enc}$. Then, $\mathcal{B}_5$ perfectly simulates Game 4 for $\mathcal{A}$. On the other hand, if $\mathcal{B}_5$ accesses the CRand oracle, the ciphertexts in group signatures for the user $i_{1-\beta}$ are chosen uniformly at random from the ciphertext space $\mathcal{C}$. Therefore, $\mathcal{B}_5$ perfectly simulates Game 5 for $\mathcal{A}$. That is, it holds that $\mathsf{Adv}^{random}_{\mathcal{SKE},\mathcal{B}_5}(\lambda) = \big|\Pr[\mathcal{A}^{\mathsf{S.Enc}(k,\cdot)}(\lambda) = 1 \mid k \leftarrow \mathsf{SKE.Gen}(1^\lambda)] - \Pr[\mathcal{A}^{\mathsf{CRand}(\cdot)}(\lambda) = 1]\big| = \big|\Pr[\beta' = \beta \text{ in Game 4}] - \Pr[\beta' = \beta \text{ in Game 5}]\big| = \big|\Pr[\mathtt{Suc}_4] - \Pr[\mathtt{Suc}_5]\big|$. □

In Game 5, the choice of the challenge bit b and the distribution of the challenge signature $\sigma^* = (C^*, \pi^*)$ are independent. Moreover, the instance of the selfless anonymity game and the responses of queries are symmetrical with respect to the challenge users i_0 and i_1. Thus, we can say that $\Pr[\mathtt{Suc}_5] = 1/2$. From this fact and Lemma 8.3.1 to Lemma 8.3.5, we get

$$\begin{aligned}\mathsf{Adv}^{anon}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda, n) \leq& \sum_{i=0}^{4}\big|\Pr[\mathtt{Suc}_i] - \Pr[\mathtt{Suc}_{i+1}]\big| + \big|\Pr[\mathtt{Suc}_5] - 1/2\big| \\ \leq& \mathsf{Adv}^{zk}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{hiding}_{\mathcal{COM},\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{hiding}_{\mathcal{COM},\mathcal{B}_3}(\lambda) \\ &+ \mathsf{Adv}^{random}_{\mathcal{SKE},\mathcal{B}_4}(\lambda) + \mathsf{Adv}^{random}_{\mathcal{SKE},\mathcal{B}_5}(\lambda).\end{aligned}$$

Since the choice of the parameter n and the adversary $\mathcal{A}$ is arbitrarily, our scheme Π_{GS} satisfies selfless anonymity. □

Remark. In order to investigate the feasibility of constructing group signature schemes with minimum security requirements, we employ selfless CPA-anonymity in which an adversary is not allowed to access the open oracle. However, a OWF and a NIZK proof system are still enough to construct a selfless CCA-anonymous group signature scheme in the standard model. Briefly, the underlying SKE scheme is replaced to a tag-based CCA secure SKE scheme, a strong one-time signature scheme is employed in such a way that a one-time verification key is used as a tag of the SKE scheme, and a strong one-time signature for the ciphertext and a NIZK proof is generated. Since a tag-based CCA secure

SKE scheme can be constructed from a OWF by slightly modifying our SKE scheme and a strong one-time signature scheme also can be constructed from a OWF [98, 92], a selfless CCA-anonymous group signature scheme can also be constructed from a OWF and a NIZK proof system in the standard model. We will give the detail of the construction of a selfless CCA-anonymous group signature scheme in the full version of this thesis.

Next, we prove that our group signature scheme is traceable.

Theorem 5.2.3. *The scheme* Π_{GS} *is traceable if the underlying NIZK proof system satisfies soundness, the underlying commitment scheme is statistical binding, the underlying SKE scheme satisfies key-robustness, and the underlying signature scheme satisfies EUF-CMA security.*

Proof. Let $\mathcal{A}$ be an adversary that attacks traceability of Π_{GS}. We can classify $\mathcal{A}$'s forgery $(m^*, (C^*, \pi^*))$ into the following types. In the following, k_i is a SKE key of the user i, ck is a commitment key, and T is the set of commitments c_i for $\langle i, \mathsf{vk}_i, k_i \rangle$.

Type 1: It holds that $\langle m^*, C^*, T, \mathsf{ck} \rangle \notin L$.
Type 2: It holds that $\langle m^*, C^*, T, \mathsf{ck} \rangle \in L$, and there exist two indices $i \in [1, n]$ such that $\perp \neq \mathsf{SKE.Dec}(k_i, C^*)$.
Type 3: It holds that $\langle m^*, C^*, T, \mathsf{ck} \rangle \in L$, and for all $i \in [1, n]$, $\mathsf{SKE.Dec}(k_i, C^*) = \perp$ holds.
Type 4: It holds that $\langle m^*, C^*, T, \mathsf{ck} \rangle \in L$, there exists exact one index $i \in [1, n]$ such that $\mathsf{SKE.Dec}(k_i, C^*) \neq \perp$, and $i^* \notin [1, n]$ for the decryption result $\langle i^*, \mathsf{vk}^*, s^* \rangle$.
Type 5: It holds that $\langle m^*, C^*, T, \mathsf{ck} \rangle \in L$, there exists exact one index $i \in [1, n]$ such that $\mathsf{SKE.Dec}(k_i, C^*) \neq \perp$, and $i^* \in [1, n]$ for the decryption result $\langle i^*, \mathsf{vk}^*, s^* \rangle$.

Let E_j denote the event that $\mathcal{A}$ outputs a forgery which is in Type j. For each event, the following lemmas hold.

Lemma 5.2.6. *There exists a PPT algorithm* $\mathcal{B}_1$ *such that* $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}_L, \mathcal{B}_1}(\lambda)$.

Proof. Let $\mathcal{B}_1$ be an adversary who attacks the soundness for the NIZK proof system. $\mathcal{B}_1$ generates the instance of the scheme Π_{GS} except for a common reference string. He uses the common reference string given by the challenger of the soundness game. Since $\mathcal{B}_1$ has all the user signing keys, he can easily simulate the Sign oracle and the $\mathsf{Corrupt}$ oracle. When $\mathcal{A}$ terminates with an output $(m^*, (C^*, \pi^*))$, $\mathcal{B}_1$ outputs $(x^*, \pi^*) = (\langle m^*, C^*, T, \mathsf{ck} \rangle, \pi^*)$ as a forgery in the soundness game.

If the event E_1 occurs, it holds that $x^* \notin L$ and $\mathsf{ZK.Verify}(\mathsf{crs}, \langle m^*, C^*, T, \mathsf{ck} \rangle, \pi^*) = 1$ from the winning condition of $\mathcal{A}$ and the condition of Type 1. Thus, (x^*, π^*) is a forgery for the soundness of the NIZK proof system. Then, we get $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}_L, \mathcal{B}_1}(\lambda)$. □

Lemma 5.2.7. *There exists a PPT algorithm* $\mathcal{B}_2$ *such that* $(2/n(n-1)) \cdot \Pr[\mathsf{E}_2] \leq \mathsf{Adv}^{key\text{-}robust}_{\mathcal{SKE},\mathcal{B}_2}(\lambda)$.

Proof. Let $\mathcal{B}_2$ be an adversary who attacks the key-robustness of the SKE scheme. At the beginning of the game, $\mathcal{B}_2$ receives two keys k and $\widetilde{k}$ from the challenger of the key-robustness game. Then, $\mathcal{B}_2$ guesses two indices $j, \widetilde{j} \in [1, n]$ randomly at the beginning of the game, and sets $k_j \leftarrow k$ and $k_{\widetilde{j}} \leftarrow \widetilde{k}$. The other user i's SKE key k_i and the other instance of the scheme Π_{GS} are generated by $\mathcal{B}_2$ himself. As with the algorithm $\mathcal{B}_1$ in Lemma 5.2.6, since $\mathcal{B}_2$ has all the users' signing keys, he is able to answer all the queries form $\mathcal{A}$. When $\mathcal{A}$ terminates with an output $(m^*, (C^*, \pi^*))$ and the event E_2 occurs, there exist more than two indices $i \in [1, n]$ such that $\mathsf{SKE.Dec}(k_i, C^*) \neq \perp$ by the condition of Type 2. If it does not hold $\mathsf{SKE.Dec}(k_j, C^*) \neq \perp$ and $\mathsf{SKE.Dec}(k_{\widetilde{j}}, C^*) \neq \perp$, $\mathcal{B}_2$ aborts with output $\perp$. Otherwise, $\mathcal{B}_2$ outputs C^* as a forgery of the key-robustness for the SKE scheme.

Since $k_j = k$ and $k_{\widetilde{j}} = \widetilde{k}$ hold if $\mathcal{B}_2$ does not abort, we get $\mathsf{SKE.Dec}(k, C^*) \neq \perp$ and $\mathsf{SKE.Dec}(\widetilde{k}, C^*) \neq \perp$. That is, the ciphertext C^* is a forgery of the key-robustness for the SKE scheme. Since $\mathcal{B}_2$'s prediction of indices $j, \widetilde{j} \in [1, n]$ and the behavior of $\mathcal{A}$ are independent, it holds that $(2/n(n-1)) \cdot \Pr[\mathsf{E}_2] \leq \mathsf{Adv}^{key\text{-}robust}_{\mathcal{SKE},\mathcal{B}_2}(\lambda)$. □

Lemma 5.2.8. *There exists an unbounded algorithm* $\mathcal{B}_3$ *such that* $\Pr[\mathsf{E}_3] \leq \mathsf{Adv}^{binding}_{\mathcal{COM},\mathcal{B}_3}(\lambda)$.

Proof. We define the event $\mathtt{Bad}$ as follows.

$\mathtt{Bad}$**:** For a commitment key ck that is generated in the traceability game, there exists a pair $(M, r, \widehat{M}, \widehat{r})$ such that $\mathsf{Commit}_{\mathsf{ck}}(M; r) = \mathsf{Commit}_{\mathsf{ck}}(\widehat{M}; \widehat{r}) \ \wedge \ M \neq \widehat{M}$.

Let $\mathcal{B}_3$ be an unbounded algorithm who is given a commitment key ck and attempts to find the pair $(M, r, \widehat{M}, \widehat{r})$ such that $\mathsf{Commit}_{\mathsf{ck}}(M; r) = \mathsf{Com}_{\mathsf{ck}}(\widehat{M}; \widehat{r}) \wedge M \neq \widehat{M}$ in order to break the statistical binding. Then, it holds that

$$\begin{aligned}\Pr[\mathsf{E}_3] &= \Pr[\mathtt{Bad} \wedge \mathsf{E}_3] + \Pr[\neg\mathtt{Bad} \wedge \mathsf{E}_3] \\ &\leq \Pr[\mathtt{Bad}] + \Pr[\neg\mathtt{Bad} \wedge \mathsf{E}_3] \\ &\leq \mathsf{Adv}^{binding}_{\mathcal{COM},\mathcal{B}_3}(\lambda) + \Pr[\neg\mathtt{Bad} \wedge \mathsf{E}_3].\end{aligned}$$

Moreover, we show that it holds that $\Pr[\neg\mathtt{Bad} \wedge \mathsf{E}_3] = 0$ in the following. By the condition of the event $\neg\mathtt{Bad}$, there is no pair $(M, r, \widehat{M}, \widehat{r})$ such that $\mathsf{Commit}_{\mathsf{ck}}(M; r) = \mathsf{Commit}_{\mathsf{ck}}(\widehat{M}; \widehat{r}) \wedge M \neq \widehat{M}$ for the commitment key ck given to $\mathcal{A}$. This means that for all the commitments c_i in the list T, a pair $(\langle i, \mathsf{vk}_i, k_i \rangle, r_i)$ which satisfies that $\mathsf{Commit}_{\mathsf{ck}}(\langle i, \mathsf{vk}_i, k_i \rangle; r_i) = c_i$ is unique. Also by the condition of Type 3, it holds that $x = \langle m^*, C^*, T, \mathsf{ck} \rangle \in L$. Let a witness w for the statement x be $\langle i^*, \mathsf{vk}^*, s^*, R^*, k^*, r^* \rangle$. Then for at least one index

i, it holds that $c_i = \mathsf{Commit}_{\mathsf{ck}}(\langle i^*, \mathsf{vk}^*, k^* \rangle; r^*)$ from the condition $(\tilde{c})$ of belonging the language L. Therefore, $(\langle i^*, \mathsf{vk}^*, k^* \rangle, r^*) = (\langle i, \mathsf{vk}_i, k_i \rangle, r_i)$ holds. Moreover since it holds that $C^* = \mathsf{SKE.Enc}(k^*, \langle i^*, \mathsf{vk}^*, s^* \rangle; R^*)$ by the condition $(\tilde{a})$ of belonging the language L, we can get $\perp \neq \mathsf{SKE.Dec}(k^*, C^*) = \mathsf{SKE.Dec}(k_i, C^*)$. However, this contradicts the latter part of the condition in Type 3 "for all the indices $i \in [1, n]$, it holds that $\mathsf{SKE.Dec}(k_i, C^*) = \perp$". Thus, $\Pr[\neg \mathsf{Bad} \wedge \mathsf{E}_3] = 0$ holds, and we get $\Pr[\mathsf{E}_3] \leq \mathsf{Adv}^{binding}_{\mathcal{COM}, \mathcal{B}_3}(\lambda)$. □

Lemma 5.2.9. *There exists an unbounded algorithm $\mathcal{B}_4$ such that* $\Pr[\mathsf{E}_4] \leq \mathsf{Adv}^{binding}_{\mathcal{COM}, \mathcal{B}_4}(\lambda)$.

Proof. As with the proof of Lemma 5.2.8, we define the event Bad. Let $\mathcal{B}_4$ be an unbounded algorithm who is given a commitment key ck and attempts to find the pair $(M, r, \widehat{M}, \widehat{r})$ such that $\mathsf{Commit}_{\mathsf{ck}}(M; r) = \mathsf{Commit}_{\mathsf{ck}}(\widehat{M}; \widehat{r}) \wedge M \neq \widehat{M}$. As well as the analysis of the probability $\Pr[\mathsf{E}_3]$, it holds that $\Pr[\mathsf{E}_4] \leq \mathsf{Adv}^{binding}_{\mathcal{COM}, \mathcal{B}_4}(\lambda) + \Pr[\neg \mathsf{Bad} \wedge \mathsf{E}_4]$ for an unbounded algorithm $\mathcal{B}_4$. In the following, we prove that $\Pr[\neg \mathsf{Bad} \wedge \mathsf{E}_4] = 0$. By the conditions of Type 4, the statement $x = \langle m^*, C^*, T, \mathsf{ck} \rangle$ is included in the language L. Therefore, there exists a witness $\langle i^*, \mathsf{vk}^*, s^*, R^*, k^*, r^* \rangle$, and it holds that $\mathsf{SKE.Enc}(k^*, \langle i^*, \mathsf{vk}^*, s^* \rangle; R^*) = C^*$ and $\mathsf{Commit}_{\mathsf{ck}}(\langle i^*, \mathsf{vk}^*, k^* \rangle; r^*) = c_j$ for at least one index $j \in [1, n]$. Moreover by the condition of the event $\neg \mathsf{Bad}$, a pair $(\langle i, \mathsf{vk}_i, k_i \rangle, r_i)$ which satisfies that $\mathsf{Commit}_{\mathsf{ck}}(\langle i, \mathsf{vk}_i, k_i \rangle; r_i) = c_i$ is unique for all the commitments c_i in the list T. Then, we get $\langle i^*, \mathsf{vk}^*, k^* \rangle = \langle j, \mathsf{vk}_j, k_j \rangle$, that is, $k^* = k_j$. By the conditions of Type 4, k_j is the unique key that does not decrypt C^* to $\perp$. Therefore, it holds that $\mathsf{SKE.Dec}(k_j, C^*) = \langle i^*, \mathsf{vk}^*, s^* \rangle = \langle j, \mathsf{vk}_j, s^* \rangle$, that is, $i^* = j \in [1, n]$. However, it contradicts the conditions of Type 4 "$i^* \notin [1, n]$". Thus, $\Pr[\neg \mathsf{Bad} \wedge \mathsf{E}_4] = 0$. □

Lemma 5.2.10. *There exists a PPT algorithm $\mathcal{B}_5$ such that* $(1/n) \cdot \Pr[\mathsf{E}_5] \leq \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG}, \mathcal{B}_5}(\lambda)$.

Proof. Let $\mathcal{B}_5$ be an adversary that attacks the EUF-CMA security of the underlying signature scheme. $\mathcal{B}_5$ guesses an index $i^* \in [1, n]$ randomly at the beginning of the game, and then sets the key vk^* given by the challenger to the user i^*'s verification key. The rest of the instance of the scheme Π_{GS} is generated by $\mathcal{B}_5$ himself. We note that $\mathcal{B}_5$ cannot obtain the whole user i^*'s signing key gsk_{i^*} since he does not know the signing key sk^* corresponding to the verification key vk^*. On the other hand, $\mathcal{B}_5$ has the SKE secret key k_{i^*} and the randomness r_{i^*} for the commitment c_{i^*} whose message is $\langle i^*, \mathsf{vk}^*, k_{i^*} \rangle$ since he generates them by himself. $\mathcal{B}_5$ answers queries from $\mathcal{A}$ with respect to the $\mathsf{Corrupt}$ oracle and the Sign oracle as follows.

- The $\mathsf{Corrupt}$ oracle: For a query j from the adversary $\mathcal{A}$, if $j \neq i^*$, $\mathcal{B}_5$ returns gsk_j. On the other hand if $j = i^*$, $\mathcal{B}_5$ aborts the game with output $\perp$.

- The Sign oracle: Let (j, m) be a query from $\mathcal{A}$. If $j \neq i^*$, $\mathcal{B}_5$ generates a group signature $\sigma = (C, \pi)$ by using the user j's signing key gsk_j that he possesses. If $j = i^*$, $\mathcal{B}_5$ queries m to the Sign oracle of the EUF-CMA game and obtains $s \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i^*}, m)$. Then, he computes $C \leftarrow \mathsf{SKE.Enc}(k_{i^*}, \langle i^*, \mathsf{vk}^*, s\rangle; R)$ and $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, \langle m, C, T, \mathsf{ck}\rangle, \langle i^*, \mathsf{vk}^*, s, R, k_{i^*}, r_{i^*}\rangle)$ where R is a randomness. Finally, $\mathcal{B}_5$ returns $\sigma = (C, \pi)$ to $\mathcal{A}$.

Let $(m^*, (C^*, \pi^*))$ be the conclusive output of the adversary $\mathcal{A}$. By the conditions of Type 5, the ciphertext C^* can be decrypted under only one key k_i, and for the decryption result $\langle i', \mathsf{vk}', s^*\rangle \leftarrow \mathsf{SKE.Dec}(k_i, C^*)$, it holds that $i' \in [1, n]$. If $i^* \neq i'$, $\mathcal{B}_5$ aborts the game with output $\perp$. On the other hand, $\mathcal{B}_5$ outputs (m^*, s^*) as a forgery.

Since it holds that $\langle m^*, C^*, T, \mathsf{ck}\rangle \in L$ by the condition of Type 5, $\mathsf{SIG.Verify}(\mathsf{vk}^*, (m^*, s^*)) = 1$ holds. Also by the winning condition of $\mathcal{A}$'s forgery, the pair (i^*, m^*) is not queried to the Sign oracle. Therefore, the message m^* is also not queried to the Sign oracle in the EUF-CMA game, and (m^*, s^*) is a forgery for the EUF-CMA security of the underlying signature scheme. Since $\mathcal{B}_5$'s prediction of indices $i^* \in [1, n]$ and the behavior of $\mathcal{A}$ are independent, we get $(1/n) \cdot \Pr[\mathsf{E}_5] \leq \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}_5}(\lambda)$. □

From Lemma 5.2.6 to 5.2.10, it holds that

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{trace}}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda, n) &= \Pr[\ \mathsf{Exp}^{\mathrm{trace}}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda, n) = 1\ ] \\
&= \Pr[\ \mathsf{E}_1 \vee \mathsf{E}_2 \vee \mathsf{E}_3 \vee \mathsf{E}_4 \vee \mathsf{E}_5\ ] \\
&= \Pr[\mathsf{E}_1] + \Pr[\mathsf{E}_2] + \Pr[\mathsf{E}_3] + \Pr[\mathsf{E}_4] + \Pr[\mathsf{E}_5] \\
&\leq \mathsf{Adv}^{sound}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) + (n(n-1)/2) \cdot \mathsf{Adv}^{key\text{-}robust}_{\mathcal{SKE},\mathcal{B}_2}(\lambda) \\
&\qquad + \mathsf{Adv}^{binding}_{\mathcal{COM},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{binding}_{\mathcal{COM},\mathcal{B}_4}(\lambda) + n \cdot \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}_5}(\lambda).
\end{aligned}$$

The choice of the parameter n and the adversary $\mathcal{A}$ is arbitrarily, and n is polynomial in λ. Therefore, our scheme Π_{GS} satisfies traceability. □

From Theorem 5.2.1 to 5.2.3, we see that a selfless anonymous group signature scheme can be constructed from a OWF and a NIZK proof system in the standard model.

5.2.4 A Symmetric Key Encryption Scheme with Key-robustness and Ciphertext-pseudorandomness

In this section, we give a construction of the SKE scheme that satisfies key-robustness and ciphertext-pseudorandomness from a PRF. Moreover, we prove that the SKE scheme satisfies the two security notions.

Description. We give the SKE scheme in Figure 5.2. We use a PRF family $\mathsf{PRF}(\cdot,\cdot) : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^{\ell+2\lambda}$ where ℓ is the message length of the SKE scheme. In our construction, a secret key consists of two parts. One is a PRF key $K \in \{0,1\}^\lambda$ and the other is a key $t \in \{0,1\}^{2\lambda}$ for checking the validity of ciphertexts. To encrypt a message m, the concatenation of m and t is masked with the PRF's output $\mathsf{PRF}(K,r)$ where r is a random input, that is, $\widehat{C} \leftarrow \mathsf{PRF}(K,r) \oplus (m\|t)$. Then, the ciphertext C is a pair of r and $\widehat{C}$. Intuitively, the SKE scheme satisfies ciphertext-pseudorandomness by the pseudorandomness of the PRF. Also, we can prove that the scheme satisfies key-robustness information-theoretically by the technique of the counting argument that is reminiscent of the Naor commitment scheme [91]. The detailed proofs are given in the next section.

$\mathsf{SKE.Gen}(1^\lambda)$:
$K \xleftarrow{\$} \{0,1\}^\lambda$; $t \xleftarrow{\$} \{0,1\}^{2\lambda}$; $k \leftarrow (K,t)$
Return k
$\mathsf{SKE.Enc}(k,m)$:
$(K,t) \leftarrow k$; $r \xleftarrow{\$} \{0,1\}^\lambda$
$\widehat{C} \leftarrow \mathsf{PRF}(K,r) \oplus (m\|t)$; $C \leftarrow (r,\widehat{C})$
Return C
$\mathsf{SKE.Dec}(k,C)$:
$(K,t) \leftarrow k$; $(r,\widehat{C}) \leftarrow C$
$m'\|t' \leftarrow \widehat{C} \oplus \mathsf{PRF}(K,r)$
Return m' if $t' = t$, otherwise return $\perp$

Figure 5.2: The SKE Scheme with Ciphertext-pseudorandomness and Key-robustness

Security Proofs. Here, we prove that the scheme in Figure 5.2 satisfies key-robustness and ciphertext-pseudorandomness.

Theorem 5.2.4. *The scheme* Π_{SKE} *satisfies ciphertext-pseudorandomness if the function family* PRF *is pseudorandom.*

Proof. Let $\mathcal{A}$ be an adversary that attacks the ciphertext-pseudorandomness of Π_{SKE}. We consider the following sequence of games. Let T_i denote the event that $\mathcal{A}$ outputs 1 in Game i.

[Game 0]: This is the experiment of the ciphertext-pseudorandomness for Π_{SKE} when $\mathcal{A}$ accesses the $\mathsf{SKE.Enc}(k,\cdot)$ oracle.

[Game 1]: In this game, the pseudorandom function PRF is changed to a random function. More precisely, we replace the function PRF with the function Rand which is sampled from

the set $\mathcal{F}(\lambda, \ell+2\lambda)$ uniformly at random where $\mathcal{F}(\lambda, \ell+2\lambda)$ is the set of all the possible functions $F : \{0,1\}^{\lambda} \to \{0,1\}^{\ell+2\lambda}$.

[Game 2]: From Game 2, the function Rand is implemented by lazy sampling. Specifically, Rand maintains the list L which stores query/answer pairs so far, and runs as follows. If a value r is queried to the Rand oracle, then Rand first checks whether there is an entry of the form (r, s) in L. If so, the Rand oracle returns s. Otherwise, Rand samples s from $\{0,1\}^{\ell+2\lambda}$ uniformly at random and returns the value s. Also, Rand adds a pair (r, s) to the list L. The difference between Game 1 and Game 2 is only conceptual. Therefore, we get $\Pr[\mathsf{T}_1] = \Pr[\mathsf{T}_2]$.

[Game 3]: Here, we change the way to answer the $\mathcal{A}$'s oracle queries. From this game, when $\mathcal{A}$ queries a message m, the challenger samples a uniformly random value $s \in \{0,1\}^{\ell+2\lambda}$ every time and computes $\widehat{C} \leftarrow s \oplus (m\|t)$. In other words, the challenger does not perform any consistency checks in calculating the value s. That is, the challenger samples random values $r \in \{0,1\}^{\lambda}$ and $s \in \{0,1\}^{\ell+2\lambda}$, and returns $C = (r, \widehat{C})$ to $\mathcal{A}$ as a ciphertext.

In Game 3, the answer for the $\mathcal{A}$'s query m is an uniform random value in the ciphertext space $\mathcal{C}$. Therefore, Game 3 is identical to the experiment of the ciphertext-pseudorandomness in which $\mathcal{A}$ accesses the CRand oracle. Now, for the $\mathcal{A}$'s advantage $\mathsf{Adv}^{random}_{\Pi_{\mathrm{SKE}},\mathcal{A}}(\lambda)$,

$$\mathsf{Adv}^{random}_{\Pi_{\mathrm{SKE}},\mathcal{A}}(\lambda) = \left|\Pr[\mathsf{T}_0] - \Pr[\mathsf{T}_3]\right| \leq \sum_{i=0}^{2} \left|\Pr[\mathsf{T}_i] - \Pr[\mathsf{T}_{i+1}]\right|$$

holds. Also, the following lemmas hold.

Lemma 5.2.11. *There exists a PPT algorithm $\mathcal{B}_1$ such that $\left|\Pr[\mathsf{T}_0] - \Pr[\mathsf{T}_1]\right| = \mathsf{Adv}^{prf}_{\mathsf{PRF},\mathcal{B}_1}(\lambda)$.*

Proof. Let $\mathcal{B}_1$ be an adversary for the pseudorandomness of the function PRF. $\mathcal{B}_1$ accesses either a function $\mathsf{PRF}(K, \cdot)$ or a function Rand where $K \in \{0,1\}^{\lambda}$ is an uniform random key and Rand is a function sampled from the set $\mathcal{F}(\lambda, \ell+2\lambda)$ uniformly at random. In the beginning of the ciphertext-pseudorandomness game with $\mathcal{A}$, the algorithm $\mathcal{B}_1$ samples a key $t \in \{0,1\}^{2\lambda}$. For the $\mathcal{A}$'s query m, $\mathcal{B}_1$ answers as follows.

1. Choose a random value $r \in \{0,1\}^{\lambda}$ and query r to $\mathcal{B}_1$'s oracle. Then, get a value s.
2. Compute $\widehat{C} \leftarrow s \oplus (m\|t)$ and return $C = (r, \widehat{C})$ to $\mathcal{A}$ as a ciphertext.

When $\mathcal{A}$ terminates with output β', $\mathcal{B}_1$ also outputs β'.

If the function $\mathcal{B}_1$ accesses is the pseudorandom function $\mathsf{PRF}(K,\cdot)$, $\mathcal{B}_1$ perfectly simulates Game 0 for $\mathcal{A}$. On the other hand if the function $\mathcal{B}_1$ accesses is the function Rand, $\mathcal{B}_1$ perfectly simulates Game 1 for $\mathcal{A}$. That is, it holds that $\mathsf{Adv}^{prf}_{\mathsf{PRF},\mathcal{B}_1}(\lambda) = \left|\Pr[\mathcal{B}_1^{\mathsf{PRF}(K,\cdot)}(\lambda) = 1 \mid K \xleftarrow{\$} \{0,1\}^\lambda] - \Pr[\mathcal{B}_1^{\mathsf{Rand}(\cdot)}(\lambda) = 1 \mid \mathsf{Rand} \xleftarrow{\$} \mathcal{F}(\lambda, \ell+2\lambda)]\right| = \left|\Pr[\beta' = 1 \text{ in Game } 0] - \Pr[\beta' = 1 \text{ in Game } 1]\right| = \left|\Pr[\mathsf{T}_0] - \Pr[\mathsf{T}_1]\right|$. □

Lemma 5.2.12. *Let q be the number of $\mathcal{A}$'s oracle queries. It holds that $\left|\Pr[\mathsf{T}_2] - \Pr[\mathsf{T}_3]\right| \leq q^2/2^\lambda$.*

Proof. For $1 \leq i \leq q$, let r_i be the randomness sampled in answering the $\mathcal{A}$'s i-th query where q is a polynomial in λ. We define the event $\mathtt{Bad}_\ell$ as follows.

$\mathtt{Bad}_\ell$**:** The event that there exist different indices i and j such that $r_i = r_j$ in Game ℓ.

Game 2 and Game 3 are identical unless the event $\mathtt{Bad}_\ell$ occurs. That is, we get $\Pr[\mathsf{T}_2 \wedge \neg\mathtt{Bad}_2] = \Pr[\mathsf{T}_3 \wedge \neg\mathtt{Bad}_3]$. Therefore, it holds that

$$\begin{aligned}\left|\Pr[\mathsf{T}_2] - \Pr[\mathsf{T}_3]\right| &= \left|\Pr[\mathsf{T}_2 \wedge \mathtt{Bad}_2] + \Pr[\mathsf{T}_2 \wedge \neg\mathtt{Bad}_2] - \Pr[\mathsf{T}_3 \wedge \mathtt{Bad}_3] - \Pr[\mathsf{T}_3 \wedge \neg\mathtt{Bad}_3]\right| \\ &= \left|\Pr[\mathsf{T}_2 \wedge \mathtt{Bad}_2] - \Pr[\mathsf{T}_3 \wedge \mathtt{Bad}_3]\right| \\ &\leq \Pr[\mathtt{Bad}_3].\end{aligned}$$

Moreover by the definition of the event $\mathtt{Bad}_3$, $\Pr[\mathtt{Bad}_3] \leq q(q-1)/2^\lambda$ holds. Thus, it holds that $\left|\Pr[\mathsf{T}_2] - \Pr[\mathsf{T}_3]\right| \leq q^2/2^\lambda$. □

From the fact that $\Pr[\mathsf{T}_1] = \Pr[\mathsf{T}_2]$, Lemma 5.2.11, and Lemma 5.2.12,

$$\mathsf{Adv}^{random}_{\Pi_{\mathrm{SKE}},\mathcal{A}}(\lambda) \leq \sum_{i=0}^{2}\left|\Pr[\mathsf{T}_i] - \Pr[\mathsf{T}_{i+1}]\right| \leq \mathsf{Adv}^{prf}_{\mathsf{PRF},\mathcal{B}_1}(\lambda) + q^2/2^\lambda$$

holds. The scheme Π_{SKE} satisfies ciphertext-pseudorandomness since q is a polynomial in λ. □

Next, we prove that our scheme satisfies key-robustness. The scheme achieves key-robustness information-theoretically unlike the case of the ciphertext-pseudorandomness.

Theorem 5.2.5. *The scheme Π_{SKE} satisfies key-robustness.*

Proof. We show that all the unbounded adversaries $\mathcal{A}$ can break the key-robustness of Π_{SKE} with only negligible probability. Let $k_1 = (K_1, t_1)$ and $k_2 = (K_2, t_2)$ be the keys that $\mathcal{A}$ obtains in the key-robustness game. In order to break the key-robustness, $\mathcal{A}$ has to output a ciphertext $C^* = (r^*, \widehat{C}^*) \in \{0,1\}^\lambda \times \{0,1\}^{\ell+2\lambda}$ such that

$$\mathsf{PRF}(K_1, r^*) \oplus \widehat{C}^* = m \| t_1 \ \wedge \ \mathsf{PRF}(K_2, r^*) \oplus \widehat{C}^* = m' \| t_2 \tag{5.2}$$

for some two messages $m, m' \in \{0,1\}^\ell$.

The necessary conditions for existing a ciphertext $C^* = (r^*, \widehat{C}^*)$ which satisfies the condition (5.2) is that there exist $m'' \in \{0,1\}^\ell$ and $r^* \in \{0,1\}^\lambda$ such that

$$\mathsf{PRF}(K_1, r^*) \oplus \mathsf{PRF}(K_2, r^*) = m'' \| (t_1 \oplus t_2). \tag{5.3}$$

Since r^* is a λ-bit string, there are at most 2^λ possible values $\mathsf{PRF}(K_1, r^*) \oplus \mathsf{PRF}(K_2, r^*)$ for the fixed keys K_1 and K_2. To satisfy the equation (5.3), at least it is necessary that the least 2λ significant bits of the both sides should correspond. When K_1, t_1, K_2, t_2 are chosen uniformly at random, the probability that the least 2λ significant bits of the both sides in the equation (5.3) correspond is at most $2^\lambda / 2^{2\lambda} = 2^{-\lambda}$. Therefore, the adversary $\mathcal{A}$ cannot break the key-robustness of Π_{SKE} with more than $2^{-\lambda}$ probability, which is negligible. □

5.2.5 The Construction in the Random Oracle Model

In Section 5.2.2, we give a construction of a selfless anonymous group signature scheme from a SKE scheme with ciphertext-pseudorandomness and key-robustness, a commitment scheme with computational hiding and statistical binding, a signature scheme, and a NIZK proof system. In addition, we show that the underlying SKE scheme can be constructed from a PRF in Section 5.2.4. Since it is known that a PRF can be constructed from a OWF in previous works [64, 58], our result implies that the SKE scheme can be constructed from a OWF. Moreover, a commitment scheme with computational hiding and statistical binding, and a signature scheme can be constructed from a OWF [91, 98]. As for a NIZK proof system (for any NP language), it is known that it can be constructed from a OWF in the random oracle model. This fact can be obtained from the following theorems.

Theorem 5.2.6 ([53, 59])**.** *If a OWF exists, a sigma-protocol for any NP language can be constructed.*

Theorem 5.2.7 ([52])**.** *A sigma-protocol for a NP language L can be transformed to a NIZK proof system for L in the random oracle model.*

Therefore, our result shown in Section 5.2.2 implies that a selfless anonymous group signature scheme can be constructed only from a OWF in the random oracle model. Strictly speaking, the corollary does not be directly implied since the security of our scheme is proved in the standard model. Especially in the random oracle model, we have to take into account the fact that an adversary can access the random oracle.

Intuitively, in the case of reducing the security of the NIZK proof system, when the adversary of the group signature scheme queries to the random oracle, the reduction

algorithm accesses his own random oracle, and returns the answer to the adversary. In the case of reducing the security of the other building blocks, the reduction algorithm simulates the random oracle by lazy sampling.

For completeness, we provide the definition of non-interactive zero-knowledge proof in the random oracle model. and proofs of the scheme.

Non-interactive Zero-knowledge Proof in the Random Oracle Model. Let R_L be an efficiently computable binary relation. For a pair $(x, w) \in R_L$, we call x a statement and w a witness. Let L be the language consisting of statements in R_L. A NIZK proof system $\mathcal{P}_L^{\mathsf{H}}$ for a language L consists of two algorithms $(\mathsf{P}^{\mathsf{H}}, \mathsf{V}^{\mathsf{H}})$ where H is a hash function modeled as the random oracle. The proof algorithm P^{H} takes 1^λ, a statement x, and a witness w, and then outputs a proof π. The verification algorithm V^{H} takes x and π, and outputs 1 if π is a valid proof for x and 0 otherwise. In the following, we give the definitions of completeness and soundness.

Definition 5.2.1 (Completeness [52])**.** *We say that $\mathcal{P}_L^{\mathsf{H}}$ satisfies completeness if for all $(x, w) \in R_L$ the equation*

$$\Pr[\ \mathsf{V}^{\mathsf{H}}(x, \pi) = 1 \mid\ \ \pi \leftarrow \mathsf{P}^{\mathsf{H}}(x, w)\ ] = 1$$

holds.

Definition 5.2.2 (Soundness [52])**.** *We say that $\mathcal{P}_L^{\mathsf{H}}$ satisfies soundness if the advantage*

$$\mathsf{Adv}^{sound}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda) = \Pr[\ x^* \notin L \wedge \mathsf{V}^{\mathsf{H}}(x^*, \pi^*) = 1 \mid (x^*, \pi^*) \leftarrow \mathcal{A}^{\mathsf{H}}(1^\lambda)\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

Also, we give the definition of zero-knowledge for NIZK proof systems in the random oracle model.

Definition 5.2.3 (Zero-knowledge in the Random Oracle Model [52])**.** *Let $\mathcal{A}$ be an adversary and $\mathcal{S} = (\mathtt{Sim}_1, \mathtt{Sim}_2)$ be a simulator for a non-interactive proof system $\mathcal{P}_L^{\mathsf{H}}$. We define the experiments $\mathsf{Exp}^{proof}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda)$ and $\mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda)$ as follows.*

$\mathsf{Exp}^{proof}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda)$:	$\mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda)$:
$b \leftarrow \mathcal{A}^{\mathsf{H}(\cdot),\mathsf{Prove}(\cdot,\cdot)}(1^\lambda)$	$b \leftarrow \mathcal{A}^{\mathsf{SimH}(\cdot),\mathsf{SimProve}(\cdot,\cdot)}(1^\lambda)$
Return b	Return b

In this experiment, the oracle Prove *takes (x, w), computes $\pi \leftarrow \mathsf{P}^{\mathsf{H}}(x, w)$, and returns π. The oracle* $\mathsf{SimProve}$ *takes (x, w), computes $\pi \leftarrow \mathtt{Sim}_2(x)$, and returns π. If*

$(x, w) \notin R_L$, *then* $\mathsf{SimProve}$ *returns* $\perp$. *The oracle* SimH *takes* x, *computes* $y \leftarrow \mathtt{Sim}_1(x)$, *and returns* y.

We say that $\mathcal{P}_L^{\mathsf{H}}$ *satisfies zero-knowledge in the random oracle model if there exists a simulator* $\mathcal{S} = (\mathtt{Sim}_1, \mathtt{Sim}_2)$ *such that for any PPT adversary* $\mathcal{A}$, *the advantage*

$$\mathsf{Adv}^{zk}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda) = \left| \Pr[\ \mathsf{Exp}^{proof}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda) = 1\ ] - \Pr[\ \mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{A}}(\lambda) = 1\ ] \right|$$

is negligible.

Here, we prove that our scheme satisfies selfless anonymity and traceability in the random oracle model.

Theorem 5.2.8. *Our scheme* Π_{GS} *is selfless anonymous in the random oracle model if the underlying NIZK proof system satisfies zero-knowledge in the random oracle model, the underlying commitment scheme is computational hiding, and the underlying SKE scheme satisfies ciphertext-pseudorandomness.*

Proof. Let $\mathcal{A}$ be an adversary that attacks selfless anonymity of our scheme Π_{GS}, and n be a polynomial in λ. We consider the following sequence of games. For the proof of the selfless anonymity in the standard model in Theorem 5.2.2, we modify only Game 1 as follows. The other games from Game 2 to Game 5 are defined as with the games in Theorem 5.2.2.

Game 1: Let $\mathcal{S} = (\mathsf{S}_1, \mathsf{S}_2)$ be a simulator of the NIZK proof system. In Game 1, the challenger answers the $\mathcal{A}$'s random oracle queries by using $\mathtt{Sim}_1$, and generates NIZK proofs by using $\mathtt{Sim}_2$ We note that in games after Game 2, the challenger answers the $\mathcal{A}$'s random oracle queries by using $\mathtt{Sim}_1$, and generates NIZK proofs by using $\mathtt{Sim}_2$ as with Game 1.

Let $\mathtt{Suc}_i$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit in Game i. For $\mathcal{A}$'s advantage $\mathsf{Adv}^{anon}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda)$, it holds that $\mathsf{Adv}^{anon}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda) = \Pr[\mathtt{Suc}_0] - 1/2 \leq \sum_{i=0}^{4} \left|\Pr[\mathtt{Suc}_i] - \Pr[\mathtt{Suc}_{i+1}]\right| + \left|\Pr[\mathtt{Suc}_5] - 1/2\right|$. We note that a reduction algorithm needs to simulate the random oracle for $\mathcal{A}$ in the random oracle model. In the case of constructing the reduction for the building blocks except for the NIZK proof system, it is easy to simulate the random oracle by the simulator $\mathtt{Sim}_1$. Therefore for $1 \leq i \leq 4$, we can easily say that $\left|\Pr[\mathtt{Suc}_i] - \Pr[\mathtt{Suc}_{i+1}]\right|$ is negligible as with Lemma 8.3.2 to Lemma 8.3.5. On the other hand in the case of constructing the reduction for the NIZK proof system, it is non-trivial. Thus, we prove that $\left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right|$ is negligible in the following lemma.

Lemma 5.2.13. *There exists a PPT algorithm* $\mathcal{B}_1$ *such that* $\left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right| = \mathsf{Adv}^{zk}_{\mathcal{P}_L^{\mathsf{H}},\mathcal{B}_1}(\lambda)$.

Proof. Let $\mathcal{B}_1$ be an adversary who attacks the zero-knowledge of the underlying NIZK proof system. $\mathcal{B}_1$ generates the whole instance of the selfless anonymity game which is give to $\mathcal{A}$. When $\mathcal{B}_1$ obtains a hash query q_i from $\mathcal{A}$, $\mathcal{B}_1$ queries the value q_i to his own hash oracle (which is either $\mathsf{H}(\cdot)$ or $\mathsf{SimH}(\cdot)$) and returns the value h_i which he gets. When $\mathcal{A}$ queries a message m to the oracle Sign_B ($B \in \{0,1\}$), the algorithm $\mathcal{B}_1$ operates by using the key $\mathsf{gsk}_{i_B} = (i_B, \mathsf{vk}_{i_B}, \mathsf{sk}_{i_B}, k_{i_B}, r_{i_B})$ which he possesses as follows.

1. Generate a signature $s \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_B}, m)$.
2. Sample a randomness R and compute $C \leftarrow \mathsf{SKE.Enc}(k_{i_B}, \langle i_B, \mathsf{vk}_{i_B}, s\rangle; R)$.
3. Let $x = \langle m, C, T, \mathsf{ck}\rangle$ and $w = \langle i_B, \mathsf{vk}_{i_B}, s, R, k_{i_B}, r_{i_B}\rangle$. Query a pair (x, w) to his proof oracle (which is either $\mathsf{Prove}(\cdot,\cdot)$ or $\mathsf{SimProve}(\cdot,\cdot)$) and obtain a proof π.
4. Send $\sigma = (C, \pi)$ to $\mathcal{A}$ as the response.

For the challenge query m^* from $\mathcal{A}$, $\mathcal{B}_1$ samples a bit β uniformly at random and generates a group signature $\sigma^* = (C^*, \pi^*)$ for the message m^* where the proof π^* is obtained by accessing the oracle as with signing queries. When $\mathcal{A}$ terminates with output β', $\mathcal{B}_1$ outputs $b' = 1$ if $\beta' = \beta$, otherwise outputs $b' = 0$.

If the oracles that $\mathcal{B}_1$ accesses are $\mathsf{H}(\cdot)$ and $\mathsf{Prove}(\cdot,\cdot)$, $\mathcal{B}_1$ perfectly simulates Game 0 for $\mathcal{A}$. On the other hand, if the oracles are $\mathsf{SimH}(\cdot)$ and $\mathsf{SimProve}(\cdot,\cdot)$, $\mathcal{B}_1$ perfectly simulates Game 1 for $\mathcal{A}$. Therefore, it holds that $\mathsf{Adv}^{zk}_{\mathcal{P}^{\mathsf{H}}_L,\mathcal{B}_1}(\lambda) = \big|\Pr[\mathsf{Exp}^{proof}_{\mathcal{P}^{\mathsf{H}}_L,\mathcal{B}_1}(\lambda) = 1] - \Pr[\mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}^{\mathsf{H}}_L,\mathcal{B}_1}(\lambda) = 1]\big| = \big|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\big|$. □

As well as the proof of Theorem 5.2.2, we can say that $\Pr[\mathtt{Suc}_5] = 1/2$. Therefore, $\mathsf{Adv}^{anon}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda, n)$ is negligible. Since the choice of the parameter n and the adversary $\mathcal{A}$ is arbitrarily, our scheme Π_{GS} is selfless anonymous in the random oracle model. □

Theorem 5.2.9. *Our scheme Π_{GS} is traceable in the random oracle model if the underlying NIZK proof system satisfies soundness in the random oracle model, the underlying commitment scheme is statistical binding, the underlying SKE scheme satisfies key-robustness, and the underlying signature scheme satisfies EUF-CMA security.*

Proof. Let $\mathcal{A}$ be an adversary for the traceability of the scheme Π_{GS}, and n a polynomial in λ. We classify $\mathcal{A}$'s forgery $(m^*, (C^*, \pi^*))$ into five types as with the proof of Theorem 5.2.3. Let E_j denote the event that $\mathcal{A}$ outputs a forgery which is in Type j. Then, it holds that $\mathsf{Adv}^{\mathrm{trace}}_{\Pi_{\mathrm{GS}},\mathcal{A}^{\mathsf{H}}}(\lambda, n) = \sum_{j=1}^{5} \Pr[\mathsf{E}_j]$. For the parts of constructing the reduction for the building blocks except for the NIZK proof system, it is easy to prove the security in the random oracle model by simulating the random oracle by lazy sampling. Therefore for $2 \le i \le 5$, we can easily say that $\Pr[\mathsf{E}_j]$ is negligible as with the proofs in the standard model. However, for the part of constructing the reduction for the NIZK proof system, it is non-trivial. Thus, we prove the following lemma for the probability $\Pr[\mathsf{E}_1]$.

Lemma 5.2.14. *There exists a PPT algorithm* $\mathcal{B}_1$ *such that* $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}^{\mathsf{H}}_L,\mathcal{B}_1}(\lambda)$.

Proof. Let $\mathcal{B}_1$ be an adversary that attacks the soundness of the NIZK proof system. $\mathcal{B}_1$ generates the whole instance of the selfless anonymity game which is given to $\mathcal{A}$. Since $\mathcal{B}_1$ has all the user secret keys, it is easy to simulate the Sign oracle and the $\mathsf{Corrupt}$ oracle. When $\mathcal{A}$ queries a hash query q_i, $\mathcal{B}_1$ queries the value q_i to his own hash oracle $\mathsf{H}(\cdot)$ and returns the value h_i which he gets. When $\mathcal{A}$ terminates with an output $(m^*, (C^*, \pi^*))$, $\mathcal{B}_1$ outputs $(x^*, \pi^*) = (\langle m^*, C^*, T, \mathsf{ck}\rangle, \pi^*)$ as a forgery in the soundness game.

If the event E_1 occurs, it holds that $x^* \notin L$ and $\mathsf{V}^{\mathsf{H}}(\langle m^*, C^*, T\rangle, \pi^*) = 1$ from the winning condition of $\mathcal{A}$ and the condition of Type 1. Therefore, (x^*, π^*) is a forgery for the soundness of the NIZK proof system, and $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}^{\mathsf{H}}_L,\mathcal{B}_1}(\lambda)$. □

Therefore, $\mathsf{Adv}^{\mathrm{trace}}_{\Pi_{\mathrm{GS}},\mathcal{A}}(\lambda, n) = \sum_{j=1}^{5} \Pr[\mathsf{E}_j]$ is negligible since the choice of the parameter n and the adversary $\mathcal{A}$ is arbitrarily. Then our scheme Π_{GS} satisfies traceability in the random oracle model. □

5.3 The Gap of Full Anonymity and Selfless Anonymity

In this section, we consider the gap between fully anonymous group signature and selfless anonymous group signature from the practical and theoretical aspects.

The Practical Gap. In the model of full anonymity, an adversary is allowed to obtain all the signing keys whereas in the model of selfless anonymity, an adversary is not allowed to obtain signing keys of the challenge users. Therefore, full anonymity is a truly stronger security notion than selfless anonymity. However, there seems to be no significant difference between them in practical use.

First, in a selfless anonymous group signature scheme, a user might be capable of breaking the anonymity of signatures that he generated, while in a fully anonymous group signature scheme, even a user himself cannot break the anonymity. However, from a practical point of view, a user can recognize whether the signature was generated by himself if he records the signatures that he generated even in a fully anonymous group signature scheme.

Second, the signing key issuer might break the anonymity of signatures since selfless anonymity cannot ensure the anonymity for an adversary who possesses all the users' signing keys. However, in the BMW model [14], which is widely recognized as a common model of group signatures, the key issuer also generates an opening key. Therefore, anonymity is not ensured against the key issuer in the BMW model at all any way.

From these facts, a selfless anonymous group signature scheme performs sufficient functionality in practice even though selfless anonymity is a weaker security notion than full anonymity. In fact, several efficient group signature schemes [78, 83, 87, 88] employ selfless anonymity.

The Theoretical Gap. On the other hand, it is seen that there is a big theoretical gap between selfless anonymous group signature and fully anonymous group signature. In particular, a selfless anonymous group signature scheme can be constructed from a one-way function and a NIZK proof system as the result in Section 5.2.2 and 5.2.3. In contrast, several results [6, 95, 49] suggest that a PKE scheme is an inevitable building block for constructing a fully anonymous group signature scheme. Therefore, it seems that the gap between selfless anonymous group signature and fully anonymous group signature is the same as that between one-way function and PKE.

Chapter 6

Group Signature with Verifier Local Revocation

In this chapter, we focus on group signature with verifier-local revocation (VLR-GS) [28], which is one of approaches for realizing a revocation functionality in the context of group signature. In a VLR-GS scheme, verifiers need to download the up-to-date information of the revoked users to verify signatures but signers are not required to do so. Thus, VLR-GS schemes are very useful for the situations that users are difficult to download the up-to-date information whenever signing a message.

The notion of VLR-GS was proposed by Boneh and Shacham [28], and Nakanishi and Funabiki [87] extended its security notion by considering backward unlinkability. After the first scheme is given by Boneh and Shacham in 2004, there have been several proposals of VLR-GS schemes [28, 87, 88, 106, 83, 78]. For example, the Libert-Vergnaud scheme [83] is the first scheme secure in the standard model, and the Langlois-Ling-Nguyen-Wang scheme [78] is the first lattice-based scheme. However, unfortunately, all of these schemes only achieve selfless-anonymity. Thus, it has been an open problem for more than a decade whether a fully anonymous VLR-GS scheme can be achieved.

Here, we give an affirmative answer to this problem. Concretely, we show a construction of a fully anonymous VLR-GS scheme from a digital signature scheme, a key-private public key encryption scheme, and a non-interactive zero- knowledge proof system. Also, we give a VLR-GS scheme with backward unlinkability, which ensures that even after a user is revoked, signatures produced by the user before the revocation remain anonymous.

6.1 Syntax

In this section, we review the syntax of VLR-GS. Especially, here, we provide the model proposed by Nakanishi and Funabiki [87]. In this model, the additional security notion

called backward unlinkability, is considered by introducing time periods. Intuitively, backward unlinkability ensures that even after a user is revoked, signatures produced by the user before the revocation remain anonymous. We note that the model given by Boneh and Shacham [28] is a special case of the Nakanishi-Funabiki model where the number of time periods is only one.

Intuitively, a VLR-GS scheme operates as follows: a token (called a revocation token) is defined for each user, and the authority reveals it in the public list (called a revocation list) if the corresponding user is revoked. That is, the revocation list contains the revocation tokens of the revoked users. A revocation token can be used to detect the signature generated by the corresponding user. Thus, a verifier can check whether the signer is revoked by using the revocation list. On the other hand, a signer can generate signatures only by using his signing key, that is, he does not need to refer the revocation list.

A VLR-GS scheme $\mathcal{VLR}$-$\mathcal{GS}$ consists of the following three algorithms ($\mathsf{VLRGS.Gen}$, $\mathsf{VLRGS.Sign}, \mathsf{VLRGS.Verify}$).

$\mathsf{VLRGS.Gen}$: The key generation algorithm takes a security parameter 1^λ ($\lambda \in \mathbb{N}$), the number of users n, and the number of time periods T, and outputs a group public key gpk, a set of user signing keys $\mathbf{gsk} = \{\mathsf{gsk}[i]\}_i$, and a set of revocation tokens $\mathbf{grt} = \{\mathsf{grt}[i][j]\}_{ij}$. Here, $\mathsf{gsk}[i]$ and $\mathsf{grt}[i][j]$ denote the user i's signing key and revocation token at the time period j, respectively.

$\mathsf{VLRGS.Sign}$: The signing algorithm takes gpk, time period j, $\mathsf{gsk}[i]$, and a message msg, and outputs a signature Σ.

$\mathsf{VLRGS.Verify}$: The verification algorithm takes gpk, j, a revocation list RL_j, msg, and Σ, and outputs either 1 or 0. The list RL_j is defined as the set of the revocation tokens $\mathsf{RL}_j = \{\mathsf{grt}[i][j] \mid i \in \mathsf{RU}_j\}$ where RU_j is the set of the revoked users' identities at the time period j.

In a VLR-GS scheme, the opening procedure can be done by using a set of revocation tokens $\mathbf{grt} = \{\mathsf{grt}[i][j]\}_{ij}$. More precisely, the implicit opening algorithm $\mathsf{VLRGS.Open}$ can be defined as follows.

$\mathsf{VLRGS.Open}$: The opening algorithm takes gpk, j, a set of revocation tokens $\mathbf{grt}$, msg, and Σ, and executes the following procedures:

[Step 1] For $1 \leq i \leq n$, set the revocation list $\mathsf{RL}_j = \{\mathsf{grt}[i][j]\}$, and then run $\mathsf{VLRGS.Verify}(\mathsf{gpk}, j, \mathsf{RL}_j, \mathsf{msg}, \Sigma)$.

[Step 2] Let i be the index that the $\mathsf{VLRGS.Verify}$ algorithm outputs 0 for the first time in Step 1. Then, output this index i. If there does not exist such an index, output $\perp$.

6.2 Security

Here, we give the definition of the security requirements for VLR-GS schemes, correctness, anonymity, and traceability. Specifically, we give two definitions of anonymity, selfless anonymity [87] and full anonymity [71].

Firstly, we give the definition of correctness. Here, we introduce the game-based definition [71].

Definition 6.2.1 (Correctness)**.** *Let $\mathcal{A}$ be an adversary for the correctness. We define the experiment $\mathsf{Exp}^{corr}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T)$ as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{corr}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T): \quad & (\mathsf{gpk}, \mathbf{gsk}, \mathbf{grt}) \leftarrow \mathsf{VLRGS.Gen}(1^\lambda, n, T) \\
& (i, j, \mathsf{msg}, \mathsf{RU}) \leftarrow \mathcal{A}(\mathsf{gpk}) \\
& \text{If } i \in \mathsf{RU}, \text{ return } 0 \\
& \mathsf{RL}_j := \{\mathsf{grt}[i][j] \mid i \in \mathsf{RU}\} \\
& \Sigma \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, j, \mathsf{gsk}[i], \mathsf{msg}) \\
& \text{Return 1 if } \mathsf{VLRGS.Verify}(\mathsf{gpk}, j, \mathsf{RL}_j, \mathsf{msg}, \Sigma) = 0 \\
& \text{else return } 0
\end{aligned}
$$

We say that $\mathcal{VLR}$-$\mathcal{GS}$ is correct if the advantage

$$
\mathsf{Adv}^{corr}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T) = \Pr[\ \mathsf{Exp}^{corr}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T) = 1\ ]
$$

is negligible for any PPT adversary $\mathcal{A}$.

Next, we give the definition of selfless anonymity. Intuitively, selfless anonymity ensures the anonymity of a signature against only the adversary who does not possess the user signing key which is used in the generation of the corresponding signature.

Definition 6.2.2 (Selfless Anonymity)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for selfless anonymity. We define the experiment $\mathsf{Exp}^{self\text{-}anon}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T)$ as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{self\text{-}anon}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T): \quad & \mathsf{CU} \leftarrow \emptyset;\ \mathsf{RU}_j \leftarrow \emptyset \\
& (\mathsf{gpk}, \mathbf{gsk}, \mathbf{grt}) \leftarrow \mathsf{VLRGS.Gen}(1^\lambda, n) \\
& (\mathsf{st}, i_0, i_1, j^*, \mathsf{msg}^*) \leftarrow \mathcal{A}_1^{\mathsf{Sign}(\cdot,\cdot,\cdot),\mathsf{Revoke}(\cdot,\cdot),\mathsf{Corrupt}(\cdot)}(\mathsf{gpk}) \\
& \text{If } i_0 \in \mathsf{RU}_{j^*} \vee i_1 \in \mathsf{RU}_{j^*}, \text{ return } 0 \\
& \text{If } i_0 \in \mathsf{CU} \vee i_1 \in \mathsf{CU}, \text{ return } 0 \\
& b \xleftarrow{\$} \{0,1\};\ \Sigma^* \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, j^*, \mathsf{gsk}[i_b], \mathsf{msg}^*) \\
& \widetilde{b} \leftarrow \mathcal{A}_2^{\mathsf{Sign}(\cdot,\cdot,\cdot),\mathsf{Revoke}(\cdot,\cdot),\mathsf{Corrupt}(\cdot)}(\mathsf{st}, \Sigma^*)
\end{aligned}
$$

$$\text{Return 1 if } b = \widetilde{b}, \text{ else return 0}$$

In this experiment, the oracle Sign *takes* (i, j, msg)*, computes* $\Sigma \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, j, \mathsf{gsk}[i], \mathsf{msg})$*, adds* (i, msg) *and returns* Σ*. The oracle* Revoke *takes* $i \in [1, n]$ *and* $j \in [1, T]$*, adds* i *to the list* RU_j*, and returns* $\mathsf{grt}[i][j]$*. We note that it is not allowed to query* (i_0, j^*) *and* (i_1, j^*) *to the* Revoke *oracle. The oracle* $\mathsf{Corrupt}$ *takes* $i \in [1, n]$*, adds* i *to the list* CU*, and returns* $\mathsf{gsk}[i]$*.*

We say that $\mathcal{VLR}$*-*$\mathcal{GS}$ *satisfies selfless anonymity if the advantage*

$$\mathsf{Adv}^{\textit{self-anon}}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T) = \left| \ \Pr[\ \mathsf{Exp}^{\textit{self-anon}}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T) = 1\ ] - 1/2 \ \right|$$

is negligible for any polynomial $n = n(\lambda)$ *and* $T = T(\lambda)$*, and any PPT adversary* $\mathcal{A}$*.*

Full anonymity is a stronger security notion than selfless anonymity and ensures that the signer's information cannot be extracted from a signature by the adversary with all the user signing keys. In the following, we review the formall definition of full anonymity.

Definition 6.2.3 (Full Anonymity)**.** *Let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be an adversary for full anonymity. We define the experiment* $\mathsf{Exp}^{\textit{full-anon}}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T)$ *as follows.*

$$\begin{aligned}
\mathsf{Exp}^{\textit{full-anon}}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T): \quad & \mathsf{RU}_j \leftarrow \emptyset;\ (\mathsf{gpk}, \mathbf{gsk}, \mathbf{grt}) \leftarrow \mathsf{VLRGS.Gen}(1^\lambda, n) \\
& (\mathsf{st}, i_0, i_1, j^*, \mathsf{msg}^*) \leftarrow \mathcal{A}_1^{\mathsf{Revoke}(\cdot,\cdot)}(\mathsf{gpk}, \mathbf{gsk}) \\
& \text{If } i_0 \in \mathsf{RU}_{j^*} \vee i_1 \in \mathsf{RU}_{j^*}, \text{ return } 0 \\
& b \xleftarrow{\$} \{0,1\};\ \Sigma^* \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, j^*, \mathsf{gsk}[i_b], \mathsf{msg}^*) \\
& \widetilde{b} \leftarrow \mathcal{A}_2^{\mathsf{Revoke}(\cdot,\cdot)}(\mathsf{st}, \Sigma^*) \\
& \text{Return 1 if } b = \widetilde{b}, \text{ else return 0}
\end{aligned}$$

In this experiment, the oracle Revoke *takes* $i \in [1, n]$ *and* $j \in [1, T]$*, adds* i *to the list* RU_j*, and returns* $\mathsf{grt}[i][j]$*. We note that it is not allowed to query* (i_0, j^*) *and* (i_1, j^*) *to the* Revoke *oracle.*

We say that $\mathcal{VLR}$*-*$\mathcal{GS}$ *satisfies full anonymity if the advantage*

$$\mathsf{Adv}^{\textit{full-anon}}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T) = \left| \ \Pr[\ \mathsf{Exp}^{\textit{full-anon}}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T) = 1\ ] - 1/2 \ \right|$$

is negligible for any polynomial $n = n(\lambda)$ *and* $T = T(\lambda)$*, and any PPT adversary* $\mathcal{A}$*.*

Finally, we give the definition of traceability.

Definition 6.2.4 (Traceability)**.** *Let* $\mathcal{A}$ *be an adversary for the traceability. We define*

the experiment $\mathsf{Exp}^{trace}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T)$ *as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{trace}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n, T): \quad & \mathsf{CU} \leftarrow \emptyset;\ \mathsf{QL} \leftarrow \emptyset \\
& (\mathsf{gpk}, \mathbf{gsk}, \mathbf{grt}) \leftarrow \mathsf{VLRGS.Gen}(1^\lambda, n, T) \\
& (j^*, \mathsf{msg}^*, \Sigma^*, \mathsf{RU}^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\cdot,\cdot,\cdot),\mathsf{Corrupt}(\cdot)}(\mathsf{gpk}, \mathbf{grt}) \\
& \mathsf{RL}^* := \{\mathsf{grt}[i][j^*] \mid i \in \mathsf{RU}^*\} \\
& i^* \leftarrow \mathsf{VLRGS.Open}(\mathsf{gpk}, j^*, \mathbf{grt}, \mathsf{msg}^*, \Sigma^*) \\
& \text{Return 1 if the following holds, else return 0} \\
& \quad \mathsf{VLRGS.Verify}(\mathsf{gpk}, j^*, \mathsf{RL}^*, \mathsf{msg}^*, \Sigma^*) = 1 \\
& \quad i^* = \perp \ \vee \ i^* \notin \mathsf{CU} \ \vee \ i^* \in \mathsf{RU}^* \\
& \quad (\cdot, \cdot, \mathsf{msg}^*, \Sigma^*) \notin \mathsf{QL}
\end{aligned}
$$

In this experiment, the oracle Sign *takes* (i, j, msg)*, computes* $\Sigma \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, j, \mathsf{gsk}[i], \mathsf{msg})$*, adds* $(i, j, \mathsf{msg}, \Sigma)$ *to the list* QL*, and returns* Σ*. The oracle* $\mathsf{Corrupt}$ *takes* $i \in [1, n]$*, adds* i *to the list* CU*, and returns* $\mathsf{gsk}[i]$*. We say that* $\mathcal{VLR}\text{-}\mathcal{GS}$ *satisfies traceability if the advantage*

$$\mathsf{Adv}^{trace}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n) = \Pr[\ \mathsf{Exp}^{trace}_{\mathcal{VLR}\text{-}\mathcal{GS},\mathcal{A}}(\lambda, n) = 1\ ]$$

is negligible for any polynomial $n = n(\lambda)$ *and* $T = T(\lambda)$*, and any PPT adversary* $\mathcal{A}$*.*

6.3 Fully anonymous Schemes

In this section, we introduce the constructions of fully anonymous VLR-GS scheme [71]. Before the work [71], all the VLR-GS schemes are only proved to satisfy selfless anonymity whereas the standard revocable group signature schemes which satisfy full anonymity are proposed so far (e.g., the schemes proposed by Libert, Peters, and Yung [80, 81] provide full anonymity). Specifically, there are trivial attacks against the full anonymity of all the existing schemes with only one exception. Fully anonymous VLR-GS schemes can be achieved by providing a different approach from the previous schemes [28, 87, 88, 106, 83, 78].

In Section 6.3.1, we give a construction of a fully anonymous VLR-GS scheme without backward unlinkability from a digital signature scheme, a PKE scheme, and a NIZK proof system. The building blocks are essentially the same as those of a standard group signature scheme [14], however, in the construction of a VLR-GS scheme, the underlying PKE scheme is additionally required to satisfy key privacy which is an essential to ensure that the VLR-GS scheme is fully anonymous.

In Section 6.3.2, we give two constructions of a fully anonymous VLR-GS scheme with backward unlinkability. The first scheme is constructed from the same building blocks as those of the scheme without backward unlinkability. However, the size of the user signing key depends on the number of time periods. In the second scheme, a key-private IBE scheme is employed as an additional building block to achieve the constant key size.

6.3.1 Scheme without Backward Unlinkability

In this section, we give a construction of a fully anonymous VLR-GS scheme without backward unlinkability, that is, in the case that the number of time periods satisfies $T = 1$. Concretely, we construct the VLR-GS scheme from a digital signature scheme, a key-private PKE scheme, and a NIZK proof system. Here, there is only one time period $j = 1$, and then we do not specify the time period and omit it.

As mentioned, all the previous schemes [28, 87, 88, 106, 83, 78] only provide selfless anonymity regardless of with or without backward unlinkability. Specifically, there is an attack against the full anonymity for most of the schemes [28, 87, 88, 106, 78] due to their structures that the revocation token for a user can be constructed from the user's signing key. Recall that the revocation token can be used to detect the signature generated by the corresponding user. Therefore, if the revocation token is constructed from the corresponding signing key, the adversary with all the user signing keys can identify the signer from any signature by computing all the users' revocation tokens. Thus, a VLR-GS scheme with such a structure never satisfy full anonymity. Therefore, in order to achieve full anonymity, a VLR-GS scheme needs not to have such a structure while the revocation token and signing key of the same user have some relation.

Intuitively, we achieve this by employing an encryption/decryption key pair of a PKE scheme as a part of user signing key and a revocation token. In the following, we explain the detail of the scheme which we call Scheme 1. Before describing the construction, we give the high level idea of this scheme.

High Level Idea. Scheme 1 mainly follows the BMW construction [14], which allows us to construct a fully anonymous group signature scheme from a digital signature scheme, a PKE scheme, and a NIZK proof system.

In the BMW construction introduced in Section 3.3, the group manager possesses a key pair $(\mathsf{vk}_{\mathsf{SIG}}, \mathsf{sk}_{\mathsf{SIG}})$ of a digital signature scheme and a key pair $(\mathsf{ek}_{\mathsf{PKE}}, \mathsf{dk}_{\mathsf{PKE}})$ of a PKE scheme. Each user possesses a key pair $(\mathsf{vk}_i, \mathsf{sk}_i)$ of a digital signature scheme and its certificate cert_i given by the manager where cert_i is the signature of the verification key vk_i under the signing key $\mathsf{sk}_{\mathsf{SIG}}$. When a user i signs a message m, the user generates an internal signature σ on the message m using his signing key sk_i, and encrypts σ using

$\mathsf{ek}_{\mathsf{PKE}}$ along with the verification key vk_i and the corresponding certificate cert_i. Let ct be this ciphertext. Also, the user produces a NIZK proof π which proves that the whole procedure is honestly done and the encrypted certificate cert_i is a valid signature on vk_i. That is, the signature Σ in the BMW construction consists of a ciphertext ct and a proof π. The full anonymity is ensured by the IND-CPA security of the underlying PKE scheme and the zero-knowledgeness of the underlying NIZK proof system. The traceability is ensured by the EUF-CMA security of the underlying digital scheme and the soundness of the underlying NIZK proof system.

In the construction of a VLR-GS scheme, we add a revocation functionality by introducing additional key pairs of a key-private PKE scheme to the BMW construction. The detail is as follows. The manager generates an encryption/decryption key pair $(\mathsf{ek}_i, \mathsf{dk}_i)$ for each user i and sends only the encryption key ek_i as a part of the signing key to the user. Also, the manager sets the decryption key dk_i as the revocation token of the user i. To certify that the key ek_i is generated for a user i by the manager, he also computes a signature cert_i on the message $\langle \mathsf{ek}_i, \mathsf{vk}_i \rangle$ under the signing key $\mathsf{sk}_{\mathsf{SIG}}$ as a certificate. As with the BMW construction, when signing a message m, a user i generates an internal signature σ on the message m using the signing key sk_i, and encrypts σ, $\langle \mathsf{ek}_i, \mathsf{vk}_i \rangle$, and cert_i under $\mathsf{ek}_{\mathsf{PKE}}$. Also, in the construction, the signer i generates a ciphertext $\widetilde{\mathsf{ct}}$ which is the encryption of the same plaintext $\langle \sigma, \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i \rangle$ as the ciphertext ct under the encryption key ek_i.[1] Then, the user produces a NIZK proof π which proves that the whole procedure is honestly done and cert_i is a valid signature on $\langle \mathsf{ek}_i, \mathsf{vk}_i \rangle$ as the case of the BMW construction. That is, the signature Σ in Scheme 1 consists of ciphertexts ct and $\widetilde{\mathsf{ct}}$, and a proof π.

Scheme 1 does not have the structure that the revocation token can be computed from the corresponding signing key since it is hard to compute the decryption key dk_i even if knowing the corresponding encryption key ek_i from the security of the underlying PKE scheme. The decryption key dk_i works as the revocation token as follows. If a user i is revoked, his revocation token $\mathsf{grt}[i] = \mathsf{dk}_i$ is listed in the revocation list RL. If a verifier check whether the ciphertext $\widetilde{\mathsf{ct}}$ can be decrypted by each element in RL as the decryption key, he can check whether the signer is a revoked user.

The security of Scheme 1 can be discussed in almost the same way as the BMW construction. However, the underlying PKE scheme is required to be key-private in our construction since the ciphertext $\widetilde{\mathsf{ct}}$ is computed by the encryption key ek_i depending on

[1]The reader might think that the ciphertext ct is redundant and it is enough that the ciphertext ct is replaced with the ciphertext $\widetilde{\mathsf{ct}}$. However, if so, it is difficult to reduce its traceability to the EUF-CMA security of the underlying digital signature scheme. More precisely, if the adversary uses an uncertified encryption key to generate $\widetilde{\mathsf{ct}}$, the reduction algorithm cannot extract the forgery of the digital signature scheme. Also in fact, we note that it is not necessary to encrypt the whole value $\langle \sigma, \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i \rangle$ in both ct and $\widetilde{\mathsf{ct}}$. Therefore, the part of the value is encrypted in the ciphertexts.

the signer i. The full anonymity is ensured by the IND-CPA security and the key privacy of the underlying PKE scheme, and the zero-knowledgeness of the underlying NIZK proof system. The traceability is ensured by the EUF-CMA security of the underlying digital scheme and the soundness of the underlying NIZK proof system.

Description. Scheme 1 is given in Figure 6.1. We construct a VLR-GS scheme $\Pi_1 = (\mathsf{VLRGS.Gen}, \mathsf{VLRGS.Sign}, \mathsf{VLRGS.Verify})$ from a digital signature scheme $\mathcal{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify})$, a PKE scheme $\mathcal{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$, and a NIZK proof system $\mathcal{P}_L = (\mathsf{ZK.Gen}, \mathsf{ZK.Prove}, \mathsf{ZK.Verify})$. We say that a statement $x = \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle$ and a witness $w = \langle \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i, \sigma, r_1, r_2 \rangle$ satisfy the relation R_L if the following equations hold:

$$\text{(a) } \widetilde{\mathsf{ct}} = \mathsf{PKE.Enc}(\mathsf{ek}_i, \sigma; r_1), \quad \text{(b) } \mathsf{ct} = \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i \rangle; r_2),$$
$$\text{(c) } \mathsf{SIG.Verify}(\mathsf{vk}_{\mathsf{SIG}}, \langle \mathsf{ek}_i, \mathsf{vk}_i \rangle, \mathsf{cert}_i) = 1, \quad \text{(d) } \mathsf{SIG.Verify}(\mathsf{vk}_i, m, \sigma) = 1.$$

Moreover, for a statement $x = \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle$, if there exists a witness that satisfies the above equations, then we say that the statement x belongs to the language L and denote it $x \in L$.

$\mathsf{VLRGS.Gen}(1^\lambda, n)$:
  $\mathsf{crs} \leftarrow \mathsf{ZK.Gen}(1^\lambda)$; $(\mathsf{vk}_{\mathsf{SIG}}, \mathsf{sk}_{\mathsf{SIG}}) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$
  $(\mathsf{ek}_{\mathsf{PKE}}, \mathsf{dk}_{\mathsf{PKE}}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$
  For $1 \le i \le n$:
    $(\mathsf{ek}_i, \mathsf{dk}_i) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$; $(\mathsf{vk}_i, \mathsf{sk}_i) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$
    $\mathsf{cert}_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{\mathsf{SIG}}, \langle \mathsf{ek}_i, \mathsf{vk}_i \rangle)$; $\mathsf{grt}[i] \leftarrow (\mathsf{dk}_i, \mathsf{vk}_i)$
  $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$; $\mathsf{gsk}[i] = (\mathsf{ek}_i, \mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$
  $\mathbf{gsk} = \{\mathsf{gsk}[i]\}_i$; $\mathbf{grt} = \{\mathsf{grt}[i]\}_i$
  Return $(\mathsf{gpk}, \mathbf{gsk}, \mathbf{grt})$

$\mathsf{VLRGS.Sign}(\mathsf{gpk}, \mathsf{gsk}[i], m)$:
  $\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_i, m)$
  $\widetilde{\mathsf{ct}} \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_i, \sigma; r_1)$
  $\mathsf{ct} \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i \rangle; r_2)$
  $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle, \langle \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i, \sigma, r_1, r_2 \rangle)$
  Return $\Sigma = (\widetilde{\mathsf{ct}}, \mathsf{ct}, \pi)$

$\mathsf{VLRGS.Verify}(\mathsf{gpk}, \mathsf{RL}, m, \Sigma)$:
  If $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle, \pi) = 0$, return 0
  For $(\mathsf{dk}, \mathsf{vk}) \in \mathsf{RL}$:
    If $\mathsf{SIG.Verify}(\mathsf{vk}, m, \mathsf{PKE.Dec}(\mathsf{dk}, \widetilde{\mathsf{ct}})) = 1$, return 0
  Return 1

Figure 6.1: **Scheme 1.** A VLR-GS Scheme without Backward Unlinkability

For the correctness of Scheme 1, the following theorem holds.

Theorem 6.3.1. *Scheme 1 is correct if the underlying NIZK proof system $\mathcal{P}_L$ satisfies completeness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security.*

Proof. Let $\mathcal{A}$ be an adversary for the correctness of Π_1 and the output of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{corr}_{\Pi_1,\mathcal{A}}(\lambda, n)$ be $(i^*, j^*, m^*, \mathsf{RU}^*)$. We note that now the number of time periods satisfies $T = 1$, then it holds that $j^* = 1$. Therefore, we do not specify the time period j^* as in the description of Scheme 1. If the experiment $\mathsf{Exp}^{corr}_{\Pi_1,\mathcal{A}}(\lambda, n)$ outputs 1, $\mathsf{VLRGS.Verify}(\mathsf{gpk}, \mathsf{RL}, m^*, \Sigma^*) = 0$ and $i^* \notin \mathsf{RU}^*$ hold where $\mathsf{RL} = \{grt[i] \mid i \in \mathsf{RU}^*\}$ and $\Sigma^* \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, \mathsf{gsk}[i^*], m^*)$. Let $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. From the definition of the $\mathsf{VLRGS.Verify}$ algorithm, either the event E_A or the event E_B happens when $\mathsf{VLRGS.Verify}(\mathsf{gpk}, \mathsf{RL}, m^*, \Sigma^*) = 0$ holds.

E_A**:** $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 0$ holds.

E_B**:** For some $i \in \mathsf{RU}^*$, $\mathsf{SIG.Verify}(\mathsf{vk}_i, m^*, \mathsf{PKE.Dec}(\mathsf{dk}_i, \widetilde{\mathsf{ct}}^*)) = 1$ holds.

However, $\Pr[\mathsf{E}_A] = 0$ holds if $\mathcal{P}_L$ satisfies completeness. Therefore, it holds that $\Pr[\mathsf{Exp}^{corr}_{\Pi_1,\mathcal{A}}(\lambda, n) = 1] = \Pr[\mathsf{E}_A \vee \mathsf{E}_B] \leq \Pr[\mathsf{E}_A] + \Pr[\mathsf{E}_B] = \Pr[\mathsf{E}_B]$.

We evaluate $\Pr[\mathsf{E}_B]$ by constructing an algorithm $\mathcal{B}$ that breaks the EUF-CMA security of the digital signature scheme $\mathcal{SIG}$. At the beginning of the game, $\mathcal{B}$ randomly chooses $\widehat{i} \in [1, n]$, and sets $\mathsf{vk}_{\widehat{i}} \leftarrow \mathsf{vk}$ where vk is the key given by the challenger of the EUF-CMA security game. $\mathcal{B}$ generates the rest of instance for the scheme Π_1 and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ to $\mathcal{A}$. For the $\mathcal{A}$'s output $(i^*, m^*, \mathsf{RU}^*)$, $\mathcal{B}$ outputs $\perp$ if $\widehat{i} = i^*$. Otherwise, if $\widehat{i} \neq i^*$, $\mathcal{B}$ computes $\Sigma^* \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, \mathsf{gsk}[i^*], m^*)$. Then, $\mathcal{B}$ computes $\sigma^* \leftarrow \mathsf{PKE.Dec}(\mathsf{dk}_{i^*}, \widetilde{\mathsf{ct}}^*)$, and outputs (m^*, σ^*) as a forged signature.

When the event E_B happens, there exists at least one pair $(\mathsf{dk}_i, \mathsf{vk}_i) \in \mathsf{RL}$ such that $\mathsf{SIG.Verify}(\mathsf{vk}_i, m^*, \mathsf{PKE.Dec}(\mathsf{dk}_i, \widetilde{\mathsf{ct}})) = 1$ holds. Let I be the set of such indexes i and $\mathtt{Good}$ be the event that $i^* \in I$ holds where i^* is the index chosen by $\mathcal{B}$ at the beginning of the game. Since the guess of $i^* \in [1, n]$ and the behavior of $\mathcal{A}$ are independent, we get $\Pr[\mathsf{E}_B \wedge \mathtt{Good}] = \Pr[\mathsf{E}_B] \cdot \Pr[\mathtt{Good}]$. When both events E_B and $\mathtt{Good}$ happen, it holds that $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \sigma^*) = 1$ where $\sigma^* \leftarrow \mathsf{PKE.Dec}(\mathsf{dk}_{i^*}, \widetilde{\mathsf{ct}})$. Therefore, (m^*, σ^*) is a forgery of the digital signature scheme $\mathcal{SIG}$, and $\Pr[\mathsf{E}_B \wedge \mathtt{Good}] \leq \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}}(\lambda)$ holds. Moreover, since $i^* \in [1, n]$ is randomly chosen, we get $\Pr[\mathtt{Good}] = 1/n$. Putting all together, we have $\mathsf{Adv}^{corr}_{\Pi_1,\mathcal{A}}(\lambda, n) = \Pr[\mathsf{Exp}^{corr}_{\Pi_1,\mathcal{A}}(\lambda, n) = 1] \leq \Pr[\mathsf{E}_B] = (1/\Pr[\mathtt{Good}]) \cdot \Pr[\mathsf{E}_B \wedge \mathtt{Good}] \leq n \cdot \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}}(\lambda)$. Therefore, Π_1 is correct if the NIZK proof system $\mathcal{P}_L$ satisfies completeness and the digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security. □

Full Anonymity of Scheme 1. For a signature $\Sigma = (\widetilde{\mathsf{ct}}, \mathsf{ct}, \pi)$ of Scheme 1, the user's information is contained in the encryption key ek_i and the plaintext σ of the

ciphertext $\widetilde{\mathsf{ct}}$, the plaintext $\langle \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i \rangle$ of the ciphertext ct, and the witness of the proof π. Intuitively, the information of the plaintexts σ and $\langle \mathsf{ek}_i, \mathsf{vk}_i, \mathsf{cert}_i \rangle$ is not revealed from the ciphertexts $\widetilde{\mathsf{ct}}$ and ct since the underlying PKE scheme is IND-CPA secure. Also, the information of the encryption key ek_i is not revealed from $\widetilde{\mathsf{ct}}$ by the key privacy of the underlying PKE scheme. Moreover, the information of the witness is not revealed from the proof π since the NIZK proof system $\mathcal{P}_L$ is zero-knowledge. Since these information is hidden from the adversary who has the corresponding signing key $(\mathsf{ek}_i, \mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$, Scheme 1 satisfies full anonymity. Formally, the following theorem holds.

Theorem 6.3.2. *Scheme 1 satisfies full anonymity if the underlying NIZK proof system $\mathcal{P}_L$ satisfies zero-knowledgeness and the underlying PKE scheme $\mathcal{PKE}$ satisfies IND-CPA security and key privacy.*

Proof. Let $\mathcal{A}$ be an adversary for full anonymity of Π_1. We consider the following sequence of games. Let $\Pr[\mathtt{Suc}_\ell]$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit in Game ℓ. Let b be the challenge bit, i_0 and i_1 be the challenge users, and m^* be the challenge message.

[Game 0]: This is the experiment $\mathsf{Exp}^{anon}_{\Pi_1,\mathcal{A}}(\lambda, n)$ itself. For simplicity, the challenge bit b is chosen at the beginning of the game. This change does not have an effect on the behavior of the adversary $\mathcal{A}$.

[Game 1]: This game is the same as Game 0, except that the common reference string crs in the group public key gpk, and a proof π^* in the challenge signature Σ^* are computed by using the simulator $\mathcal{S} = (\mathtt{Sim}_1, \mathtt{Sim}_2)$ of the NIZK proof system.

[Game 2]: In this game, we change the plaintext of the ciphertext ct^* in the challenge signature Σ^*. Concretely, the plaintext $0^{|\langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|}$ is encrypted to the ciphertext ct^* instead of $\langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle$.

[Game 3]: In this game, we change the plaintext of the ciphertext $\widetilde{\mathsf{ct}}^*$ in the challenge signature Σ^*. Concretely, the plaintext $0^{|\sigma^*|}$ is encrypted to the ciphertext $\widetilde{\mathsf{ct}}^*$ instead of σ^* where $\sigma^* = \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.

[Game 4]: In this game, we change the encryption key of the ciphertext $\widetilde{\mathsf{ct}}^*$. Concretely, we use a random key ek^* to compute $\widetilde{\mathsf{ct}}^*$ instead of using the key ek_{i_b}.

For the advantage $\mathsf{Adv}^{anon}_{\Pi_1,\mathcal{A}}(\lambda, n)$,

$$\mathsf{Adv}^{anon}_{\Pi,\mathcal{A}}(\lambda, n) = \left|\Pr[\mathtt{Suc}_0] - 1/2\right| \leq \sum_{\ell=0}^{3} \left|\Pr[\mathtt{Suc}_\ell] - \Pr[\mathtt{Suc}_{\ell+1}]\right| + \left|\Pr[\mathtt{Suc}_4] - 1/2\right|$$

holds. Moreover, the following lemmas hold.

Lemma 6.3.1. *There exists a* PPT *algorithm* $\mathcal{B}_1$ *such that* $\left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right| = \mathsf{Adv}^{zk}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda)$.

Proof. Let $\mathcal{B}_1$ be an adversary for the zero-knowledgeness of $\mathcal{P}_L$. First, $\mathcal{B}_1$ chooses the challenge bit b, and receives the common reference string crs from the challenger. Next, $\mathcal{B}_1$ generates the rest of instance for the scheme Π_1, and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{ek}_i, \mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. If $\mathcal{A}$ sends a query i to the Revoke oracle, $\mathcal{B}_1$ returns $\mathsf{grt}[i] = (\mathsf{dk}_i, \mathsf{vk}_i)$. For the challenge query (i_0, i_1, m^*), $\mathcal{B}_1$ computes the challenge signature Σ^* as follows:

1. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
2. Choose values r_1^* and r_2^* uniform randomly, and compute $\widetilde{\mathsf{ct}}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{i_b}, \sigma^*; r_1^*)$ and $\mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b}\rangle; r_2^*)$.
3. Set $x \leftarrow \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^*\rangle$ and $w \leftarrow \langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b}, \sigma^*, r_1^*, r_2^*\rangle$, and send (x, w) to the oracle of the NIZK proof system. Then, obtain a proof π.
4. Set $\pi^* \leftarrow \pi$, and send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_1$ outputs 1 if $b = \widetilde{b}$, and 0 otherwise. If crs is generated by the $\mathsf{ZK.Gen}$ algorithm and $\mathcal{B}_1$ accesses the Prove oracle, then $\mathcal{B}_1$ perfectly simulates Game 0 for $\mathcal{A}$. On the other hand, if crs is generated by using the simulator $\mathtt{Sim}_1$ and $\mathcal{B}_1$ accesses the $\mathsf{SimProve}$ oracle, then $\mathcal{B}_1$ perfectly simulates Game 1. Thus, since $\Pr[\mathsf{Exp}^{proof}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) = 1] = \Pr[\mathtt{Suc}_0]$ and $\Pr[\mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) = 1] = \Pr[\mathtt{Suc}_1]$ hold, we get $\mathsf{Adv}^{zk}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) = \left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right|$. □

Lemma 6.3.2. *There exists a* PPT *algorithm* $\mathcal{B}_2$ *such that* $\left|\Pr[\mathtt{Suc}_1] - \Pr[\mathtt{Suc}_2]\right| = 2 \cdot \mathsf{Adv}^{ind\text{-}cpa}_{\mathcal{PKE},\mathcal{B}_2}(\lambda)$.

Proof. Let $\mathcal{B}_2$ be an adversary for the IND-CPA security of $\mathcal{PKE}$ and β be the challenge bit in the IND-CPA security game. $\mathcal{B}_2$ chooses the challenge bit b, and receives the public key ek from the challenger. $\mathcal{B}_2$ sets $\mathsf{ek}_{\mathsf{PKE}} \leftarrow \mathsf{ek}$ and generates the common reference string crs by using the simulator $\mathtt{Sim}_1$ where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathtt{Sim}_1(1^\lambda)$. Also, $\mathcal{B}_2$ the rest of instance for the scheme Π_1 by himself. Then, $\mathcal{B}_2$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{ek}_i, \mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. If $\mathcal{A}$ sends a query i to the Revoke oracle, then $\mathcal{B}_2$ returns $\mathsf{grt}[i] = (\mathsf{dk}_i, \mathsf{vk}_i)$. For the challenge query (i_0, i_1, m^*), $\mathcal{B}_1$ computes the challenge signature Σ^* as follows:

1. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
2. Choose a value r_1^* uniform randomly, and compute $\widetilde{\mathsf{ct}}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{i_b}, \sigma^*; r_1^*)$.
3. Set $M_0 \leftarrow 0^{|\langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b}\rangle|}$ and $M_1 \leftarrow \langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b}\rangle$. Then, send (M_0, M_1) to the challenger for the IND-CPA game and obtain a ciphertext ct^*.

4. Compute $\pi^* \leftarrow \mathtt{Sim}_2(\mathsf{crs}, \mathsf{td}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle)$ where td is the trapdoor generated by $\mathtt{Sim}_1$.
5. Send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_2$ outputs $\widetilde{\beta} = 1$ if $b = \widetilde{b}$, and $\widetilde{\beta} = 0$ otherwise. If $\beta = 0$, ct^* is represented as $\mathsf{ct}^* = \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, 0^{|\langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|})$. Therefore, $\mathcal{B}_2$ perfectly simulates Game 2 if $\beta = 0$. On the other hand, if $\beta = 1$, ct^* is represented as $\mathsf{ct}^* = \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle)$, and thus $\mathcal{B}_2$ perfectly simulates Game 1. Thus, we get $\mathsf{Adv}^{ind\text{-}cpa}_{\mathcal{PKE},\mathcal{B}_2}(\lambda) = \left|\Pr[\beta = \widetilde{\beta}] - 1/2\right| = 1/2 \cdot \left|\Pr[\widetilde{\beta} = 1 | \beta = 1] - \Pr[\widetilde{\beta} = 1 | \beta = 0]\right| = 1/2 \cdot \left|\Pr[b = \widetilde{b} | \beta = 1] - \Pr[b = \widetilde{b} | \beta = 0]\right| = 1/2 \cdot \left|\Pr[\mathtt{Suc}_1] - \Pr[\mathtt{Suc}_2]\right|$. That is, it holds that $\left|\Pr[\mathtt{Suc}_1] - \Pr[\mathtt{Suc}_2]\right| = 2 \cdot \mathsf{Adv}^{ind\text{-}cpa}_{\mathcal{PKE},\mathcal{B}_2}(\lambda)$. □

Lemma 6.3.3. *There exists a* PPT *algorithm* $\mathcal{B}_3$ *such that* $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| = 2n \cdot \mathsf{Adv}^{ind\text{-}cpa}_{\mathcal{PKE},\mathcal{B}_3}(\lambda)$.

Proof. Let $\mathcal{B}_3$ be an adversary for the IND-CPA security of $\mathcal{PKE}$ and β be the challenge bit in the IND-CPA security game. $\mathcal{B}_3$ chooses the challenge bit b, and receives the public key ek from the challenger. Moreover, $\mathcal{B}_3$ chooses the index $i^* \in [1, n]$ uniform randomly. Then, $\mathcal{B}_3$ sets $\mathsf{ek}_{i^*} \leftarrow \mathsf{ek}$ and generates the common reference string crs by using the simulator $\mathtt{Sim}_1$ where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathtt{Sim}_1(1^\lambda)$. Also, $\mathcal{B}_3$ the rest of instance for the scheme Π_1 by himself. Then, $\mathcal{B}_3$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{ek}_i, \mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. We remark that $\mathcal{B}_3$ cannot compute $\mathsf{grt}[i^*]$ since $\mathcal{B}_3$ does not know the decryption key dk_{i^*} corresponding to the encryption key ek_{i^*}. However, $\mathcal{B}_3$ can generate the i^*'s user signing key $\mathsf{gsk}[i^*] = (\mathsf{ek}_{i^*}, \mathsf{vk}_{i^*}, \mathsf{sk}_{i^*}, \mathsf{cert}_{i^*})$ without knowing the value dk_{i^*}. When $\mathcal{A}$ sends a query i to the Revoke oracle, then $\mathcal{B}_3$ returns $\mathsf{grt}[i] = (\mathsf{dk}_i, \mathsf{vk}_i)$ if $i \neq i^*$. Otherwise, if $i = i^*$, then $\mathcal{B}_3$ outputs a random bit $\widetilde{\beta}$. For the challenge (i_0, i_1, m^*), $\mathcal{B}_3$ computes the challenge signature Σ^* as follows:

1. If $i_b \neq i^*$, then output a random bit $\widetilde{\beta}$. If $i_b = i^*$, go to the next step.
2. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
3. Set $M_0 \leftarrow 0^{|\sigma^*|}$ and $M_1 \leftarrow \sigma^*$. Then, send (M_0, M_1) to the challenger for the IND-CPA game and receive a ciphertext $\widetilde{\mathsf{ct}}^*$.
4. Choose a value r_2^* uniform randomly, and compute $\mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, 0^{|\langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|}; r_2^*)$.
5. Compute $\pi^* \leftarrow \mathtt{Sim}_2(\mathsf{crs}, \mathsf{td}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle)$ where td is the trapdoor generated by $\mathtt{Sim}_1$.
6. Send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_3$ outputs $\widetilde{\beta} = 1$ if $b = \widetilde{b}$, and $\widetilde{\beta} = 0$ otherwise. If $i_b = i^*$ and $\beta = 0$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{PKE.Enc}(\mathsf{ek}_{i_b}, 0^{|\sigma^*|})$. Therefore, $\mathcal{B}_3$ perfectly simulates Game 3. On the other hand, if $i_b = i^*$ and $\beta = 1$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{PKE.Enc}(\mathsf{ek}_{i_b}, \sigma^*)$, and thus $\mathcal{B}_3$ perfectly simulates Game 2. Let $\mathtt{Good}$ be the event that $i_b = i^*$ holds where i^* is chosen by $\mathcal{B}_3$ at the beginning of the game. Since the guess of $i^* \in [1, n]$ and the behavior of $\mathcal{A}$ are independent with each other, $\Pr[\mathtt{Good}] = 1/n$ holds. Thus, we get $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| = 2n \cdot \mathsf{Adv}^{\mathit{ind\text{-}cpa}}_{\mathcal{PKE},\mathcal{B}_3}(\lambda)$. □

Lemma 6.3.4. *There exists a* PPT *algorithm* $\mathcal{B}_4$ *such that* $\left|\Pr[\mathtt{Suc}_3] - \Pr[\mathtt{Suc}_4]\right| = 2n \cdot \mathsf{Adv}^{\mathit{key\text{-}priv}}_{\mathcal{PKE},\mathcal{B}_4}(\lambda)$.

Proof. Let $\mathcal{B}_4$ be an adversary for the key privacy of $\mathcal{PKE}$ and β be the challenge bit in the key privacy game. $\mathcal{B}_4$ chooses the challenge bit b, and receives two public keys ek and ek^* from the challenger. Moreover, $\mathcal{B}_4$ chooses the index $i^* \in [1, n]$ uniform randomly. Then, $\mathcal{B}_4$ sets $\mathsf{ek}_{i^*} \leftarrow \mathsf{ek}$ and generates the common reference string crs by using the simulator $\mathtt{Sim}_1$ where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathtt{Sim}_1(1^\lambda)$. Also, $\mathcal{B}_4$ the rest of instance for the scheme Π_1 by himself. Then, $\mathcal{B}_4$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{ek}_i, \mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. We remark that $\mathcal{B}_4$ cannot compute $\mathsf{grt}[i^*]$ since $\mathcal{B}_4$ does not know the decryption key dk_{i^*} corresponding to the encryption key ek_{i^*}. However, $\mathcal{B}_4$ can generate the i^*'s user signing key $\mathsf{gsk}[i^*] = (\mathsf{ek}_{i^*}, \mathsf{vk}_{i^*}, \mathsf{sk}_{i^*}, \mathsf{cert}_{i^*})$ without knowing dk_{i^*}. When $\mathcal{A}$ sends a query i to the Revoke oracle, then $\mathcal{B}_4$ returns $\mathsf{grt}[i] = (\mathsf{dk}_i, \mathsf{vk}_i)$ if $i \neq i^*$. Otherwise, if $i = i^*$, then $\mathcal{B}_4$ outputs a random bit $\widetilde{\beta}$. For the challenge (i_0, i_1, m^*), $\mathcal{B}_4$ computes the challenge signature Σ^* as follows:

1. If $i_b \neq i^*$, then output a random bit $\widetilde{\beta}$. If $i_b = i^*$, go to the next step.
2. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
3. Set $M^* \leftarrow 0^{|\sigma^*|}$. Then, send M^* to the challenger for the key privacy game and receive $\widetilde{\mathsf{ct}}^*$.
4. Choose a value r_2^* uniform randomly, and compute $\mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, 0^{|\langle \mathsf{ek}_{i_b}, \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|}; r_2^*)$.
5. Compute $\pi^* \leftarrow \mathtt{Sim}_2(\mathsf{crs}, \mathsf{td}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle)$ where td is the trapdoor generated by $\mathtt{Sim}_1$.
6. Send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_3$ outputs $\widetilde{\beta} = 1$ if $b = \widetilde{b}$, and $\widetilde{\beta} = 0$ otherwise. If $i_b = i^*$ and $\beta = 0$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{PKE.Enc}(\mathsf{ek}_{i_b}, 0^{|\sigma^*|})$, and thus $\mathcal{B}_4$ perfectly simulates Game 3. On the other hand, if $i_b = i^*$ and $\beta = 1$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{PKE.Enc}(\mathsf{ek}^*, 0^{|\sigma^*|})$, and thus $\mathcal{B}_4$ perfectly simulates Game 4. Let $\mathtt{Good}$ be the event that $i_b = i^*$ holds where i^* is chosen by $\mathcal{B}_4$ at the beginning of the game. Since the guess

of $i^* \in [1,n]$ and the behavior of $\mathcal{A}$ are independent, $\Pr[\mathtt{Good}] = 1/n$ holds. As in the same formula deformation in the proof of Lemma 8.3.3, we get $\left|\Pr[\mathtt{Suc}_3] - \Pr[\mathtt{Suc}_4]\right| = 2n \cdot \mathsf{Adv}^{key\text{-}priv}_{\mathcal{PKE},\mathcal{B}_4}(\lambda)$. □

In Game 4, the choice of the challenge bit b and the distribution of the challenge signature $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ are independent. Thus, $\Pr[\mathtt{Suc}_4] = 1/2$ holds. Putting all together, we get

$$\begin{aligned}\mathsf{Adv}^{anon}_{\Pi_1,\mathcal{A}}(\lambda,n,T) \leq& \sum_{i=0}^{3}\left|\Pr[\mathtt{Suc}_i] - \Pr[\mathtt{Suc}_{i+1}]\right| + \left|\Pr[\mathtt{Suc}_4] - 1/2\right| \\ =& \mathsf{Adv}^{zk}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) + 2 \cdot \mathsf{Adv}^{ind\text{-}cpa}_{\mathcal{PKE},\mathcal{B}_2}(\lambda) \\ &+ 2n \cdot \mathsf{Adv}^{ind\text{-}cpa}_{\mathcal{PKE},\mathcal{B}_3}(\lambda) + 2n \cdot \mathsf{Adv}^{key\text{-}priv}_{\mathcal{PKE},\mathcal{B}_4}(\lambda).\end{aligned}$$

Since the choice of the parameter n and the adversary $\mathcal{A}$ is arbitrarily, our scheme Π_1 satisfies full anonymity if the underlying NIZK proof system $\mathcal{P}_L$ satisfies zero-knowledgeness and the underlying PKE scheme $\mathcal{PKE}$ satisfies IND-CPA security and key privacy. □

Traceability of Scheme 1. Intuitively, due to the soundness of $\mathcal{P}_L$, the probability that a valid proof π for a statement $\langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m\rangle \notin L$ can be constructed is negligible where L is the language defined in Section 6.3.1. Therefore, if $\Sigma = (\widetilde{\mathsf{ct}}, \mathsf{ct}, \pi)$ is a valid signature on m, it holds that $\langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m\rangle \in L$ with high probability. Thus, there exists a witness $\langle \mathsf{ek}^*, \mathsf{vk}^*, \mathsf{cert}^*, \sigma^*, r_1^*, r_2^*\rangle$ satisfying the equations (a) $\widetilde{\mathsf{ct}} = \mathsf{PKE.Enc}(\mathsf{ek}^*, \sigma^*; r_1^*)$, (b) $\mathsf{ct} = \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{ek}^*, \mathsf{vk}^*, \mathsf{cert}^*\rangle; r_2^*)$, (c) $\mathsf{SIG.Verify}(\mathsf{vk}_{\mathsf{SIG}}, \langle \mathsf{ek}^*, \mathsf{vk}^*\rangle, \mathsf{cert}^*) = 1$, and (d) $\mathsf{SIG.Verify}(\mathsf{vk}^*, m, \sigma^*) = 1$. From the EUF-CMA security of the scheme $\mathcal{SIG}$, it is difficult to generate the value cert^* which satisfies Equation (c) for an uncertified key pair $\langle \mathsf{ek}^*, \mathsf{vk}^*\rangle$. Therefore, for some index $i \in [1,n]$, $(\mathsf{ek}^*, \mathsf{vk}^*) = (\mathsf{ek}_i, \mathsf{vk}_i)$ holds. Thus, the only way to generate a forgery is to produce a signature σ^* which satisfies Equation (d). However, it is also difficult to produce such a signature due to the EUF-CMA security of $\mathcal{SIG}$. Therefore, Scheme 1 satisfies traceability. Formally, the following theorem holds.

Theorem 6.3.3. *Scheme 1 satisfies traceability if the underlying NIZK proof system $\mathcal{P}_L$ satisfies soundness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security.*

Proof. Let $\mathcal{A}$ be an adversary for traceability of Π_1, and $(m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{trace}_{\Pi_1,\mathcal{A}}(\lambda,n,T)$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. Let i^* be the output of the $\mathsf{VLRGS.Open}$ algorithm with an input (m^*, Σ^*). We consider the following four cases:

I. $\langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^*\rangle \notin L$,

II. $i^* = \bot$, **III.** $i^* \notin \mathsf{CU}$, and **IV.** $i^* \in \mathsf{RU}^*$.

If the output of the experiment $\mathsf{Exp}^{trace}_{\Pi_1,\mathcal{A}}(\lambda, n, T)$ is 1 (i.e., $\mathcal{A}$ succeeds in producing a forged signature), we can classify the type of the forgery as follows: (1) I, (2) $\neg$I $\wedge$ II, (3) $\neg$I $\wedge$ III, and (4) $\neg$I $\wedge$ IV.

Let E_ℓ be the event that $\mathcal{A}$ outputs a forged signature in Type ℓ. We estimate the each probability that the event E_ℓ happens in the following lemmas.

Lemma 6.3.5. *There exists a* PPT *algorithm* $\mathcal{B}_1$ *such that* $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda)$.

Proof. Let $\mathcal{B}_1$ be an adversary for soundness of $\mathcal{P}_L$. First, $\mathcal{B}_1$ receives the common reference string crs from the challenger. Next, $\mathcal{B}_1$ generates the rest of instance for the scheme Π_1, and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ and $\mathbf{grt} = \{\mathsf{grt}[i]\}$ to $\mathcal{A}$ where $\mathsf{grt}[i] = (\mathsf{dk}_i, \mathsf{vk}_i)$. Since $\mathcal{B}_1$ has all the signing keys, he can easily simulate the $\mathsf{GS.Sign}$ oracle and the $\mathsf{Corrupt}$ oracle. Let $(m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. Then, $\mathcal{B}_1$ outputs $(\langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*)$ as a forgery for the soundness of $\mathcal{P}_L$. When $\mathcal{A}$'s output $(m^*, \Sigma^*, \mathsf{RU}^*)$ is a forgery in Type 1, $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ and $\langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle \notin L$ hold. Therefore, $(\langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*)$ is the forgery for the soundness of $\mathcal{P}_L$. Thus, we have $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda)$. □

Lemma 6.3.6. *There exists a* PPT *algorithm* $\mathcal{B}_2$ *such that* $\Pr[\mathsf{E}_2] \leq \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}_2}(\lambda)$.

Proof. Let $\mathcal{B}_2$ be an adversary for the EUF-CMA security of $\mathcal{SIG}$. First, $\mathcal{B}_2$ receives the verification key vk from the challenger of the EUF-CMA security game, and sets $\mathsf{vk}_{\mathsf{SIG}} \leftarrow \mathsf{vk}$. Next, $\mathcal{B}_2$ generates the rest of instance for the scheme Π_1, except for the certificates cert_i where $i \in [1, n]$. In terms of the certificates, $\mathcal{B}_2$ sends $\langle \mathsf{ek}_i, \mathsf{vk}_i \rangle$ to the Sign oracle of the scheme $\mathcal{SIG}$, and receives cert_i. $\mathcal{B}_2$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ and $\mathbf{grt} = \{\mathsf{grt}[i]\}$ to $\mathcal{A}$ where $\mathsf{grt}[i] = (\mathsf{dk}_i, \mathsf{vk}_i)$. Since $\mathcal{B}_1$ has all the signing keys, he can easily simulate the Sign oracle and the $\mathsf{Corrupt}$ oracle. Let $(m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. Then, $\mathcal{B}_2$ outputs $(\langle \mathsf{ek}^*, \mathsf{vk}^* \rangle, \mathsf{cert}^*)$ as a forged signature of $\mathcal{SIG}$. When $\mathcal{A}$'s output $(m^*, \Sigma^*, \mathsf{RU}^*)$ is a forgery in Type 2, $\langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle \in L$ holds. Also, $\mathsf{SIG.Verify}(\mathsf{vk}_{\mathsf{SIG}}, \langle \mathsf{ek}^*, \mathsf{vk}^* \rangle, \mathsf{cert}^*) = 1$ holds where $\langle \mathsf{ek}^*, \mathsf{vk}^*, \mathsf{cert}^* \rangle$ is the decryption result of ct^* by the decryption key $\mathsf{dk}_{\mathsf{PKE}}$. Since for all $i \in [1, n]$, $(\mathsf{ek}^*, \mathsf{vk}^*) \neq (\mathsf{ek}_i, \mathsf{vk}_i)$ holds, $\mathcal{B}_2$ does not send $(\mathsf{ek}^*, \mathsf{vk}^*)$ to the Sign oracle. Thus, $(\langle \mathsf{ek}^*, \mathsf{vk}^* \rangle, \mathsf{cert}^*)$ is a forged signature of the digital signature scheme $\mathcal{SIG}$, and we get $\Pr[\mathsf{E}_2] \leq \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}_2}(\lambda)$. □

Lemma 6.3.7. *There exists a* PPT *algorithm* $\mathcal{B}_3$ *such that* $\Pr[\mathsf{E}_3] \leq n \cdot \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}_3}(\lambda)$.

Proof. Let $\mathcal{B}_3$ be an adversary for the EUF-CMA security of $\mathcal{SIG}$. First, $\mathcal{B}_3$ receives vk from the challenger of the EUF-CMA security game, and randomly chooses the index $i^* \in [1, n]$. Then, $\mathcal{B}_3$ sets $\mathsf{vk}_{i^*} \leftarrow \mathsf{vk}$. Next, $\mathcal{B}_3$ generates the rest of instance

for the scheme Π_1, and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}})$ and $\mathbf{grt} = \{\mathsf{grt}[i]\}$ to $\mathcal{A}$ where $\mathsf{grt}[i] = (\mathsf{dk}_i, \mathsf{vk}_i)$. We remark that $\mathcal{B}_3$ does not know the signing key sk_{i^*} corresponding to vk_{i^*}. Thus, $\mathcal{B}_3$ cannot compute $\mathsf{gsk}[i^*]$. If $\mathcal{A}$ sends (i, m) to the Sign oracle and it holds $i \neq i^*$, then $\mathcal{B}_3$ easily computes Σ since $\mathcal{B}_3$ knows $\mathsf{gsk}[i]$ for all the users $i \neq i^*$. On the other hand, if $\mathcal{A}$ sends (i^*, m) to the Sign oracle, $\mathcal{B}_3$ sends m to the Sign oracle of $\mathcal{SIG}$ and receives a signature σ. Then, he computes $(\widetilde{\mathsf{ct}}, \mathsf{ct})$ and π according to the $\mathsf{VLRGS.Sign}$ algorithm, and returns $\Sigma = (\widetilde{\mathsf{ct}}, \mathsf{ct}, \pi)$ to $\mathcal{A}$. If $\mathcal{A}$ sends i^* to the $\mathsf{Corrupt}$ oracle, then $\mathcal{B}_3$ outputs $\perp$. Otherwise, if $\mathcal{A}$ sends i such that $i \neq i^*$ to the $\mathsf{Corrupt}$ oracle, then $\mathcal{B}_3$ returns $\mathsf{gsk}[i]$. Let $(m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. If $\mathsf{VLRGS.Open}(\mathsf{gpk}, \mathbf{grt}, m^*, \Sigma^*) \neq i^*$, $\mathcal{B}_3$ outputs $\perp$. Otherwise, if $\mathsf{VLRGS.Open}(\mathsf{gpk}, \mathbf{grt}, m^*, \Sigma^*) = i^*$, then $\mathcal{B}_3$ decrypts $\widetilde{\mathsf{ct}}^*$ by using dk_{i^*} and gets the decryption result σ^*. Then, $\mathcal{B}_3$ outputs (m^*, σ^*) as a forged signature. Since the guess of $i^* \in [1, n]$ and the behavior of $\mathcal{A}$ are independent with each other, the probability that $\mathsf{VLRGS.Open}(\mathsf{gpk}, \mathbf{grt}, m^*, \Sigma^*) = i^*$ holds is $1/n$. If $\mathsf{VLRGS.Open}(\mathsf{gpk}, \mathbf{grt}, m^*, \Sigma^*) = i^*$, $i^* \notin \mathsf{CU}$ holds by the condition of Type 3. Thus, i^* is not queried to the $\mathsf{Corrupt}$ oracle and $\mathcal{B}_3$ can succeed in simulating the $\mathsf{Corrupt}$ oracle. Moreover, due to the success condition of $\mathcal{A}$, (i^*, m^*) is not queried to the Sign oracle. That is, m^* is not queried to the Sign oracle. Moreover, if $\mathsf{VLRGS.Open}(\mathsf{gpk}, \mathbf{grt}, m^*, \Sigma^*) = i^*$, it holds that $\mathsf{VLRGS.Verify}(\mathsf{gpk}, \mathsf{grt}[i^*], m^*, \Sigma^*) = 0$. Thus, either $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 0$ or $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{PKE.Dec}(\mathsf{dk}_{i^*}, \widetilde{\mathsf{ct}}^*)) = 1$ hold. Here, due to the success condition of $\mathcal{A}$, $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ holds. Thus, $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{PKE.Dec}(\mathsf{dk}_{i^*}, \widetilde{\mathsf{ct}}^*)) = 1$ holds. Therefore, if $\mathsf{VLRGS.Open}(\mathsf{gpk}, \mathbf{grt}, m^*, \Sigma^*) = i^*$ holds, (m^*, σ^*) is a forged signature of the $\mathcal{SIG}$ scheme where σ^* is the decryption result of $\widetilde{\mathsf{ct}}^*$ by using the decryption key dk_{i^*}. Since the probability that $\mathsf{VLRGS.Open}(\mathsf{gpk}, \mathbf{grt}, m^*, \Sigma^*) = i^*$ holds is $1/n$, we have $(1/n) \cdot \Pr[\mathsf{E}_3] \leq \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG}, \mathcal{B}_3}(\lambda)$. □

Lemma 6.3.8. $\Pr[\mathsf{E}_4] = 0$ *holds.*

Proof. Let $(m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$, and i^* be the result of the the $\mathsf{VLRGS.Open}$ algorithm with the input (m^*, Σ^*). If $i^* \in \mathsf{RU}^*$, then $\mathsf{grt}[i^*] \in \mathsf{RL}^*$. Due to the success probability, $\mathsf{VLRGS.Verify}(\mathsf{gpk}, \mathsf{RL}^*, m^*, \Sigma^*) = 1$ holds. Thus, due to the description of the $\mathsf{VLRGS.Verify}$ algorithm, $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ holds. Also, for $\mathsf{grt}[i^*] = (\mathsf{dk}_{i^*}, \mathsf{vk}_{i^*}) \in \mathsf{RL}^*$, it holds that $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{PKE.Dec}(\mathsf{dk}_{i^*}, \widetilde{\mathsf{ct}}^*)) = 0$. Moreover, since the opening result is i^*, we have $\mathsf{VLRGS.Verify}(\mathsf{gpk}, \mathsf{grt}[i^*], m^*, \Sigma^*) = 0$. Thus, either $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ or $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{PKE.Dec}(\mathsf{dk}_{i^*}, \widetilde{\mathsf{ct}}^*)) = 1$ holds. However, this contradicts the condition that $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{gpk}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ and $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{PKE.Dec}(\mathsf{dk}_{i^*}, \widetilde{\mathsf{ct}}^*)) = 0$. Thus, we get $\Pr[\mathsf{E}_4] = 0$. □

Putting all together, we get

$$\begin{aligned}
\mathsf{Adv}^{trace}_{\Pi_1,\mathcal{A}}(\lambda,n,T) &= \Pr[\ \mathsf{Exp}^{trace}_{\Pi_1,\mathcal{A}}(\lambda,n,T)=1\ ] \\
&= \Pr[\ \mathsf{E}_1 \vee \mathsf{E}_2 \vee \mathsf{E}_3 \vee \mathsf{E}_4\ ] \\
&= \Pr[\mathsf{E}_1] + \Pr[\mathsf{E}_2] + \Pr[\mathsf{E}_3] + \Pr[\mathsf{E}_4] \\
&\leq \mathsf{Adv}^{sound}_{\mathcal{P}_L,\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}_2}(\lambda) + n \cdot \mathsf{Adv}^{euf\text{-}cma}_{\mathcal{SIG},\mathcal{B}_4}(\lambda).
\end{aligned}$$

Since the choice of the parameter n and the adversary $\mathcal{A}$ is arbitrarily, our scheme Π_1 satisfies traceability if the underlying NIZK proof system $\mathcal{P}_L$ satisfies soundness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security. □

6.3.2 Schemes with Backward Unlinkability

In this section, we give two constructions of a fully anonymous VLR-GS scheme with backward unlinkability. The first scheme called Scheme 2, is a naive scheme extended Scheme 1. The second scheme called Scheme 3, is obtained by modifying Scheme 2. and has the size of the user signing key which does not depend on the number of time periods.

Firstly, we explain about Scheme 2. In Scheme 2, encryption/decryption key pairs of a user i are provided for each time period j. When generating a signature at the time period j, a user i uses the corresponding encryption key $\mathsf{ek}_i^{(j)}$ to encrypt a signature σ. Also, the decryption key $\mathsf{dk}_i^{(j)}$ is set to be i's revocation token for the time period j. To force users to use the encryption key related to the appropriate time period, each encryption key $\mathsf{ek}_i^{(j)}$ is certified along with the verification key vk_i by using the manager's signing key $\mathsf{sk}_{\mathsf{SIG}}^{(j)}$ that depends on the time period j.

Scheme 2 is given in Figure 6.2. The building blocks are the same as those of Scheme 1, that is, we use a digital signature scheme $\mathcal{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify})$, a PKE scheme $\mathcal{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$, and a NIZK proof system $\mathcal{P}_L = (\mathsf{ZK.Gen}, \mathsf{ZK.Prove}, \mathsf{ZK.Verify})$ where the relation R_L is defined in Section 6.3.1 as building blocks.

For Scheme 2, Theorem 6.3.4 to 6.3.6 hold. Since these theorems can be shown as the case of Scheme 1, we do not give their proofs in this thesis.

Theorem 6.3.4. *Scheme 2 is correct if the underlying NIZK proof system $\mathcal{P}_L$ satisfies completeness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security.*

Theorem 6.3.5. *Scheme 2 satisfies full anonymity if the underlying NIZK proof system $\mathcal{P}_L$ satisfies zero-knowledgeness and the underlying PKE scheme $\mathcal{PKE}$ satisfies IND-CPA security and key privacy.*

$\mathsf{VLRGS.Gen}(1^\lambda, n, T)$:
$\mathsf{crs} \leftarrow \mathsf{ZK.Gen}(1^\lambda)$
For $1 \le j \le T$: $(\mathsf{vk}_{\mathsf{SIG}}^{(j)}, \mathsf{sk}_{\mathsf{SIG}}^{(j)}) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$
$(\mathsf{ek}_{\mathsf{PKE}}, \mathsf{dk}_{\mathsf{PKE}}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$
For $1 \le i \le n$ and $1 \le j \le T$:
$(\mathsf{ek}_i^{(j)}, \mathsf{dk}_i^{(j)}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$; $(\mathsf{vk}_i, \mathsf{sk}_i) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$
$\mathsf{cert}_i^{(j)} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{\mathsf{SIG}}^{(j)}, \langle \mathsf{ek}_i^{(j)}, \mathsf{vk}_i \rangle)$; $\mathsf{grt}[i][j] \leftarrow (\mathsf{dk}_i^{(j)}, \mathsf{vk}_i)$
$\mathsf{gpk} = (\mathsf{crs}, \{\mathsf{vk}_{\mathsf{SIG}}^{(j)}\}_j, \mathsf{ek}_{\mathsf{PKE}})$; $\mathsf{gsk}[i] = (\{\mathsf{ek}_i^{(j)}\}_j, \mathsf{vk}_i, \mathsf{sk}_i, \{\mathsf{cert}_i^{(j)}\}_j)$
$\mathbf{gsk} = \{\mathsf{gsk}[i]\}_i$; $\mathbf{grt} = \{\mathsf{grt}[i][j]\}_{ij}$
Return $(\mathsf{gpk}, \mathbf{gsk}, \mathbf{grt})$

$\mathsf{VLRGS.Sign}(\mathsf{gpk}, j, \mathsf{gsk}[i], m)$:
$\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_i, m)$
$\widetilde{\mathsf{ct}} \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_i^{(j)}, \sigma; r_1)$
$\mathsf{ct} \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{ek}_i^{(j)}, \mathsf{vk}_i, \mathsf{cert}_i^{(j)} \rangle; r_2)$
$\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}^{(j)}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle, \langle \mathsf{ek}_i^{(j)}, \mathsf{vk}_i, \mathsf{cert}_i^{(j)}, \sigma, r_1, r_2 \rangle)$
Return $\Sigma = (\widetilde{\mathsf{ct}}, \mathsf{ct}, \pi)$

$\mathsf{VLRGS.Verify}(\mathsf{gpk}, j, \mathsf{RL}_j, m, \Sigma)$:
If $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}^{(j)}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle, \pi) = 0$, return 0
For $(\mathsf{dk}, \mathsf{vk}) \in \mathsf{RL}_j$:
If $\mathsf{SIG.Verify}(\mathsf{vk}, m, \mathsf{PKE.Dec}(\mathsf{dk}, \widetilde{\mathsf{ct}})) = 1$, return 0
Return 1

Figure 6.2: **Scheme 2.** A Naive VLR-GS Scheme with Backward Unlinkability

Theorem 6.3.6. *Scheme 2 satisfies traceability if the underlying NIZK proof system $\mathcal{P}_L$ satisfies soundness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security.*

A Drawback of Scheme 2. Although Scheme 2 satisfies backward unlinkability, it has one drawback that the size of the user signing key depends on the number of time periods. This is because the user needs to change the encryption key to encrypt a signature σ for the time period when generating a signature Σ. Therefore, the user i possesses T encryption keys $\mathsf{ek}_i^{(1)}, \ldots, \mathsf{ek}_i^{(T)}$ as a part of the user signing key where T is the number of time periods. Consequently, the size of certificate is also grown.

Since the number of time periods is fixed in the setup phase and the user signing keys are also fixed at the beginning of using the system, it is not necessary to redistribute the user signing keys. However, it is still undesirable that the size of the user signing key depends on the number of time periods.

Secondly, we give a description of Scheme 3 where the size of the user signing key does not depend on the number of time periods. Scheme 3 can be obtained by modifying

Scheme 2. Intuitively, we replace the T encryption/decryption key pairs of a PKE scheme in Scheme 2 with one identity (an encryption key) and the corresponding decryption keys of an IBE scheme.

More precisely, in Scheme 3, key pairs of an IBE scheme $(\mathsf{params}^{(1)}, \mathsf{msk}^{(1)}), \ldots, (\mathsf{params}^{(T)}, \mathsf{msk}^{(T)})$ are introduced instead of key pairs of a PKE scheme depending on the users and the time periods. Therefore, users no longer have their own encryption keys in Scheme 3. When a user i generates a signature at the time period j, he encrypts the internal signature σ by using his verification key vk_i as an encryption key under the system parameters $\mathsf{params}^{(j)}$ with the corresponding index j. Then, by using NIZK proof system, the user proves that he used the appropriate system parameters with the current time period and his certified verification key as an encryption key. The manager considers a user i's verification key vk_i as an identity and generates the corresponding decryption keys dk_{ij} using each master secret key $\mathsf{msk}^{(j)}$. Then, the key dk_{ij} is set to be i's revocation token for the time period j. Since the decryption key dk_{ij} can be used to decrypt only the ciphertext which is generated with the verification key vk_i and the system parameters $\mathsf{params}^{(j)}$, it works as the revocation token of the user i at the time period j.

Due to employing an IBE scheme, a user does not need to possess an additional key of a PKE scheme. Thus, a user i's signing key is a tuple of a signing/verification key pair $(\mathsf{vk}_i, \mathsf{sk}_i)$ of a signature scheme and its certificate cert_i. This size no longer depends on the number of time periods and especially, is the same as that of the BMW construction [14].

The description of Scheme 3 is given in Figure 6.3. Concretely, we construct a VLR-GS scheme $\Pi_3 = (\mathsf{GS.Gen}, \mathsf{GS.Sign}, \mathsf{GS.Verify})$ from a digital signature scheme $\mathcal{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify})$, a PKE scheme $\mathcal{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$, an IBE scheme $\mathcal{IBE} = (\mathsf{IBE.Gen}, \mathsf{IBE.Ext}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$, and a NIZK proof system $\mathcal{P}_{\widehat{L}} = (\mathsf{ZK.Gen}, \mathsf{ZK.Prove}, \mathsf{ZK.Verify})$. We say that a statement $x = \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j)}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle$ and a witness $w = \langle \mathsf{vk}_i, \mathsf{cert}_i, \sigma, r_1, r_2 \rangle$ satisfy the relation $R_{\widehat{L}}$ if the following equations hold:

(a) $\widetilde{\mathsf{ct}} = \mathsf{IBE.Enc}(\mathsf{params}^{(j)}, \mathsf{vk}_i, \sigma; r_1)$,

(b) $\mathsf{ct} = \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{vk}_i, \mathsf{cert}_i \rangle; r_2)$,

(c) $\mathsf{SIG.Verify}(\mathsf{vk}_{\mathsf{SIG}}, \mathsf{vk}_i, \mathsf{cert}_i) = 1$,

(d) $\mathsf{SIG.Verify}(\mathsf{vk}_i, m, \sigma) = 1$.

Moreover, for a statement $x = \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j)}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle$, if there exists a witness that satisfies the above equations, then we say that the statement x belongs to the language $\widehat{L}$ and denote it $x \in \widehat{L}$.

For the correctness of Scheme 3, the following theorem holds.

$\mathsf{GS.Gen}(1^\lambda, n, T)$:
  $\mathsf{crs} \leftarrow \mathsf{ZK.Gen}(1^\lambda)$; $(\mathsf{vk_{SIG}}, \mathsf{sk_{SIG}}) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$
  For $1 \le j \le T$: $(\mathsf{params}^{(j)}, \mathsf{msk}^{(j)}) \leftarrow \mathsf{IBE.Gen}(1^\lambda)$
  $(\mathsf{ek_{PKE}}, \mathsf{dk_{PKE}}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$
  For $1 \le i \le n$ and $1 \le j \le T$:
    $(\mathsf{vk}_i, \mathsf{sk}_i) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$; $\mathsf{cert}_i \leftarrow \mathsf{SIG.Sign}(\mathsf{sk_{SIG}}, \mathsf{vk}_i)$
    $\mathsf{dk}_{ij} \leftarrow \mathsf{IBE.Ext}(\mathsf{params}^{(j)}, \mathsf{msk}^{(j)}, \mathsf{vk}_i)$; $\mathsf{grt}[i][j] \leftarrow (\mathsf{dk}_{ij}, \mathsf{vk}_i)$
  $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk_{SIG}}, \mathsf{ek_{PKE}}, \{\mathsf{params}^{(j)}\}_j)$; $\mathsf{gsk}[i] = (\mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$
  $\mathbf{gsk} = \{\mathsf{gsk}[i]\}_j$; $\mathbf{grt} = \{\mathsf{grt}[i][j]\}_{ij}$
  Return $(\mathsf{gpk}, \mathbf{gsk}, \mathbf{grt})$

$\mathsf{GS.Sign}(\mathsf{gpk}, j, \mathsf{gsk}[i], m)$:
  $\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_i, m)$
  $\widetilde{\mathsf{ct}} \leftarrow \mathsf{IBE.Enc}(\mathsf{params}^{(j)}, \mathsf{vk}_i, \sigma; r_1)$
  $\mathsf{ct} \leftarrow \mathsf{PKE.Enc}(\mathsf{ek_{PKE}}, \langle \mathsf{vk}_i, \mathsf{cert}_i \rangle; r_2)$
  $\pi \leftarrow \mathsf{ZK.Prove}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j)}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle, \langle \mathsf{vk}_i, \mathsf{cert}_i, \sigma, r_1, r_2 \rangle)$
  Return $\Sigma = (\widetilde{\mathsf{ct}}, \mathsf{ct}, \pi)$

$\mathsf{GS.Verify}(\mathsf{gpk}, j, \mathsf{RL}_j, m, \Sigma)$:
  If $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j)}, \widetilde{\mathsf{ct}}, \mathsf{ct}, m \rangle, \pi) = 0$, return 0
  For $(\mathsf{dk}, \mathsf{vk}) \in \mathsf{RL}_j$:
    If $\mathsf{SIG.Verify}(\mathsf{vk}, m, \mathsf{IBE.Dec}(\mathsf{dk}, \widetilde{\mathsf{ct}})) = 1$, return 0
  Return 1

Figure 6.3: **Scheme 3.** A VLR-GS Scheme with Backward Unlinkability

Theorem 6.3.7. *Scheme 3 is correct if the underlying NIZK proof system $\mathcal{P}_{\widehat{L}}$ satisfies completeness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security.*

Proof. Let $\mathcal{A}$ be an adversary for the correctness of Π_3 and the output of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{corr}_{\Pi_3,\mathcal{A}}(\lambda, n, T)$ be $(i^*, j^*, m^*, \mathsf{RU}^*)$. If the experiment $\mathsf{Exp}^{corr}_{\Pi_3,\mathcal{A}}(\lambda, n, T)$ outputs 1, $\mathsf{VLRGS.Verify}(\mathsf{gpk}, j^*, \mathsf{RL}_{j^*}, m^*, \Sigma^*) = 0$ and $i^* \notin \mathsf{RU}^*$ hold where $\mathsf{RL}_{j^*} = \{grt[i][j^*] \mid i \in \mathsf{RU}^*\}$ and $\Sigma^* \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, j^*, \mathsf{gsk}[i^*], m^*)$. Let $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. When $\mathsf{VLRGS.Verify}(\mathsf{gpk}, j^*, \mathsf{RL}_{j^*}, m^*, \Sigma^*) = 0$ holds, either the following event E_A or the event E_B happens.

E_A: $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 0$ holds.

E_B: For some $i \in \mathsf{RU}^*$, $\mathsf{SIG.Verify}(\mathsf{vk}_i, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{ij^*}, \widetilde{\mathsf{ct}}^*)) = 1$ holds.

However, $\Pr[\mathsf{E}_A] = 0$ holds if $\mathcal{P}_{\widehat{L}}$ satisfies completeness. Therefore, it holds that $\Pr[\mathsf{Exp}^{corr}_{\Pi_3,\mathcal{A}}(\lambda, n, T) = 1] = \Pr[\mathsf{E}_A \vee \mathsf{E}_B] \le \Pr[\mathsf{E}_A] + \Pr[\mathsf{E}_B] = \Pr[\mathsf{E}_B]$.

We evaluate $\Pr[\mathsf{E}_B]$ by constructing an algorithm $\mathcal{B}$ that breaks the EUF-CMA security of the digital signature scheme $\mathcal{SIG}$. At the beginning of the game, $\mathcal{B}$ randomly

chooses $\widehat{i} \in [1, n]$, and sets $\mathsf{vk}_{\widehat{i}} \leftarrow \mathsf{vk}$ where vk is the key given by the challenger of the EUF-CMA security game. $\mathcal{B}$ generates the rest of instance for the scheme Π_3 and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ to $\mathcal{A}$. For the $\mathcal{A}$'s output $(i^*, j^*, m^*, \mathsf{RU}^*)$, $\mathcal{B}$ outputs $\perp$ if $\widehat{i} = i^*$. Otherwise, if $\widehat{i} \neq i^*$, $\mathcal{B}$ computes $\Sigma^* \leftarrow \mathsf{VLRGS.Sign}(\mathsf{gpk}, j^*, \mathsf{gsk}[i^*], m^*)$. Then, $\mathcal{B}$ computes $\sigma^* \leftarrow \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)$, and outputs (m, σ^*) as a forged signature.

When the event E_B happens, there exists at least one pair $(\mathsf{dk}_{ij^*}, \mathsf{vk}_i) \in \mathsf{RL}_{j^*}$ such that $\mathsf{SIG.Verify}(\mathsf{vk}_i, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{ij^*}, \widetilde{\mathsf{ct}}^*)) = 1$ holds. Let I be the set of such indexes i and $\mathtt{Good}$ be the event that $i^* \in I$ holds. Since the guess of $i^* \in [1, n]$ and the behavior of $\mathcal{A}$ are independent, we get $\Pr[\mathsf{E}_B \wedge \mathtt{Good}] = \Pr[\mathsf{E}_B] \cdot \Pr[\mathtt{Good}]$. When both events E_B and $\mathtt{Good}$ happen, it holds that $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \sigma^*) = 1$ where $\sigma^* \leftarrow \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)$. Therefore, (m^*, σ^*) is a forgery of the digital signature scheme $\mathcal{SIG}$, and $\Pr[\mathsf{E}_B \wedge \mathtt{Good}] \leq \mathsf{Adv}^{unforge}_{\mathcal{SIG},\mathcal{B}}(\lambda)$ holds. Since $i^* \in [1, n]$ is randomly chosen, we get $\Pr[\mathtt{Good}] = 1/n$, we have $\mathsf{Adv}^{corr}_{\Pi_3,\mathcal{A}}(\lambda, n, T) = \Pr[\mathsf{Exp}^{corr}_{\Pi_3,\mathcal{A}}(\lambda, n, T) = 1] \leq \Pr[\mathsf{E}_B] = (1/\Pr[\mathtt{Good}]) \cdot \Pr[\mathsf{E}_B \wedge \mathtt{Good}] \leq n \cdot \mathsf{Adv}^{unforge}_{\mathcal{SIG},\mathcal{B}}(\lambda)$. Since the choice of the parameters n and T, and the adversary $\mathcal{A}$ is arbitrarily, Π_3 is correct if the NIZK proof system $\mathcal{P}_{\widehat{L}}$ satisfies completeness and the digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security. □

Moreover, for the security of Scheme 3, Theorem 6.3.8 and 6.3.9 hold.

Theorem 6.3.8. *Scheme 3 satisfies full anonymity if the underlying NIZK proof system $\mathcal{P}_{\widehat{L}}$ satisfies zero-knowledgeness, the underlying PKE scheme $\mathcal{PKE}$ satisfies IND-CPA security, and the underlying IBE scheme $\mathcal{IBE}$ satisfies IND-ID-CPA security and key privacy.*

Proof. Let $\mathcal{A}$ be an adversary for full anonymity of Scheme 3. We consider the following sequence of games. Let $\Pr[\mathtt{Suc}_\ell]$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit in Game ℓ. Let b be the challenge bit, i_0 and i_1 be the challenge users, j^* be the challenge time period, and m^* be the challenge message.

[Game 0]: This is the experiment $\mathsf{Exp}^{anon}_{\Pi_3,\mathcal{A}}(\lambda, n, T)$ itself. For simplicity, the challenge bit b is chosen at the beginning of the game. This change does not have an effect on the behavior of the adversary $\mathcal{A}$.

[Game 1]: This game is the same as Game 0, except that the common reference string crs in the group public key gpk and a proof π^* in the challenge signature Σ^* are computed by using the simulator $\mathcal{S}$ of the NIZK proof system.

[Game 2]: In this game, we change the plaintext of the ciphertext ct^* in the challenge signature Σ^*. Concretely, the plaintext $0^{|\langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|}$ is encrypted to the ciphertext ct^* instead of $\langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle$.

[Game 3]: In Game 3, we change the plaintext of the ciphertext $\widetilde{\mathsf{ct}}^*$ in the challenge

signature Σ^*. Concretely, the plaintext $0^{|\sigma^*|}$ is encrypted to the ciphertext $\widetilde{\mathsf{ct}}^*$ instead of σ^* where $\sigma^* = \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.

[Game 4]: In this game, we change the encryption key of the ciphertext $\widetilde{\mathsf{ct}}^*$. Concretely, we use a random key vk^* to compute $\widetilde{\mathsf{ct}}^*$ instead of using the key vk_{i_b}.

For these games, the following lemmas hold.

Lemma 6.3.9. *There exists a* PPT *algorithm* $\mathcal{B}_1$ *such that* $\left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right| = \mathsf{Adv}^{zk}_{\mathcal{P}_{\widehat{L}},\mathcal{B}_1}(\lambda)$.

Proof. Let $\mathcal{B}_1$ be an adversary for the zero-knowledgeness of $\mathcal{P}_{\widehat{L}}$. First, $\mathcal{B}_1$ chooses the challenge bit b, and receives the common reference string crs from the challenger in the zero-knowledge game. Next, $\mathcal{B}_1$ generates the rest of instance for the scheme Π_3, and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}_i$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. If $\mathcal{A}$ sends a query (i, j) to the Revoke oracle, $\mathcal{B}_1$ returns $\mathsf{grt}[i][j] = (\mathsf{dk}_{ij}, \mathsf{vk}_i)$. For the challenge query (i_0, i_1, j^*, m^*), $\mathcal{B}_1$ computes the challenge signature Σ^* as follows:

1. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
2. Choose values r_1^* and r_2^* uniform randomly. Then, compute ciphertexts $\widetilde{\mathsf{ct}}^* \leftarrow \mathsf{IBE.Enc}(\mathsf{params}^{(j^*)}, \mathsf{vk}_{i_b}, \sigma^*; r_1^*)$ and $\mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle; r_2^*)$.
3. Set $x \leftarrow \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle$ and $w \leftarrow \langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b}, \sigma^*, r_1^*, r_2^* \rangle$, and send (x, w) to the oracle in the zero-knowledge game. Then, obtain a proof π.
4. Set $\pi^* \leftarrow \pi$, and send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_1$ outputs 1 if $b = \widetilde{b}$, and 0 otherwise.

If the common reference string crs which $\mathcal{B}_1$ obtained is generated by the $\mathsf{ZK.Gen}$ algorithm, and the oracle which $\mathcal{B}_1$ accessed is the Prove oracle, then $\mathcal{B}_1$ perfectly simulates Game 0 for $\mathcal{A}$. On the other hand, if the common reference string crs which $\mathcal{B}_1$ obtained is generated by the simulator $\mathtt{Sim}_1$, and the oracle which $\mathcal{B}_1$ accessed is the $\mathsf{SimProve}$ oracle, then $\mathcal{B}_1$ perfectly simulates Game 1. Thus, we get $\mathsf{Adv}^{zk}_{\mathcal{P}_{\widehat{L}},\mathcal{B}_1}(\lambda) = \left|\Pr[\mathsf{Exp}^{proof}_{\mathcal{P}_{\widehat{L}},\mathcal{B}_1}(\lambda) = 1] - \Pr[\mathsf{Exp}^{sim\text{-}proof}_{\mathcal{P}_{\widehat{L}},\mathcal{B}_1}(\lambda) = 1]\right| = \left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right|$. □

Lemma 6.3.10. *There exists a* PPT *algorithm* $\mathcal{B}_2$ *such that* $\left|\Pr[\mathtt{Suc}_1] - \Pr[\mathtt{Suc}_2]\right| = 2 \cdot \mathsf{Adv}^{ind\text{-}cpa}_{\mathcal{PKE},\mathcal{B}_2}(\lambda)$.

Proof. Let $\mathcal{B}_2$ be an adversary for the IND-CPA security of $\mathcal{PKE}$ and β be the challenge bit in the IND-CPA security game. At the beginning of the game, $\mathcal{B}_2$ chooses the challenge bit b for the anonymity game, and receives the public key ek from the challenger. $\mathcal{B}_2$ sets

$\mathsf{ek}_{\mathsf{PKE}} \leftarrow \mathsf{ek}$ and generates the common reference string crs by using the simulator $\mathtt{Sim}_1$ where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathtt{Sim}_1(1^\lambda)$. Also, $\mathcal{B}_2$ the rest of instance for the scheme Π_3 by himself. Then, $\mathcal{B}_2$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}_i$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. If $\mathcal{A}$ sends a query (i, j) to the Revoke oracle, then $\mathcal{B}_2$ returns $\mathsf{grt}[i][j] = (\mathsf{dk}_{ij}, \mathsf{vk}_i)$. For the challenge query (i_0, i_1, j^*, m^*), $\mathcal{B}_2$ computes the challenge signature Σ^* as follows:

1. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
2. Choose a value r_1^* uniform randomly, and compute a ciphertext $\widetilde{\mathsf{ct}}^* \leftarrow \mathsf{IBE.Enc}(\mathsf{params}^{(j^*)}, \mathsf{vk}_{i_b}, \sigma^*; r_1^*)$.
3. Set $M_0 \leftarrow 0^{|\langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|}$ and $M_1 \leftarrow \langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle$. Then, send (M_0, M_1) to the challenger for the IND-CPA game and obtain a ciphertext ct. Set $\mathsf{ct}^* \leftarrow \mathsf{ct}$.
4. Compute $\pi^* \leftarrow \mathtt{Sim}_2(\mathsf{crs}, \mathsf{td}, \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle)$ where td is the trapdoor generated by $\mathtt{Sim}_1$.
5. Send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_2$ outputs $\widetilde{\beta} = 1$ if $b = \widetilde{b}$, and $\widetilde{\beta} = 0$ otherwise.

If $\beta = 0$, ct^* is represented as $\mathsf{ct}^* = \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, 0^{|\langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|})$. Therefore, $\mathcal{B}_2$ perfectly simulates Game 2. On the other hand, if $\beta = 1$, ct^* is to be $\mathsf{ct}^* = \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, \langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle)$, and thus $\mathcal{B}_2$ perfectly simulates Game 1. Thus, we get $\left|\Pr[\mathtt{Suc}_1] - \Pr[\mathtt{Suc}_2]\right| = 2 \cdot \mathsf{Adv}^{\mathit{ind\text{-}cpa}}_{\mathcal{PKE}, \mathcal{B}_2}(\lambda)$. □

Lemma 6.3.11. *There exists a* PPT *algorithm* $\mathcal{B}_3$ *such that* $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| = 2T \cdot \mathsf{Adv}^{\mathit{ind\text{-}id\text{-}cpa}}_{\mathcal{IBE}, \mathcal{B}_3}(\lambda)$.

Proof. Let $\mathcal{B}_3$ be an adversary for the IND-ID-CPA security of $\mathcal{IBE}$ and β be the challenge bit in the IND-ID-CPA security game. At the beginning of the game, $\mathcal{B}_3$ chooses the challenge bit b, and guesses a time period $j^* \in [1, T]$ uniform randomly. Then, $\mathcal{B}_3$ sets $\mathsf{params}^{(j^*)} \leftarrow \mathsf{params}$ where params is the system parameters given by the challenger. The rest of instance for the scheme Π_3 is generated by $\mathcal{B}_3$, however, only the common reference string crs is yielded by the simulator $\mathtt{Sim}_1$. $\mathcal{B}_3$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}_i$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. We note that $\mathcal{B}_3$ can compute all the user signing keys $\mathbf{gsk}$ even he does not know the master secret key $\mathsf{msk}^{(j^*)}$ corresponding to $\mathsf{params}^{(j^*)}$. When $\mathcal{A}$ sends a query (i, j) to the Revoke oracle, $\mathcal{B}_3$ answers it as follows. If $j \neq j^*$, $\mathcal{B}_3$ computes $\mathsf{dk}_{ij} \leftarrow \mathsf{IBE.Ext}(\mathsf{params}^{(j)}, \mathsf{msk}^{(j)}, \mathsf{vk}_i)$ and returns $\mathsf{grt}[i][j] = (\mathsf{dk}_{ij}, \mathsf{vk}_i)$ to $\mathcal{A}$. If $j = j^*$, $\mathcal{B}_3$ queries vk_i to the $\mathsf{Extract}$ oracle of the IND-ID-CPA security game and obtains dk_{ij^*}. Then, he returns $\mathsf{grt}[i][j^*] = (\mathsf{dk}_{ij^*}, \mathsf{vk}_i)$ as the

reply for $\mathcal{A}$. Let $(i_0, i_1, j_{\mathsf{ch}}, m^*)$ be the challenge query of $\mathcal{A}$. If $j_{\mathsf{ch}} \neq j^*$, then $\mathcal{B}_3$ outputs a random bit $\widetilde{\beta}$ as his final output. Otherwise, $\mathcal{B}_3$ computes the challenge signature Σ^* as follows:

1. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
2. Set $M_0 \leftarrow 0^{|\sigma^*|}$ and $M_1 \leftarrow \sigma^*$. Then, send $(\mathsf{vk}_{i_b}, M_0, M_1)$ to the challenger for the IND-ID-CPA security game and receive a ciphertext $\widetilde{\mathsf{ct}}^*$.
3. Compute $\mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, 0^{|\langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|}; r_2^*)$ where r_2^* is an uniformly random value.
4. Compute $\pi^* \leftarrow \mathtt{Sim}_2(\mathsf{crs}, \mathsf{td}, \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle)$ where td is the trapdoor generated by $\mathtt{Sim}_1$.
5. Send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

We note that the identity vk_{i_b} was not queried to the $\mathsf{Extract}$ oracle since the pair (i_b, j^*) is not allowed to query the Revoke oracle. Thus, the tuple $(\mathsf{vk}_{i_b}, M_0, M_1)$ is the valid challenge query for the IND-ID-CPA security game. Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_3$ outputs $\widetilde{\beta} = 1$ if $b = \widetilde{b}$, and $\widetilde{\beta} = 0$ otherwise.

We consider the case that $j_{\mathsf{ch}} = j^*$. If $\beta = 0$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{IBE.Enc}(\mathsf{params}^{(j)}, \mathsf{vk}_{i_b}, 0^{|\sigma^*|})$ Therefore, $\mathcal{B}_3$ perfectly simulates Game 3. On the other hand, if $\beta = 1$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{IBE.Enc}(\mathsf{params}^{(j)}, \mathsf{vk}_{i_b}, \sigma^*)$, and thus $\mathcal{B}_3$ perfectly simulates Game 2. Since the guess of $j^* \in [1, T]$ and the behavior of $\mathcal{A}$ are independent, the probability that $j_{\mathsf{ch}} = j^*$ holds is $1/T$. Thus, we get $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| = 2T \cdot \mathsf{Adv}^{\mathit{ind\text{-}id\text{-}cpa}}_{\mathcal{IBE}, \mathcal{B}_3}(\lambda)$. □

Lemma 6.3.12. *There exists a* PPT *algorithm* $\mathcal{B}_4$ *such that* $\left|\Pr[\mathtt{Suc}_3] - \Pr[\mathtt{Suc}_4]\right| = 2T \cdot \mathsf{Adv}^{\mathit{key\text{-}priv}}_{\mathcal{IBE}, \mathcal{B}_4}(\lambda)$

Proof. Let $\mathcal{B}_4$ be an adversary for the key privacy of $\mathcal{IBE}$ and β be the challenge bit in the key privacy game. At the beginning of the game, $\mathcal{B}_4$ chooses the challenge bit b, and guesses a time period $j^* \in [1, T]$ uniform randomly. Then, $\mathcal{B}_4$ sets $\mathsf{params}^{(j^*)} \leftarrow \mathsf{params}$ where params is the system parameters given by the challenger. The rest of instance for the scheme Π_3 is generated by $\mathcal{B}_4$, however, only the common reference string crs is yielded by the simulator $\mathtt{Sim}_1$. $\mathcal{B}_4$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ and $\mathbf{gsk} = \{\mathsf{gsk}[i]\}_i$ to $\mathcal{A}$ where $\mathsf{gsk}[i] = (\mathsf{vk}_i, \mathsf{sk}_i, \mathsf{cert}_i)$. As with the algorithm $\mathcal{B}_3$ in Lemma 6.3.11, $\mathcal{B}_4$ can compute all the user signing keys $\mathbf{gsk}$ even he does not know the master secret key $\mathsf{msk}^{(j^*)}$. Moreover, for a revocation query (i, j) from $\mathcal{A}$, $\mathcal{B}_4$ can easily compute $\mathsf{grt}[i][j] = (\mathsf{dk}_{ij}, \mathsf{vk}_i)$ by using the master secret key $\mathsf{msk}^{(j)}$ on the condition that $j \neq j^*$. If $j = j^*$, $\mathcal{B}_4$ obtains

the decryption key dk_{ij^*} by querying vk_i to the $\mathsf{Extract}$ oracle of the key privacy game, and returns $\mathsf{grt}[i][j^*] = (\mathsf{dk}_{ij^*}, \mathsf{vk}_i)$ as the reply for $\mathcal{A}$. Let $(i_0, i_1, j_{\mathsf{ch}}, m^*)$ be the challenge query of $\mathcal{A}$. If $j_{\mathsf{ch}} \neq j^*$, then $\mathcal{B}_4$ outputs a random bit $\widetilde{\beta}$ as his final output. Otherwise, $\mathcal{B}_4$ computes the challenge signature Σ^* as follows:

1. Compute $\sigma^* \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{i_b}, m^*)$.
2. Choose a random key vk^* and set $M^* \leftarrow 0^{|\sigma^*|}$. Then, send $(\mathsf{vk}^*, \mathsf{vk}_{i_b}, M^*)$ to the challenger for the key privacy game and receive a ciphertext $\widetilde{\mathsf{ct}}^*$.
3. Compute $\mathsf{ct}^* \leftarrow \mathsf{PKE.Enc}(\mathsf{ek}_{\mathsf{PKE}}, 0^{|\langle \mathsf{vk}_{i_b}, \mathsf{cert}_{i_b} \rangle|}; r_2^*)$ where r_2^* is an uniformly random value.
4. Compute $\pi^* \leftarrow \mathtt{Sim}_2(\mathsf{crs}, \mathsf{td}, \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle)$ where td is the trapdoor generated by $\mathtt{Sim}_1$.
5. Send $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ to $\mathcal{A}$ as the challenge signature.

We note that the identity vk_{i_b} was not queried to the $\mathsf{Extract}$ oracle since the pair (i_b, j^*) is not allowed to query the Revoke oracle. Also, the identity vk^* was not queried to the $\mathsf{Extract}$ oracle with high probability since it is randomly chosen. Thus, the tuple $(\mathsf{vk}^*, \mathsf{vk}_{i_b}, M^*)$ is the valid challenge query for the key privacy game. Finally, when $\mathcal{A}$ terminates with $\widetilde{b}$, $\mathcal{B}_3$ outputs $\widetilde{\beta} = 1$ if $b = \widetilde{b}$, and $\widetilde{\beta} = 0$ otherwise.

We consider the case that $j_{\mathsf{ch}} = j^*$. If $\beta = 0$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{IBE.Enc}(\mathsf{params}^{(j)}, \mathsf{vk}^*, 0^{|\sigma^*|})$ Therefore, $\mathcal{B}_4$ perfectly simulates Game 4. On the other hand, if $\beta = 1$, $\widetilde{\mathsf{ct}}^*$ is represented as $\widetilde{\mathsf{ct}}^* = \mathsf{IBE.Enc}(\mathsf{params}^{(j)}, \mathsf{vk}_{i_b}, 0^{|\sigma^*|})$, and thus $\mathcal{B}_4$ perfectly simulates Game 3. Since the guess of $j^* \in [1, T]$ and the behavior of $\mathcal{A}$ are independent with each other, the probability that $j_{\mathsf{ch}} = j^*$ holds is $1/T$. Thus, we get $\left|\Pr[\mathtt{Suc}_3] - \Pr[\mathtt{Suc}_4]\right| = 2T \cdot \mathsf{Adv}^{\mathit{key\text{-}priv}}_{\mathcal{IBE},\mathcal{B}_4}(\lambda)$. □

In Game 4, the choice of the challenge bit b and the distribution of the challenge signature $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$ are independent. Thus, $\Pr[\mathtt{Suc}_4] = 1/2$ holds. Putting all together, we get

$$\begin{aligned}
\mathsf{Adv}^{\mathit{anon}}_{\Pi_3,\mathcal{A}}(\lambda, n, T) \leq & \sum_{i=0}^{3} \left|\Pr[\mathtt{Suc}_i] - \Pr[\mathtt{Suc}_{i+1}]\right| + \left|\Pr[\mathtt{Suc}_4] - 1/2\right| \\
= & \mathsf{Adv}^{\mathit{zk}}_{\mathcal{P}_{\widehat{L}},\mathcal{B}_1}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathit{ind\text{-}cpa}}_{\mathcal{PKE},\mathcal{B}_2}(\lambda) \\
& + 2T \cdot \mathsf{Adv}^{\mathit{ind\text{-}id\text{-}cpa}}_{\mathcal{IBE},\mathcal{B}_3}(\lambda) + 2T \cdot \mathsf{Adv}^{\mathit{key\text{-}priv}}_{\mathcal{IBE},\mathcal{B}_4}(\lambda).
\end{aligned}$$

Since the choice of the parameters n and T, and the adversary $\mathcal{A}$ is arbitrarily, our scheme Π_3 satisfies full anonymity if the underlying NIZK proof system $\mathcal{P}_{\widehat{L}}$ satisfies zero-

knowledgeness, the underlying PKE scheme $\mathcal{PKE}$ satisfies IND-CPA security, and the underlying IBE scheme $\mathcal{IBE}$ satisfies IND-ID-CPA security and key privacy. □

Theorem 6.3.9. *Scheme 3 satisfies traceability if the underlying NIZK proof system $\mathcal{P}_{\widehat{L}}$ satisfies soundness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security.*

Proof. Let $\mathcal{A}$ be an adversary for traceability of Π_3, and $(j^*, m^*, \Sigma^*, \mathsf{RU}^*)$ the output of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{trace}_{\Pi_3,\mathcal{A}}(\lambda, n, T)$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. Let i^* be the output of the $\mathsf{GS.Open}$ algorithm with an input (j^*, m^*, Σ^*). We consider the following cases:

I. $\langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle \notin \widehat{L}$,

II. $i^* = \perp$, **III.** $i^* \notin \mathsf{CU}$, and **IV.** $i^* \in \mathsf{RU}^*$.

If the output of the experiment $\mathsf{Exp}^{trace}_{\Pi_3,\mathcal{A}}(\lambda, n, T)$ is 1 (i.e., $\mathcal{A}$ succeeds in producing a forged signature), we can classify the type of the forgery as follows: (1) I, (2) $\neg$I $\wedge$ II, (3) $\neg$I $\wedge$ III, and (4) $\neg$I $\wedge$ IV.

Let E_ℓ be the event that $\mathcal{A}$ outputs a forged signature in Type ℓ where $1 \leq \ell \leq 4$. We estimate the each probability that the event E_ℓ happens in the following lemmas.

Lemma 6.3.13. *There exists a* PPT *algorithm $\mathcal{B}_1$ such that* $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}_{\widehat{L}},\mathcal{B}_1}(\lambda)$.

Proof. Let $\mathcal{B}_1$ be an adversary for soundness of $\mathcal{P}_{\widehat{L}}$. First, $\mathcal{B}_1$ receives the common reference string crs from the challenger. Next, $\mathcal{B}_1$ generates the rest of instance for the scheme Π_3, and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk_{SIG}}, \mathsf{ek_{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ and $\mathbf{grt} = \{\mathsf{grt}[i][j]\}$ to $\mathcal{A}$ where $\mathsf{grt}[i][j] \leftarrow (\mathsf{dk}_{ij}, \mathsf{vk}_i)$. Since $\mathcal{B}_1$ has all the user signing keys, he can easily simulate the $\mathsf{GS.Sign}$ oracle and the $\mathsf{Corrupt}$ oracle. Let $(j^*, m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. Then, $\mathcal{B}_1$ outputs $(\langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*)$ as a forgery for the soundness of $\mathcal{P}_{\widehat{L}}$.

When $\mathcal{A}$'s output $(j^*, m^*, \Sigma^*, \mathsf{RU}^*)$ is a forgery in Type 1, $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ and $\langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m \rangle \notin \widehat{L}$ hold. Therefore, $(\langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*)$ is the forgery for the soundness of $\mathcal{P}_{\widehat{L}}$. Thus, we have $\Pr[\mathsf{E}_1] \leq \mathsf{Adv}^{sound}_{\mathcal{P}_{\widehat{L}},\mathcal{B}_1}(\lambda)$. □

Lemma 6.3.14. *There exists a* PPT *algorithm $\mathcal{B}_2$ such that* $\Pr[\mathsf{E}_2] \leq \mathsf{Adv}^{unforge}_{\mathcal{SIG},\mathcal{B}_2}(\lambda)$.

Proof. Let $\mathcal{B}_2$ be an adversary for the EUF-CMA security of $\mathcal{SIG}$. First, $\mathcal{B}_2$ receives the verification key vk from the challenger of the EUF-CMA security game, and sets $\mathsf{vk_{SIG}} \leftarrow \mathsf{vk}$. Next, $\mathcal{B}_2$ generates the rest of instance for the scheme Π_3, except for the certificates cert_i where $i \in [1, n]$. In terms of the certificates, $\mathcal{B}_2$ sends vk_i to the Sign oracle of the scheme $\mathcal{SIG}$, and sets cert_i to be the signature which he received from the

oracle. $\mathcal{B}_2$ sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ and $\mathbf{grt} = \{\mathsf{grt}[i][j]\}$ to $\mathcal{A}$ where $\mathsf{grt}[i][j] \leftarrow (\mathsf{dk}_{ij}, \mathsf{vk}_i)$. Since $\mathcal{B}_2$ has all the user signing keys, he can easily simulate the $\mathsf{GS.Sign}$ oracle and the $\mathsf{Corrupt}$ oracle. Let $(j^*, m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. $\mathcal{B}_2$ decrypts the ciphertext ct^* by using the key $\mathsf{dk}_{\mathsf{PKE}}$ and obtains $\langle \mathsf{vk}^*, \mathsf{cert}^* \rangle$. Then, he outputs $(\mathsf{vk}^*, \mathsf{cert}^*)$ as a forged signature of $\mathcal{SIG}$.

When $\mathcal{A}$'s output $(j^*, m^*, \Sigma^*, \mathsf{RU}^*)$ is a forgery in Type 2, $\langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle \in \widehat{L}$ and $i^* = \perp$ hold. The equation $i^* = \perp$ means that there does not exist an index i such that $\mathsf{VLRGS.Verify}(\mathsf{gpk}, j^*, \{\mathsf{grt}[i][j^*]\}, m^*, \Sigma^*) = 0$. That is, it holds that $\mathsf{SIG.Verify}(\mathsf{vk}_i, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{ij^*}, \widetilde{\mathsf{ct}}^*)) = 0$ for all the indices i. Also, since $\langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle \in \widehat{L}$, there exist σ^* and r_1^* such that $\widetilde{\mathsf{ct}}^* = \mathsf{IBE.Enc}(\mathsf{params}^{(j^*)}, \mathsf{vk}^*, \sigma^*; r_1^*)$, and $\mathsf{SIG.Verify}(\mathsf{vk}^*, m^*, \sigma^*) = 1$ for the plaintext $\langle \mathsf{vk}^*, \mathsf{cert}^* \rangle$ of ct^*. From these facts, we get $\mathsf{vk}^* \neq \mathsf{vk}_i$ for any index i. Moreover, it satisfies that $\mathsf{SIG.Verify}(\mathsf{vk}_{\mathsf{SIG}}, \mathsf{vk}^*, \mathsf{cert}^*) = 1$ since $\langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle \in \widehat{L}$ holds. Therefore, $(\mathsf{vk}^*, \mathsf{cert}^*)$ is a forged signature of the digital signature scheme $\mathcal{SIG}$, and we get $\Pr[\mathsf{E}_2] \leq \mathsf{Adv}^{unforge}_{\mathcal{SIG}, \mathcal{B}_2}(\lambda)$. □

Lemma 6.3.15. *There exists a* PPT *algorithm* $\mathcal{B}_3$ *such that* $\Pr[\mathsf{E}_3] \leq n \cdot \mathsf{Adv}^{unforge}_{\mathcal{SIG}, \mathcal{B}_3}(\lambda)$.

Proof. Let $\mathcal{B}_3$ be an adversary for the EUF-CMA security of $\mathcal{SIG}$. First, $\mathcal{B}_3$ receives vk from the challenger of the EUF-CMA security game, and randomly chooses the index $i^* \in [1, n]$. Then, $\mathcal{B}_3$ sets $\mathsf{vk}_{i^*} \leftarrow \mathsf{vk}$. Next, $\mathcal{B}_3$ generates the rest of instance for the scheme Π_3, and sends $\mathsf{gpk} = (\mathsf{crs}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{ek}_{\mathsf{PKE}}, \{\mathsf{params}^{(j)}\}_j)$ and $\mathbf{grt} = \{\mathsf{grt}[i][j]\}$ to $\mathcal{A}$ where $\mathsf{grt}[i][j] \leftarrow (\mathsf{dk}_{ij}, \mathsf{vk}_i)$. If $\mathcal{A}$ sends (i, j, m) to the $\mathsf{GS.Sign}$ oracle and it holds $i \neq i^*$, $\mathcal{B}_3$ easily computes the corresponding signature since $\mathcal{B}_3$ knows $\mathsf{gsk}[i]$ for all the users $i \neq i^*$. However, $\mathcal{B}_3$ does not know the signing key sk_{i^*} corresponding to vk_{i^*}. Thus, $\mathcal{B}_3$ does not know i^*'s user signing key. Then, if $\mathcal{A}$ sends (i^*, j, m) to the $\mathsf{GS.Sign}$ oracle, $\mathcal{B}_3$ sends m to the Sign oracle of $\mathcal{SIG}$ and receives a signature σ. Furthermore, he computes $(\widetilde{\mathsf{ct}}, \mathsf{ct})$ and π according to the $\mathsf{GS.Sign}$ algorithm, and returns $\Sigma = (\widetilde{\mathsf{ct}}, \mathsf{ct}, \pi)$ to $\mathcal{A}$. We note that the signing key sk_{i^*} is not needed to generate a NIZK proof π. If $\mathcal{A}$ sends i to the $\mathsf{Corrupt}$ oracle and $i = i^*$ holds, then $\mathcal{B}_3$ outputs $\perp$. Otherwise, if $i \neq i^*$, then $\mathcal{B}_3$ returns $\mathsf{gsk}[i]$. Let $(j^*, m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$ where $\Sigma^* = (\widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, \pi^*)$. If $\mathsf{VLRGS.Open}(\mathsf{gpk}, j^*, \mathbf{grt}, m^*, \Sigma^*) \neq i^*$, $\mathcal{B}_4$ outputs $\perp$. If $\mathsf{VLRGS.Open}(\mathsf{gpk}, j^*, \mathbf{grt}, m^*, \Sigma^*) = i^*$, then $\mathcal{B}_3$ decrypts $\widetilde{\mathsf{ct}}^*$ by using $\mathsf{dk}_{i^*j^*}$ and obtains the decryption result σ^*, that is, $\sigma^* \leftarrow \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)$. Then, $\mathcal{B}_3$ outputs (m^*, σ^*) as a forged signature.

We consider the case that $\mathcal{B}_3$ succeeded in guessing the index i^*, that is, $\mathsf{VLRGS.Open}(\mathsf{gpk}, j^*, \mathbf{grt}, m^*, \Sigma^*) = i^*$ holds. In this case, it holds that $\mathsf{VLRGS.Verify}(\mathsf{gpk}, j^*, \mathsf{grt}[i^*][j^*], m^*, \Sigma^*) = 0$. That is, either $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek}_{\mathsf{PKE}}, \mathsf{vk}_{\mathsf{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 0$ or $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)) = 1$ holds. Since Σ^* is a

valid signature, $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ holds. Therefore, we can say that the latter condition $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)) = \mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \sigma^*) = 1$ holds. Also, due to the success condition of $\mathcal{A}$, $(\cdot, \cdot, m^*)$ is not queried to the $\mathsf{GS.Sign}$ oracle. Thus, m^* is not queried to the Sign oracle and (m^*, σ^*) is a valid forged signature. The probability of succeeding in guessing the index i^* is $1/n$ since the guess of $i^* \in [1, n]$ and the behavior of $\mathcal{A}$ are independent with each other. Then, we have $(1/n) \cdot \Pr[\mathsf{E}_3] \leq \mathsf{Adv}^{unforge}_{\mathcal{SIG}, \mathcal{B}_3}(\lambda)$. □

Lemma 6.3.16. $\Pr[\mathsf{E}_4] = 0$ *holds.*

Proof. Let $(j^*, m^*, \Sigma^*, \mathsf{RU}^*)$ be the output of $\mathcal{A}$, and i^* be the result of the the $\mathsf{VLRGS.Open}$ algorithm with the input (j^*, m^*, Σ^*). If $i^* \in \mathsf{RU}^*$, then $\mathsf{grt}[i^*][j^*] \in \mathsf{RL}^*$. Also, due to the success condition, $\mathsf{VLRGS.Verify}(\mathsf{gpk}, j^*, \mathsf{RL}^*, m^*, \Sigma^*) = 1$ holds, that is, $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ and $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)) = 0$ hold. Moreover, since the opening result is i^*, we have $\mathsf{VLRGS.Verify}(\mathsf{gpk}, j^*, \{\mathsf{grt}[i^*][j^*]\}, m^*, \Sigma^*) = 0$. That is, either $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 0$ or $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)) = 1$ holds. However, both cases contradict the success conditions $\mathsf{ZK.Verify}(\mathsf{crs}, \langle \mathsf{ek_{PKE}}, \mathsf{vk_{SIG}}, \mathsf{params}^{(j^*)}, \widetilde{\mathsf{ct}}^*, \mathsf{ct}^*, m^* \rangle, \pi^*) = 1$ and $\mathsf{SIG.Verify}(\mathsf{vk}_{i^*}, m^*, \mathsf{IBE.Dec}(\mathsf{dk}_{i^*j^*}, \widetilde{\mathsf{ct}}^*)) = 0$ Thus, we get $\Pr[\mathsf{E}_4] = 0$. □

Putting all together, we get

$$\begin{aligned}
\mathsf{Adv}^{trace}_{\Pi_3, \mathcal{A}}(\lambda, n, T) &= \Pr[\ \mathsf{Exp}^{trace}_{\Pi_3, \mathcal{A}}(\lambda, n, T) = 1\ ] \\
&= \Pr[\ \mathsf{E}_1 \vee \mathsf{E}_2 \vee \mathsf{E}_3 \vee \mathsf{E}_4\ ] \\
&= \Pr[\mathsf{E}_1] + \Pr[\mathsf{E}_2] + \Pr[\mathsf{E}_3] + \Pr[\mathsf{E}_4] \\
&\leq \mathsf{Adv}^{sound}_{\mathcal{P}_{\widehat{L}}, \mathcal{B}_1}(\lambda) + \mathsf{Adv}^{unforge}_{\mathcal{SIG}, \mathcal{B}_2}(\lambda) + n \cdot \mathsf{Adv}^{unforge}_{\mathcal{SIG}, \mathcal{B}_3}(\lambda).
\end{aligned}$$

Since the choice of the parameters n and T, and the adversary $\mathcal{A}$ is arbitrarily, our scheme Π_3 satisfies traceability if the underlying NIZK proof system $\mathcal{P}_{\widehat{L}}$ satisfies soundness and the underlying digital signature scheme $\mathcal{SIG}$ satisfies EUF-CMA security. □

Chapter 7

Group Signature with Deniability

Group signature allows for users to anonymously prove their membership, but in the case of incidents (e.g., crimes), the identity of the signer can be revealed by the opener. Such a property seems quite useful for protecting the users' privacy and tracing malicious users simultaneously. However, for some situations, this property is insufficient.

For example, assume that the police needs to know whether a suspect was in a specific building at the time of a crime, and the entrance and exit control for the building is managed using a group signature scheme. A naive way to check this is to ask the authority (i.e., the manager of the building) to just reveal the signer identities of all the signatures that were used for the authentication within that specified time period. This is what can be done by standard group signature. Obviously, this results in a serious violation of the privacy of the innocent users who have entered the building in that time period.

To avoid this situation, it is further required that a group signature scheme provides functionality for generating yet another kind of digital evidence which only proves that for a given signature and identity of a suspect, the signer is not the suspect.

In this chapter, we introduce group signature that provides the above-mentioned functionality, called deniable group signature [68, 69]. In addition to all the functionalities of standard group signature, deniable group signature provides another functionality that the opener can generate a denial proof that proves non-ownership of a signature. More precisely, for a given signature and an identity of a user, the opener can generate a proof of the fact that the actual signer is not that particular user (if this is the case).

7.1 Syntax

Deniable group signature is a natural extension of the Bellare-Shi-Zhang (BSZ) model [17] introduced in Chapter 4. In a deniable group signature scheme, we require that for the signature Σ of a message msg and a user j, the opener can establish a proof that the open

result is not j. Hence in addition to the standard functionality of group signature, two new algorithms, GS.DOpen and GS.DJudge are introduced to the BSZ model. The opener produces a denial proof by using the GS.DOpen algorithm and validity of the proof can be checked by the GS.DJudge algorithm.

Formally, a deniable group signature scheme $\mathcal{D}$-$\mathcal{GS}$ consists of the algorithms (GS.GGen, GS.UGen, GS.Join/GS.Issue, GS.Sign, GS.Verify, GS.Open, GS.Judge, GS.DOpen, GS.DJudge) where the GS.DOpen and the GS.DJudge algorithms are defined below, and the other algorithms are defined in Section 4.1.

GS.DOpen**:** The denial opening algorithm takes gpk, j, ok, msg, Σ, and reg, and returns η_j, where j is a user identity, and η_j is a proof that user j did not compute Σ.

GS.DJudge**:** The denial judgement algorithm takes gpk, j, upk_j, msg, Σ, and η_j, and returns 1 or 0.

We note that $\mathsf{GS.Judge}(\mathsf{gpk}, i, \mathsf{upk}_i, \mathsf{msg}, \Sigma, \tau) = 0$ does not imply that the signature Σ is not generated by the user i. For example, if a proof τ is not generated honestly, the GS.Judge algorithm outputs 0. Similarly, $\mathsf{GS.DJudge}(\mathsf{gpk}, j, \mathsf{upk}_j, \mathsf{msg}, \Sigma, \eta_j) = 0$ does not imply that the signature Σ is generated by the user j. For example, if a proof η_j is not generated honestly, the GS.DJudge algorithm outputs 0.

7.2 Security

As in the case of standard group signature, we provide definitions of correctness, anonymity, non-frameability, traceability, and opening soundness. The security model is extended from that of standard group signature defined in Section 4.2 and therefore, is almost the same except for anonymity.

Firstly, we define a new oracle, the DOpen oracle.

$\mathsf{DOpen}(\cdot,\cdot,\cdot)$**:** This deniable opening oracle takes as input a user identity j, msg and Σ, and returns $\eta_j \leftarrow \mathsf{GS.DOpen}(\mathsf{gpk}, j, \mathsf{ok}, \mathsf{msg}, \Sigma, \mathsf{reg})$ if $(\mathsf{msg}, \Sigma) \notin \mathsf{GSet} \vee j \notin \mathsf{ISet}$ and $\perp$ otherwise.

Secondly, we define the security requirements by using the DOpen oracle and the oracles defined in Section 4.2.

Correctness of standard group signature is required for ensuring that any honestly generated group signature is valid (i.e., it is accepted by the GS.Verify algorithm), and the GS.Open algorithm correctly identifies its actual signer and a proof generated by the algorithm is accepted by the GS.Judge algorithm.

In addition to this, we require that a proof of an arbitrary user j who is not the actual signer i, generated by the $\mathsf{GS.DOpen}$ algorithm is accepted by the $\mathsf{GS.DJudge}$ algorithm. Formally, correctness for deniable group signature is defined as follows.

Definition 7.2.1 (Correctness)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for D-GS scheme $\mathcal{D}$-$\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{corr}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{corr}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda): \quad & (\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{HU} \leftarrow \emptyset;\ \mathsf{CU} \leftarrow \emptyset \\
& (i, \mathsf{msg}, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathsf{AddU}(\cdot),\mathsf{RReg}(\cdot)}(\mathsf{gpk}) \\
& \text{If } i \notin \mathsf{HU},\ \text{return } 0 \\
& \Sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, \mathsf{msg}) \\
& (\widetilde{i}, \tau) \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{msg}, \Sigma, \mathsf{reg}) \\
& j \leftarrow \mathcal{A}_2^{\mathsf{AddU}(\cdot),\mathsf{RReg}(\cdot)}(\Sigma, \widetilde{i}, \tau, \mathsf{st}) \\
& \text{If } i = j \ \vee\ j \notin \mathsf{HU},\ \text{return } 0 \\
& \eta_j \leftarrow \mathsf{GS.DOpen}(\mathsf{gpk}, j, \mathsf{upk}_j, \mathsf{ok}, \mathsf{msg}, \Sigma, \mathsf{reg}) \\
& \text{Output 1 if the following holds :} \\
& \qquad \mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{msg}, \Sigma) = 0 \ \vee\ i \neq \widetilde{i} \\
& \qquad\qquad \vee\ \mathsf{GS.Judge}(\mathsf{gpk}, i, \mathsf{upk}_i, \mathsf{msg}, \Sigma, \tau) = 0 \\
& \qquad\qquad\qquad \vee\ \mathsf{GS.DJudge}(\mathsf{gpk}, j, \mathsf{upk}_j, \mathsf{msg}, \Sigma, \eta_j) = 0 \\
& \text{Otherwise return } 0
\end{aligned}
$$

We say that $\mathcal{D}$-$\mathcal{GS}$ is correct if the advantage

$$\mathsf{Adv}^{corr}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{corr}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

Anonymity of standard group signature is required so that an adversary who can corrupt the issuer and malicious users cannot extract any user information from group signatures of honest users in the case when the adversary is able to access the Open oracle.

In anonymity of deniable group signature, the adversary can also access the DOpen oracle. As mentioned above, the adversary can even query the challenge signature to the DOpen oracle except for querying the challenge users i_0 and i_1. Then, anonymity is guaranteed even if denial proofs for all the users except i_0 and i_1 are provided for the challenge group signature.[1] Formally, anonymity for deniable group signature is defined as follows.

[1]Recall that we exclude the case that an adversary requests a denial proof of either i_0 or i_1 for the challenge signature, since this trivially breaks the anonymity. See the definition of DOpen oracle above.

Definition 7.2.2 (Anonymity). *Let $\mathcal{A}$ be an adversary for a D-GS scheme $\mathcal{D}$-$\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{anon}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{anon}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda):\quad & b \xleftarrow{\$} \{0,1\};\ (\mathsf{gpk},\mathsf{ik},\mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{CU} \leftarrow \emptyset;\ \mathsf{HU} \leftarrow \emptyset \\ & \widetilde{b} \leftarrow \mathcal{A}^{\mathsf{Corrupt}(\cdot,\cdot),\mathsf{SndToU}(\cdot),\mathsf{WReg}(\cdot,\cdot),\mathsf{USK}(\cdot),\mathsf{Open}(\cdot,\cdot),\mathsf{DOpen}(\cdot,\cdot,\cdot),\mathsf{Ch}(b,\cdot,\cdot,\cdot)}(\mathsf{gpk},\mathsf{ik}) \\ & \text{Return 1 if } b = \widetilde{b}, \text{ otherwise return 0}\end{aligned}$$

We say that $\mathcal{D}$-$\mathcal{GS}$ satisfies anonymity if the advantage

$$\mathsf{Adv}^{anon}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}} = \left|\ \Pr[\ \mathsf{Exp}^{anon}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ] - 1/2\ \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

Traceability of standard group signature is required for ensuring that an adversary, who can corrupt the opener, and malicious users cannot produce a valid group signature whose opening result is not valid (i.e., an invalid identity) or opening proof is not accepted by the GS.Judge algorithm.

In the case of deniable group signature, it is also required that the adversary should not be able to produce a valid group signature whose opening proof is accepted by the GS.Judge algorithm, but for the same user, the denial opening proof is accepted by the GS.DJudge algorithm. The formal definition of traceability for deniable group signature is given as follows.

Definition 7.2.3 (Traceability). *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary for a D-GS scheme $\mathcal{D}$-$\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{trace}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned}\mathsf{Exp}^{trace}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda):\quad & (\mathsf{gpk},\mathsf{ik},\mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{CU} \leftarrow \emptyset;\ \mathsf{HU} \leftarrow \emptyset \\ & (\mathsf{msg},\Sigma,\mathsf{st}) \leftarrow \mathcal{A}_1^{\mathsf{Corrupt}(\cdot,\cdot),\mathsf{SndToI}(\cdot),\mathsf{AddU}(\cdot),\mathsf{RReg}(\cdot),\mathsf{USK}(\cdot)}(\mathsf{gpk},\mathsf{ok}) \\ & (i,\tau) \leftarrow \mathsf{GS.Open}(\mathsf{gpk},\mathsf{ok},\mathsf{msg},\Sigma,\mathsf{reg}) \\ & \eta_i \leftarrow \mathcal{A}_2^{\mathsf{Corrupt}(\cdot,\cdot),\mathsf{SndToI}(\cdot),\mathsf{AddU}(\cdot),\mathsf{RReg}(\cdot),\mathsf{USK}(\cdot)}(i,\tau,\mathsf{st}) \\ & \text{Return 1 if the following two conditions hold :} \\ & \quad \mathsf{GS.Verify}(\mathsf{gpk},\mathsf{msg},\Sigma) = 1 \\ & \quad i = 0\ \vee\ \mathsf{GS.Judge}(\mathsf{gpk},i,\mathsf{upk}_i,\mathsf{msg},\Sigma,\tau) = 0 \\ & \qquad\qquad \vee\ \mathsf{GS.DJudge}(\mathsf{gpk},i,\mathsf{upk}_i,\mathsf{msg},\Sigma,\eta_j) = 1 \\ & \text{Otherwise return 0}\end{aligned}$$

We say that $\mathcal{D}$-$\mathcal{GS}$ satisfies traceability if the advantage

$$\mathsf{Adv}^{trace}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{trace}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

Non-frameability of standard group signature is required so that an adversary who can corrupt the issuer, the opener, and malicious users except one honest user cannot produce a valid group signature of the honest user and its opening proof, which is accepted by the $\mathsf{GS.Judge}$ algorithm.

In non-frameability of deniable group signature, it is required that the adversary should not be able to forge a valid group signature whose denial opening proof for the honest user j is not accepted by the $\mathsf{GS.DJudge}$ algorithm.

Definition 7.2.4 (Non-Frameability)**.** *Let $\mathcal{A}$ be an adversary for a D-GS scheme $\mathcal{D}$-$\mathcal{GS}$. We define an experiment $\mathsf{Exp}^{\textit{non-frame}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$\mathsf{Exp}^{\textit{non-frame}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda)$:

$(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda);\ \mathsf{CU} \leftarrow \emptyset;\ \mathsf{HU} \leftarrow \emptyset$

$(\mathsf{msg}, \Sigma, j, \tau) \leftarrow \mathcal{A}^{\mathsf{SndToU}(\cdot),\mathsf{Sign}(\cdot,\cdot),\mathsf{WReg}(\cdot,\cdot),\mathsf{USK}(\cdot),\mathsf{Corrupt}(\cdot,\cdot)}(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$

$\eta_j \leftarrow \mathsf{GS.DOpen}(\mathsf{gpk}, j, \mathsf{upk}_j, \mathsf{ok}, \mathsf{msg}, \Sigma, \mathsf{reg})$

Return 1 if all of the following hold :

- $j \in \mathsf{HU}$
- $\mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{msg}, \Sigma) = 1$
- $\mathsf{GS.Judge}(\mathsf{gpk}, j, \mathsf{upk}_j, m, \Sigma, \tau) = 1$
  $\vee\ \mathsf{GS.DJudge}(\mathsf{gpk}, j, \mathsf{upk}_j, \mathsf{msg}, \Sigma, \eta_j) = 0$
- $\mathcal{A}$ did not query $\mathsf{USK}(j)$ and $\mathsf{Sign}(j, \mathsf{msg})$

Otherwise return 0

We say that $\mathcal{D}$-$\mathcal{GS}$ satisfies non-frameability if the advantage

$$\mathsf{Adv}^{\textit{non-frame}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{\textit{non-frame}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = 1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$.

Opening soundness of standard group signature is required so that an adversary who can corrupt the issuer, the opener, and malicious users cannot produce a valid group signature and its opening proofs for i and j, which are both accepted by the $\mathsf{GS.Judge}$ algorithm.

In opening soundness of deniable group signature, it is also required that the adversary should not be able to produce a valid group signature and its denial opening proof for the actual signer which is accepted by the $\mathsf{GS.DJudge}$ algorithm.

Definition 7.2.5 (Opening Soundness)**.** *Let $\mathcal{A}$ be an adversary for a D-GS scheme $\mathcal{D}$-$\mathcal{GS}$.*

We define an experiment $\mathsf{Exp}^{\textit{open-sound}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda)$ *as follows.*

$$
\begin{aligned}
\mathsf{Exp}^{\textit{open-sound}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda): \quad & (\mathsf{gpk},\mathsf{ik},\mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^\lambda) \\
& (i,j,\tau_i,\tau_j,\eta_i,\mathsf{msg},\Sigma) \leftarrow \mathcal{A}^{\mathsf{Corrupt}(\cdot),\mathsf{WReg}(\cdot,\cdot)}(\mathsf{gpk},\mathsf{ok},\mathsf{ik}) \\
& \text{Return 1 if all of the following hold :} \\
& \quad \mathsf{GS.Verify}(\mathsf{gpk},\mathsf{msg},\Sigma)=1 \\
& \quad \mathsf{GS.Judge}(\mathsf{gpk},i,\mathsf{upk}_i,\mathsf{msg},\Sigma,\tau_i)=1 \\
& \quad \{i\neq j \wedge \mathsf{GS.Judge}(\mathsf{gpk},j,\mathsf{upk}_j,\mathsf{msg},\Sigma,\tau_j)=1\} \\
& \qquad\qquad \vee\ \mathsf{GS.DJudge}(\mathsf{gpk},i,\mathsf{upk}_i,\mathsf{msg},\Sigma,\eta_i)=1 \\
& \text{Otherwise return 0}
\end{aligned}
$$

We say that $\mathcal{D}$-$\mathcal{GS}$ *satisfies opening soundness if the advantage*

$$\mathsf{Adv}^{\textit{open-sound}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda) = \Pr[\ \mathsf{Exp}^{\textit{open-sound}}_{\mathcal{D}\text{-}\mathcal{GS},\mathcal{A}}(\lambda)=1\ ]$$

is negligible for any PPT adversary $\mathcal{A}$*.*

7.3 Generic Construction and Its Limitation

We show that deniable group signature can be constructed by applying the technique to generically construct (standard) group signature presented by Bellare, Shi, and Zhang [17]. Then, we explain the difficulty of instantiating an efficient scheme even though a generic construction is given.

In the BSZ construction, each user i has a key pair $(\mathsf{vk}_i,\mathsf{sk}_i)$ of a signature scheme. The issuer also has a key pair $(\mathsf{vk}_{\mathsf{SIG}},\mathsf{sk}_{\mathsf{SIG}})$ of a signature scheme, and the opener has a key pair $(\mathsf{ek}_{\mathsf{PKE}},\mathsf{dk}_{\mathsf{PKE}})$ of a public key encryption scheme. To issue a signing key to a user i, the issuer signs the message (i,vk_i) using his key $\mathsf{sk}_{\mathsf{SIG}}$ and sends the signature cert_i to the user i. When generating a signature on a message m, a signer i firstly produces a signature σ on the message m under vk_i. Then, to make this verifiable without losing anonymity, the user makes an encryption C of $(i,\mathsf{vk}_i,\mathsf{cert}_i,\sigma)$ using $\mathsf{ek}_{\mathsf{PKE}}$ and also makes a NIZK proof π which proves that cert_i is a valid certificate on (i,vk_i), i.e., $\mathsf{SIG.Verify}(\mathsf{vk}_{\mathsf{SIG}},(i,\mathsf{vk}_i),\mathsf{cert}_i)=1$, σ is a valid signature on m, and the ciphertext C is correctly generated. The opener can identify the signer by decrypting C using $\mathsf{dk}_{\mathsf{PKE}}$. Moreover, the opener produces a NIZK proof τ which proves that C decrypts to $(i,\mathsf{vk}_i,\mathsf{cert}_i,\sigma)$ under $\mathsf{dk}_{\mathsf{PKE}}$.

We can add deniability to the BSZ construction as follows. The opener produces a NIZK proof η_j which proves that C decrypts to $(i,\mathsf{vk}_i,\mathsf{cert}_i,\sigma)$ under $\mathsf{dk}_{\mathsf{PKE}}$, and cert_i is

not a valid certificate on (j, vk_j), i.e., $\mathsf{SIG.Verify}(\mathsf{vk}_{\mathsf{SIG}}, (j, \mathsf{vk}_j), \mathsf{cert}_i) \neq 1$. Although this denial proof can be constructed by using general NIZK proofs [22], it is quite inefficient.

Therefore, the next attempt is to add deniability to an efficient group signature scheme (e.g., the modified Groth scheme [100]) by using an efficient NIZK proof (e.g., the Groth-Sahai proofs [63]). Unfortunately, this type of language, i.e., inequality statements is not compatible with the Groth-Sahai proofs, especially the NIZK ones.

7.4 Concrete Scheme

In this section, we provide a concrete construction of a deniable group signature scheme. We give a description of the scheme in Section 7.4.1 and prove its security in Section 7.4.2.

7.4.1 Description

In this section, we describe a deniable group signature scheme which is extended the modified Groth scheme described in Section 4.3.

In a deniable group signature scheme, the opener needs to issue a denial proof which proves that the user j is not the actual signer without revealing user i itself. It seems this functionality can be achieved by simple modifications where the opener makes a NIZK proof for the statement $e(\sigma, v_j G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) \neq e(G, G)$. However, in fact, this does not work by using the Groth-Sahai proof system since its languages are limited, especially inequality statements are not compatible with the Groth-Sahai proof system. Thus, to prove $i \neq j$, we employ the technique for proving an inequality statement introduced by e.g., [101]. The technique is that to prove $a \neq b$, a prover picks $\ell \in \mathbb{Z}_p$ randomly, and sets $c \leftarrow (a/b)^{\ell}$, and then a verifier checks $c \neq 1$ and the knowledge of the value ℓ.

We give our scheme in Figure 7.1. Here, we describe only the $\mathsf{GS.DOpen}$ and the $\mathsf{GS.DJudge}$ algorithms since the other schemes are identical those of the modified Groth scheme. In the $\mathsf{GS.DOpen}$ algorithm, the statement $v_i \neq v_j$ is proved, which indicates that a user j is not the actual signer i. More precisely, the opener takes random $\ell \leftarrow \mathbb{Z}_p$ and set $c = \tau_\ell \cdot (\tau'_\ell)^{-1}$ where $\tau_\ell = v_i^\ell$ and $\tau'_\ell = v_j^\ell$. Moreover, the opener computes the proof ϕ, via the NIZK proof system $\mathsf{P}_{\mathrm{NIZK}}$, which shows the knowledge of the opening proof $(i, (\sigma, \tau_F, \tau_H))$, v_i, τ_ℓ, and τ'_ℓ which satisfy

$$e(F, \tau_F) = e(y_1, G) \ \wedge \ e(H, \tau_H) = e(y_2, G) \ \wedge \ \sigma\tau_F\tau_H = y_3$$
$$\wedge \ e(\sigma, v_i G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) = e(G, G) \ \wedge \ e(v_i, \tau'_\ell) = e(v_j, \tau_\ell) \ \wedge \ c = \tau_\ell \cdot (\tau'_\ell)^{-1}.$$

The first four equations demonstrate that i is the actual signer of Σ, and the fifth equation demonstrates that the discrete logarithm of τ_ℓ and that of τ'_ℓ are the same. In the

GS.DJudge algorithm, a verifier checks the validity of the proof and whether $c \neq 1$.

$\mathsf{GS.DOpen}(\mathsf{gpk}, j, \mathsf{ok}, \mathsf{reg}, m, \Sigma)$: $(i, (\sigma, \tau_F, \tau_H)) \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \Sigma)$ If $(i, (\sigma, \tau_F, \tau_H)) = (0, \perp)$, return $\perp$ $\ell \leftarrow \mathbb{Z}_p; \tau_\ell \leftarrow v_i^\ell; \tau'_\ell \leftarrow v_j^\ell; c \leftarrow \tau_\ell \cdot (\tau'_\ell)^{-1}$ $\phi \leftarrow \mathsf{P}_{\mathrm{NIZK}}(\mathsf{crs}, (\mathsf{gpk}, y, v_j, c), (\sigma, \tau_F, \tau_H, v_i, \tau_\ell, \tau'_\ell))$ Return (ϕ, c)
$\mathsf{GS.DJudge}(\mathsf{gpk}, j, \mathsf{reg}, m, \Sigma, (\phi, c))$: Return 1 if the following hold: $\mathsf{GS.Verify}(\mathsf{gpk}, \mathsf{reg}, m, \Sigma) = 1$ $1 = \mathsf{V}_{\mathrm{NIZK}}(\mathsf{crs}, (\mathsf{gpk}, y, v_j, c), \phi)$ $c \neq 1$ else return 0

Figure 7.1: The GS.DOpen and the GS.DJudge Algorithms of Our Scheme

Since the proposed deniable group signature scheme is exactly the same as the modified Groth scheme [100], except for the algorithms for generating and verifying denial proofs, the efficiency of the other algorithms is identical to that of the modified Groth scheme. In the following, we estimate the size of a denial proof.

We modify the equations above to produce a zero-knowledge proof as follows. The first equation $e(F, \tau_F) = e(y_1, G)$ is changed to two equations $e(F, \tau_F) \cdot e(y'_1, G^{-1}) = 1 \ \wedge \ y'_1 \cdot y_1^{-1} = 1$ where y'_1 is a new witness. In the same way, $e(H, \tau_H) = e(y_2, G)$ is changed to $e(H, \tau_H) \cdot e(y'_2, G^{-1}) = 1 \ \wedge \ y'_2 \cdot y_2^{-1} = 1$ where y'_2 is a witness. Moreover, $e(\sigma, v_i G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) = e(G, G)$ is changed to $e(\sigma, v_i G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) \cdot e(G', G^{-1}) = 1 \ \wedge \ G' \cdot G^{-1} = 1$ where G' is a witness.

Thus, a denial proof consists of 6 commitments and 3 new commitments, which consist of 3 group elements each, and 4 pairing product equations, which consist of 9 group elements but a linear equation consisting of 3 group elements, and 5 multi-scalar multiplication equations, which consist of 9 group elements each. Therefore, the denial proof consists of 96 group elements in total.

The size of a denial proof seems to be adequate from the following reasons. In our scheme, the opener proves that "a signer is a member of the group," and "the signer does not generate a group signature without revealing the signer itself." The first part of denial proof can be regarded as functionality of standard group signature, and the second part of denial proof can be regarded as the revocation functionality, which proves that a signer is not revoked without revealing the signer itself. Since revocable group signature schemes, e.g., [81, 80, 12, 89], require approximately 50-100 group elements in addition to the membership proof part, it seems reasonable that denial proof requires 96 group elements in total.

Remark. In the modified Groth scheme, we can confirm that a user i is the actual signer by checking the equation $e(\sigma, v_i G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) = e(G, G)$. Thus, the statement $e(\sigma, v_j G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) \neq e(G, G)$ indicates that a user j is not the signer. Therefore, we think that it is natural to prove this statement by using the technique [101] for proving an inequality statement. However, this does not work well as follows.

If we prove the inequality statement $e(\sigma, v_j G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) \neq e(G, G)$, we transform this statement into the equal statement $\{e(\sigma, v_i g^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})})/e(\sigma, v_j g^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})})\}^{\ell} = c$ for a random value ℓ. Then, we prove this new statement, and a verifier checks whether $c \neq 1$. If we provide a NIZK proof for the equation $\{e(\sigma, v_i g^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})})/e(\sigma, v_j g^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})})\}^{\ell} = c$ in the Groth-Sahai proof system, we need to find $\tilde{g}, \hat{g} \in \mathbb{G}$ such that $c = e(\tilde{g}, \hat{g})$. However, it is hard to find such elements because of the pairing inversion problem [56].

In the proposed scheme, all the witnesses are base group elements. Therefore, we can avoid to solve the pairing inversion problem since all the target group elements are decomposed into known base group elements.

Blazy et al. [20] proposed a new NIZK proof of non-membership based on the Groth-Sahai proof. However, for the same reason, the NIZK proof is not directly applicable to prove deniability in the modified Groth scheme. Nguyen [93] proposed a revocable group signature scheme by employing accumulators, where a user specific value is accumulated to a constant-size value and a user whose value is accumulated can prove that the value is accumulated without revealing the value. Since Li et al. [79] and Damgård et al. [45] extended this membership proofs to non-membership proofs where a user can prove that the value is not accumulated without revealing the value, this technique might be applied to realize the deniability. However, as in the Nguyen group signature scheme, a signer is required to compute an updated accumulated value and this cost depends on the number of (revoked) users, and therefore this solution does not yield an efficient construction.

7.4.2 Security Proof

In this section, we give proofs of security requirements, correctness, anonymity, non-frameability, traceability, and opening soundness defined in Section 7.2.

In the proof of anonymity, an adversary is allowed to issue DOpen queries even for the challenge group signature. Since the opening query for the challenge group signature is not allowed in the anonymity game of the modified Groth scheme, we cannot use the challenger of the modified Groth scheme. That is, the simulator needs to respond DOpen queries for the challenge group signature without knowing its opening result. The detail is given in Theorem 7.4.1. Except for from Game 7 to Game 8, translations between games are almost same as those of the modified Groth scheme [100].

In the proofs of other security requirements, we can directly break the modified Groth

scheme by using an adversary who breaks our scheme if the winning conditions are independent of deniability. In the deniability-related parts, we can also give a proof in a similar way by assuming the security of building blocks.

Formally, the following theorems hold.

Theorem 7.4.1. *The proposed group signature scheme satisfies anonymity if the DLIN assumption holds in $\mathbb{G}$, the one-time signature scheme is strong existential unforgeable, and the hash function is universal one-way.*

Proof. Let $\mathcal{A}_{anon}$ be an adversary that has the advantage ϵ in the anonymity game. Now, we gradually modify the game played by $\mathcal{A}_{anon}$. In the following S_i denotes the event that $\mathcal{A}_{anon}$ successfully guesses the bit $b = b'$ interacting with the environment of Game i.

[Game 0]: Game 0 is identical to the game in the definition of anonymity. In this game, we have $\Pr[S_0] = 1/2 + \epsilon$.

[Game 1]: We modify the behavior of the Open oracle and the DOpen oracle as follows. If they receive a valid group signature which reuses the verification key $\mathsf{vk}^*_{\mathrm{sots}}$ of the challenge signature Σ^*, the game aborts. By the strong existential unforgeability of one-time signature scheme, this modification does not change the success probability of $\mathcal{A}_{anon}$ with more than negligible amount, that is, we have that $|\Pr[S_0] - \Pr[S_1]|$ is negligible.

[Game 2]: We further modify the Open oracle and the DOpen oracle to abort when a queried group signature contains $\mathsf{vk}_{\mathrm{sots}}$ where $\mathcal{H}(\mathsf{vk}_{\mathrm{sots}}) = \mathcal{H}(\mathsf{vk}^*_{\mathrm{sots}})$. By the universal one-wayness of the hash function, this modification does not change the success probability of $\mathcal{A}_{anon}$ with more than negligible amount, that is, we have that $|\Pr[S_1] - \Pr[S_2]|$ is also negligible.

[Game 3]: Now, we modify the way to generate the public key for the tag-based encryption. We set $K = F^{\zeta}, L = H^{\xi}$, and store ζ and ξ. This modification does not vary the behavior of the adversary, that is, $\Pr[S_2] = \Pr[S_3]$.

[Game 4]: We then modify how the Open oracle and the DOpen oracle obtain a signer identity i. Until Game 3, when the Open oracle and the DOpen oracle receive a query, they first extract a witness (b, v, σ) from the proof π by using the extraction key xk and search for i such that $\mathsf{reg}[i] = v$. However, in Game 4, the Open oracle and the DOpen oracle search for i such that $e(\sigma, v_i G^{\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}) = e(G, G)$ going through reg. This verification equation uniquely defines v_i given σ and $\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})$. Furthermore, since the soundness of π guarantees that σ is a valid signature on $\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})$ under the extracted v, v_i identified in above equation must be identical to v. Hence, $\Pr[S_3] = \Pr[S_4]$.

[Game 5]: In Game 5, we modify how the Open oracle and the DOpen oracle obtain the signature σ. When the oracles receive a valid group signature, they decrypt the tag-based ciphertext with $\mathsf{xk} = (\log_G F, \log_G H)$ and extract σ instead of extracting from the

proof π. By the validity check of the tag-based ciphertext and the soundness of the NIZK proof ψ, this gives the same signature σ which we obtain when running the extractor on the NIWI proof system. Hence, $\Pr[S_4] = \Pr[S_5]$.

[Game 6]: We change for any Open and DOpen queries that are not equal to the challenge signature. We change how to produce (τ_F, τ_H), which is a part of opening proof, is generated. Instead of using xk, the Open oracle and the DOpen oracle use ζ and ξ to compute (τ_F, τ_H) as $\tau_F = (y_4/y_1^{\zeta})^{1/\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}, \tau_H = (y_5/y_2^{\xi})^{1/\mathcal{H}(\mathsf{vk}_{\mathrm{sots}})}$, and $\sigma = y_3/\tau_F\tau_H$. The response of the Open oracle and the DOpen oracle are exactly same with one in Game 5. Hence, $\Pr[S_5] = \Pr[S_6]$.

[Game 7]: We change how the DOpen oracle responses the query that is equal to the challenge signature. When generating a challenge signature, the Ch oracle keeps the randomness for the challenge signature secretly. Once the DOpen query that is equal to the challenge signature is issued, the DOpen oracle uses the randomness kept by the Ch to generate (τ_F, τ_H). The response of DOpen oracle in Game 7 is exactly same as that of Game 7. Hence $\Pr[S_6] = \Pr[S_7]$.

[Game 8]: In Game 7, the Open oracle and the DOpen oracle no longer need the extraction key xk. Therefore, we now can switch to using a simulated common reference string crs that provides perfect witness-indistinguishability and perfect zero-knowledge. In addition, we change the non-interactive proofs included in the challenge signature and the responses to the DOpen queries to be generated by the simulation algorithm S_2. Since a simulated common reference string and a real common reference string are computationally indistinguishable under the DLIN assumption, and the proofs included in the challenge signature and the responses to the DOpen queries are also computationally indistinguishable, the success probability of the adversary $\mathcal{A}_{anon}$ will not change by more than a negligible amount, hence we have that $|\Pr[S_6] - \Pr[S_7]|$ is negligible.

[Game 9]: Finally, we change the component y_3 in the challenge to a random element. As shown in [100], this will not introduce more than a negligible change in the success probability of the adversary $\mathcal{A}_{anon}$ assuming the DLIN assumption holds. However, one point is different from the proof of the modified Groth scheme. In anonymity game of deniable group signature, the adversary can query even the challenge signature to the DOpen oracle except for the challenge users. Therefore, a denial proof $\phi \leftarrow \mathsf{P}_{\mathrm{NIZK}}(\mathsf{crs}, (\mathsf{gpk}, y, v_j, c), (\sigma, \tau_F, \tau_H, v_i, \tau_\ell, \tau'_\ell))$ needs to be generated even though the witnesses $(\sigma, \tau_F, \tau_H, v_i, \tau_\ell, \tau'_\ell)$ are not known. Since the change in Game 7 and Game 8, we can generate a denial proof for the challenge signature without knowing the witness for this signature. More precisely, when the simulator receives a denial open query (m, Σ, j), the simulator verifies the signature first and, if it is not valid, he returns $\perp$. In the case that the signature is valid, he generates a simulated proof ϕ from the zero-knowledge trapdoor and random

c from $\mathbb{G}$, and outputs (ϕ, c). The randomness c has the same distribution as $(v_i/v_j)^\ell$ where ℓ is random in $\mathbb{Z}_p$, hence $|\Pr[S_7] - \Pr[S_8]|$ is negligible.

In Game 9, we can conclude that $\Pr[S_8] = 1/2$, because the view of the adversary is independent from the challenge bit b. In particular, the challenge $(\mathsf{vk}^*_{\mathrm{sots}}, a, \pi, y, \psi, \sigma^*_{\mathrm{sots}})$ contains no information of bit b. Indeed, $\mathsf{vk}^*_{\mathrm{sots}}$ is independently generated, a is re-randomized and uniformly random, the perfectly witness-indistinguishable proof π distributes independently from the witness, and y is a random encryption. Also, the proof ψ does not contain the information of b since the proof is computed from y and π by using the zero-knowledge trapdoor. Moreover, $\psi, \sigma^*_{\mathrm{sots}}$ is a signature of $(\mathsf{vk}^*_{\mathrm{sots}}, m, a, \pi, y, \psi)$ and the oracles behave independently of bit b. □

Theorem 7.4.2. *The proposed group signature scheme satisfies correctness.*

Proof. Correctness of the scheme follows from the correctness of both the modified Groth scheme and the NIZK proof system. □

Theorem 7.4.3. *The proposed group signature scheme satisfies non-frameability if the certified signature is EUF-wCMA, the one-time signature scheme is strong existential unforgeable, and the hash function is universal one-way.*

Proof. We assume the adversary $\mathcal{A}_{nf}$ who breaks the non-frameability of the proposed scheme. Let (m, Σ, j, τ_O) be the output of $\mathcal{A}_{nf}$. The adversary $\mathcal{A}_{nf}$ has two types of forgery, one is producing the valid group signature of honest user j and its acceptable opening proof and the other is producing the valid group signature of honest user j where his denial opening of the signature by $\mathsf{GS.DOpen}$ is not accepted by $\mathsf{GS.DJudge}$.

The first forgery, where the $\mathsf{GS.Judge}$ algorithm outputs 1, is trivially captured by the forgery of the non-frameability of the modified Groth scheme. Since the modified Groth scheme has non-frameability under the q-SDH assumption, the strong existential unforgeability of one-time signature scheme, and universal one-wayness of hash function, this type of forgery will not happen.

In the second forgery, where the $\mathsf{GS.DJudge}$ algorithm outputs 0, because the group signature is valid, opening result $(i, (\sigma, \tau_F, \tau_H))$ of the signature has valid structure, that is, $\sigma = G^{1/(x_i + \mathcal{H}(\mathsf{vk}_{\mathrm{sots}}))}$. If $i \neq j$, j's denial opening of the signature, that the adversary outputs, by $\mathsf{GS.DOpen}$ is accepted by $\mathsf{GS.DJudge}$, because the proposed scheme satisfies correctness. This contradicts the winning condition of the adversary. So, from now on, we assume that $i = j$. Then, σ is a forgery of the certified signature. More precisely, if $i = j$, σ is a forgery of the certified signature. More precisely, let $\mathcal{A}^{type2}_{nf}$ be an adversary who breaks the non-frameability of the proposed scheme using second type of forgery and we can construct the adversary $\mathcal{B}$ who breaks the existential unforgeability against weak chosen message attack of the certified signature. Here, by assuming that $\mathcal{A}^{type2}_{nf}$ will

query to the Sign oracle in $q(k)$ times, $\mathcal{B}$ is constructed as follows. Before running the game, $\mathcal{B}$ tries to guess the user j that $\mathcal{A}_{nf}^{type2}$ will frame in the non-frameability game of the proposed scheme. The probability is at least $\frac{1}{N(k)}$ where $N(k)$ is an upper bound of the number of honest users. In the existential unforgeability game, $\mathcal{B}$ gets the challenge verification key and certificate (v^*, cert^*) after he gets a description gk and sends a public authority key ak to the challenger. $\mathcal{B}$ queries $\mathcal{H}(\mathsf{vk}_{\mathrm{sots},i})$ and gets $\sigma_i = G^{1/(x^*+\mathcal{H}(\mathsf{vk}_{\mathrm{sots},i}))}$ in advance, where $1 \leq i \leq q$ and x^* is the signing key of v^* which is unknown to $\mathcal{B}$. After that, $\mathcal{B}$ generates keys of the proposed scheme by following $\mathsf{GS.GGen}$ and using gk which is given by the challenger and sends them to $\mathcal{A}_{nf}^{type2}$. When $\mathcal{A}_{nf}^{type2}$ accesses to oracles $\mathsf{SndToU}(\cdot)$, $\mathsf{WReg}(\cdot)$, $\mathsf{Sign}(\cdot,\cdot)$, $\mathsf{USK}(\cdot)$, and $\mathsf{Corrupt}(\cdot,\cdot)$, $\mathcal{B}$ can easily respond to the queries because he has all the keys of the proposed scheme. However, only when $\mathcal{A}_{nf}^{type2}$ accesses to oracles $\mathsf{SndToU}(j)$ and $\mathsf{Sign}(j,\cdot)$, for each queries, $\mathcal{B}$ lets (v^*, cert^*) as j's verification key and certificate, and uses σ_i which he got from the challenger of the existential unforgeability game to simulate the signing. We do not need to care about $\mathsf{USK}(j)$ since $\mathcal{A}_{nf}^{type2}$ never queries $\mathsf{USK}(j)$ on the winning condition of the non-frameability game. Finally, $\mathcal{A}_{nf}^{type2}$ outputs the forgery $(m, \Sigma = (\mathsf{vk}^*_{\mathrm{sots}}, \cdot, \cdot, \cdot, \cdot, \cdot), j, \tau_O)$ which satisfies $\sigma = G^{1/(x^*+\mathcal{H}(\mathsf{vk}^*_{\mathrm{sots}}))}$ where $(i, (\sigma, \tau_F, \tau_H)) \leftarrow \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \Sigma)$ and $i \neq j$. By the strong existential unforgeability of one-time signature scheme, $\mathsf{vk}^*_{\mathrm{sots}}$ is not one of $\mathsf{vk}_{\mathrm{sots},i}$ which is used in $\mathsf{Sign}(j,\cdot)$ with overwhelming probability. Moreover, by universal one-wayness of hash function, $\mathcal{H}(\mathsf{vk}^*_{\mathrm{sots}})$ does not collide with one of $vk_{\mathrm{sots},i}$, that is, $\mathcal{H}(\mathsf{vk}^*_{\mathrm{sots}}) \neq \mathcal{H}(\mathsf{vk}_{\mathrm{sots},i})$ holds, with overwhelming probability. Therefore, $\mathcal{B}$ can extract σ from (m, Σ) and output $(\mathsf{cert}^*, \mathcal{H}(\mathsf{vk}^*_{\mathrm{sots}}), \sigma)$ as a forgery of the certified signature. This contradicts that the certified signature is existential unforgeable against weak chosen message attack. □

Theorem 7.4.4. *The proposed group signature scheme satisfies traceability if the modified Groth scheme has traceability.*

Proof. We assume the adversary $\mathcal{A}_{trace}$ who breaks the traceability of the proposed scheme. The adversary $\mathcal{A}_{trace}$ has two types of forgery, one is producing the valid group signature whose opening result is not valid or opening proof is not accepted by $\mathsf{GS.Judge}$ and the other is producing the valid group signature, whose opening proof is accepted by $\mathsf{GS.Judge}$, but for the same user the denial opening proof is accepted by $\mathsf{GS.DJudge}$. The first forgery is just also the forgery of the traceability of the modified Groth scheme. Since the modified Groth scheme has traceability, this type of forgery will not happen. In the second forgery, from the fact that denial opening proof is accepted by $\mathsf{GS.DJudge}$, we require $c \neq 1$ where denial opening proof is (ϕ, c). However, since the $\mathsf{GS.Judge}$ algorithm outputs 1 for i, i is the signer of Σ. Then $c = 1$ holds. Therefore, the $\mathsf{GS.DJudge}$ algorithm, that checks $c \neq 1$, never output 1, and this case never happens. □

Theorem 7.4.5. *The proposed group signature scheme satisfies opening soundness if the modified Groth scheme has opening soundness.*

Proof. We assume the adversary $\mathcal{A}_{os}$ who breaks the opening soundness of the proposed scheme. The adversary $\mathcal{A}_{os}$ has two types of forgery, one is producing the valid group signature and its opening proofs of i and j which are both accepted by GS.Judge and the other is producing the valid group signature , its opening proof which is accepted by GS.Judge, and its denial opening proof which is also accepted by GS.DJudge. The first forgery is just also the forgery of the opening soundness of the modified Groth scheme. Since the modified Groth scheme has opening soundness, this type of forgery will not happen. In the second forgery, from the fact that denial opening proof is accepted by GS.DJudge, we require $c \neq 1$ where denial opening proof is (ϕ, c). However, since the GS.Judge algorithm outputs 1 for i, i is the signer of Σ. Then $c = 1$ holds. Therefore, the GS.DJudge algorithm, that checks $c \neq 1$, never output 1, and this case never happens. □

7.5 Related Works

Komano, Ohta, Shimbo, and Kawamura [77] pointed out that "*the ring signature scheme allows the signer to shift the blame to entities (victims) because of its anonymity,*" and proposed a deniable ring signature, where a verifier and a user run interactive confirm/disavow protocols and the user can insist that "I am the actual signer" (confirm) or "I am NOT the actual signer" (disavow). Komano et al.'s scheme is secure in the random oracle model, and later, Zeng et al. [105] proposed an improved scheme which is provably secure in the standard model. In deniable ring signatures, the *user* can claim that he is not the signer of the signature, while in the case of deniable group signatures, the *opener* can generate the proof of non-ownership of the signature. That is, in deniable group signatures, if all the users except for a user collude, his anonymity is still guaranteed unless the opener is not corrupted by them, unlike in the case of deniable ring signatures.

As group signatures with additional functionality, group signatures with message-dependent opening (GS-MDO) [99] were considered in order to restrict the authority of the opener. In GS-MDO, the opener can open group signatures on specific signed messages, as decided by another authority called the admitter. In particular, an automated private parking garage scenario was considered as an application of GS-MDO. In this case, a customer generates a group signature on the date he/she enters a private garage, which anyone cannot enter without authentication. Then if there is an accident (e.g., a person is murdered) in the garage, the opener opens all the signatures for the date of the accident to determine the customers present in the garage at the time of the accident. Again, if multiple customers enter the garage on the same date, then the false accusation

problem occurs.

Abe et al. [7] considered non-snatching and undeniability in the traceable signature context, where no one (but the actual signer) can claim to be the signer of a signature, and no actual signer can deny being the signer of his signatures, respectively. Abe et al.'s traceable signature scheme, in addition to the opening and user tracing, allows the signer to claim non-ownership of a signature (as in the case of deniable ring signatures [77, 105]) while in the case of deniable group signatures, the opener can generate the proof for non-ownership of a signature. That is, as we discussed about deniable ring signatures, a user's anonymity is not guaranteed if all the users except for the user collude.

Lyuu and Wu [85] considered group undeniable signatures where a verifier and a group manager run an interactive protocol that can prove the validity/invalidity of signatures without compromising anonymity. To the contrary, deniable group signatures support the non-interactive verification.

Brickell et al. [32] proposed direct anonymous attestations (DAA), which can be seen as group signatures without the opening functionality. They introduced a tag, called basename, which enables to link signatures produced by the same user with the same basename. After that, Desmoulins et al. [47] introduced DAA with dependent basename opening, which is the extension of DAA. In these primitives, even if the user can simply use a different basename for each time to sign a message, he can only prove that he generated the signature by producing a new signature on the same message with the same basename. That is, he cannot prove that he did not generate the signature.

Recently, Blazy et al. [21] proposed group signatures with verifiable controllable linkability (VCL-GS), where a dedicated linking authority (LA) can determine whether two given signatures stem from the same signer without being able to identify the signer(s). Compared to group signatures with controllable linkability, VCL-GS does not require trusted LAs.

Chapter 8

The Standardized Group Signature Scheme

In this chapter, we discuss the scheme denoted as Mechanism 6 in the ISO/IEC 20008-2 standard [2], which is a standardized group signature scheme.

In 2013, the ISO/IEC 20008-2 standard [2] was published, and several anonymous digital signature schemes are standardized by ISO/IEC JTC 1. This is a joint technical committee established by the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). The ISO/IEC standards are some of the most important reference documents representing a consensus among the experts in the field of information security. In practice, it is generally required to utilize the technologies which are specified in standards to ensure interoperability.

In the ISO/IEC 20008-2 standard, seven anonymous digital signature schemes (Mechanism 1 to 7) are specified. They are classified into two types: schemes with linking capability (Mechanism 1 to 4) and schemes with opening capability (Mechanism 5 to 7). The opening capability is the functionality that a special entity can trace the signer of a signature, and the linking capability is the notion relaxing the opening capability. Concretely, Mechanism 1 is a list signature scheme [38] and Mechanism 2, 3, and 4 are direct anonymous attestation schemes [32]. Moreover, Mechanism 5, 6, and 7 are group signature schemes [39] in a broad sense.

Among the group signature schemes in ISO/IEC 20008-2, Mechanism 6 is the only plain group signature scheme which does not aim at providing additional functionalities. More precisely, Mechanism 5 (the original paper [72]) introduces a special authority called a user-revocation manager, and Mechanism 7 has an additional functionality called controllable linkability [66]. Since Mechanism 6 is simpler than these schemes, it is the most efficient group signature scheme in the standards.

8.1 Description

Mechanism 6 is identical to the Furukawa-Imai group signature scheme [54] (the journal version [55]). Although their model is slightly different from the Bellare-Shi-Zhang (BSZ) model [17] which is reviewed in Chapter 4, it is easily seen that they are essentially same. Therefore in this thesis, we introduce Mechanism 6 by using the algorithms given by Bellare et al. Originally, the judging algorithm is not defined in Mechanism 6. However, we also describe its judging algorithm since it can be defined implicitly.

The description of Mechanism 6 is given in Figure 8.1 and 8.2. The intuition of the scheme is as follows: Consider a bilinear group pair $(\mathbb{G}_1, \mathbb{G}_2)$ with a computable isomorphism Ψ, and a group $\mathbb{G}$ in which the DDH assumption holds. We denote elements in $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, and $\mathbb{G}$ by upper case letters, and elements in $\mathbb{Z}_p$ by lower case letters. $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p$ is a family of hash functions modeled as random oracles. In Mechanism 6, a user i possesses a SDH pair (A_i, y_i) and a discrete logarithm x_i as a signing key where A_i is the certificate of x_i. When signing on a message m, the user encrypts the certificate A_i and the value $Q_i = G^{x_i}$, and generates a signature of knowledge on m for the statement that the encrypted certificate is valid, and the encryption procedure is honestly done. The signature is accepted when the signature of knowledge is valid. When opening a signature, the opener extracts Q_i by using the decryption key and outputs the ID i with a proof which shows that the decryption is honestly done.

$\mathsf{GS.GGen}(1^\lambda)$:
$G_2 \xleftarrow{r} \mathbb{G}_2;\ G \xleftarrow{r} \mathbb{G};\ G_1 \leftarrow \Psi(G_2);\ \mathsf{H} \xleftarrow{r} \mathcal{H}$
$H, K \xleftarrow{r} \mathbb{G}_1;\ w \xleftarrow{r} \mathbb{Z}_p;\ u, v \xleftarrow{r} \mathbb{Z}_p^*;\ Y \leftarrow {G_2}^w;\ U \leftarrow G^u;\ V \leftarrow G^v$
Return $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) = ((G_1, G_2, G, \mathsf{H}, H, K, Y, U, V), w, (u, v))$

$\mathsf{GS.UGen}(1^\lambda, \mathsf{gpk})$:
$x_i, z_i' \xleftarrow{r} \mathbb{Z}_p;\ Q_i \leftarrow G^{x_i};\ H_i \leftarrow H^{x_i} K^{z_i'}$
Return $(\mathsf{upk}_i, \mathsf{usk}_i) = ((Q_i, H_i), (x_i, z_i'))$

$\mathsf{GS.Join}/\mathsf{GS.Issue}$(User i: $\mathsf{gpk}, \mathsf{upk}_i, \mathsf{usk}_i$; Issuer: $\mathsf{gpk}, \mathsf{upk}_i, \mathsf{ik}$):
User: $\rho_{x_i}, \rho_{z_i'} \xleftarrow{r} \mathbb{Z}_p;\ R_1 \leftarrow G^{\rho_{x_i}};\ R_2 \leftarrow H^{\rho_{x_i}} K^{\rho_{z_i'}}$; Send (R_1, R_2) to the issuer
Issuer: $c_i \xleftarrow{r} \mathbb{Z}_p$; Send c_i to the user
User: $\sigma_{x_i} \leftarrow x_i \cdot c_i + \rho_{x_i};\ \sigma_{z_i'} \leftarrow z_i' \cdot c_i + \rho_{z_i'}$; Send $(\sigma_{x_i}, \sigma_{z_i'})$ to the issuer
Issuer: $R_1' \leftarrow G^{\sigma_{x_i}} / {Q_i}^{c_i};\ R_2' \leftarrow H^{\sigma_{x_i}} K^{\sigma_{z_i'}} / {H_i}^{c_i}$
Return $\perp$ to the user if $R_1' \neq R_1 \vee R_2' \neq R_2$
$y_i, z_i'' \xleftarrow{r} \mathbb{Z}_p;\ A_i \leftarrow \left(\frac{G_1}{H_i \cdot K^{z_i''}}\right)^{\frac{1}{w+y_i}};\ \mathsf{reg}[i] \leftarrow Q_i$
Send (A_i, y_i, z_i'') to the user
User: $z_i \leftarrow z_i' + z_i''$
Set $\mathsf{gsk}_i \leftarrow (A_i, y_i, z_i, x_i, Q_i)$ if $e(A_i, Y \cdot {G_2}^{y_i}) e(H^{x_i}, G_2) e(K^{z_i}, G_2) = e(G_1, G_2)$

Figure 8.1: Mechanism 6 (Former)

$\mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m)$:
$r, q \xleftarrow{r} \mathbb{Z}_p;\ T_1 \leftarrow A_i \cdot K^q;\ T_2 \leftarrow G^{x_i+r};\ T_3 \leftarrow U^r;\ T_4 \leftarrow V^r;\ \rho_{x_i}, \rho_{y_i}, \rho_\delta, \rho_q, \rho_r \xleftarrow{r} \mathbb{Z}_p$
$R_1 \leftarrow e(H, G_2)^{\rho_{x_i}} \cdot e(K, G_2)^{\rho_\delta} \cdot e(K, Y)^{-\rho_q} \cdot e(T_1, G_2)^{\rho_{y_i}}$
$R_2 \leftarrow G^{\rho_{x_i}+\rho_r};\ R_3 \leftarrow U^{\rho_r};\ R_4 \leftarrow V^{\rho_r}$
$c \leftarrow \mathsf{H}(\mathsf{gpk}, \{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m);\ \delta \leftarrow z_i - qy_i$
$\sigma_{x_i} \leftarrow x_i \cdot c + \rho_{x_i};\ \sigma_{y_i} \leftarrow y_i \cdot c + \rho_{y_i};\ \sigma_\delta \leftarrow \delta \cdot c + \rho_\delta;\ \sigma_q \leftarrow q \cdot c + \rho_q;\ \sigma_r \leftarrow r \cdot c + \rho_r$
Return $\Sigma = (\{T_i\}_{i\in[1,4]}, c, \sigma_{x_i}, \sigma_{y_i}, \sigma_\delta, \sigma_q, \sigma_r)$

$\mathsf{GS.Verify}(\mathsf{gpk}, m, \Sigma)$:
$R'_1 \leftarrow e(H, G_2)^{\sigma_{x_i}} \cdot e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} \cdot e(T_1, G_2)^{\sigma_{y_i}} \cdot \left(\frac{e(G_1,G_2)}{e(T_1,Y)}\right)^{-c}$
$R'_2 \leftarrow G^{\sigma_{x_i}+\sigma_r} \cdot {T_2}^{-c};\ R'_3 \leftarrow U^{\sigma_r} \cdot {T_3}^{-c};\ R'_4 \leftarrow V^{\sigma_r} \cdot {T_4}^{-c}$
Return 1 if $c = \mathsf{H}(\mathsf{gpk}, \{T_i\}_{i\in[1,4]}, \{R'_i\}_{i\in[1,4]}, m)$, else return 0

$\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \Sigma)$:
Return $\perp$ if $\mathsf{GVf}(\mathsf{gpk}, m, \Sigma) = 0$
$Q \leftarrow T_2 \cdot ({T_3}^{\frac{1}{u}})^{-1}$
Return $\perp$ if there is no user index i such that $\mathsf{reg}[i] = Q$
$\rho_u \xleftarrow{r} \mathbb{Z}_p;\ R \leftarrow (Q \cdot {T_2}^{-1})^{\rho_u};\ d \leftarrow \mathsf{H}(\mathsf{gpk}, Q, T_2, T_3, R);\ \sigma_u \leftarrow u \cdot d + \rho_u;\ \tau \leftarrow (d, \sigma_u)$
Return (i, τ)

$\mathsf{GS.Judge}(\mathsf{gpk}, \mathsf{reg}, m, \Sigma, (i, \tau))$:
Return $\perp$ if $\mathsf{GVf}(\mathsf{gpk}, m, \Sigma) = 0$
$Q \leftarrow \mathsf{reg}[i];\ R' \leftarrow (Q \cdot {T_2}^{-1})^{\sigma_u} \cdot T_3^{-d}$
Return 1 if $d = \mathsf{H}(\mathsf{gpk}, Q, T_2, T_3, R')$, else return 0

Figure 8.2: Mechanism 6 (Latter)

8.2 Cryptanalysis

In this section, we show an attack against the anonymity of Mechanism 6 [70]. In a nutshell, the reason why Mechanism 6 can be broken is that the underlying proof system does not satisfy simulation soundness. If a proof system is not simulation sound, it might be possible to create a valid proof without a witness after seeing some valid proofs.

In Mechanism 6, this possibility of creating a valid proof allows for the adversary to re-randomize the challenge signature and helps to break the anonymity. Specifically, in our attack, the challenge signature is re-randomized by using the issuing key. Then, the adversary queries the re-randomized signature to the opening oracle and can obtain the identity of the signer. Since the adversary is allowed to corrupt the issuer and to access the opening oracle in the anonymity game of the BSZ model, our attack is valid in this model. In the following, we provide more details of our attack.

Firstly, we show that the underlying proof system does not satisfy simulation soundness. In the proof system, for the group public key gpk and values $\{T_i\}_{i\in[1,4]}$, four equations are proved with witnesses x, y, δ, q, and r. A valid proof $\sigma_{\mathsf{set}} = \{\sigma_x, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r\}$

satisfies the following equations:

$$R_1 = e(H, G_2)^{\sigma_x} \cdot e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} \cdot e(T_1, G_2)^{\sigma_y} \cdot \Big(\frac{e(G_1, G_2)}{e(T_1, Y)}\Big)^{-c},$$
$$R_2 = G^{\sigma_x + \sigma_r} \cdot {T_2}^{-c},\ R_3 = U^{\sigma_r} \cdot {T_3}^{-c},\ R_4 = V^{\sigma_r} \cdot {T_4}^{-c}$$

where R_1, R_2, R_3, and R_4 are the commitments generated in the way of computing a signature, and c is a challenge value computed as $c \leftarrow \mathsf{H}(\mathsf{gpk}, \{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m)$ for a message m. When we focus on the first equation, the second and third terms of the right side on the equation have a common base $e(K, G_2)$ since $Y = {G_2}^w$ holds for the issuing key w. Thus, we can denote that $e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} = e(K, G_2)^{\sigma_\delta - \sigma_q \cdot w}$.

In fact, this property allows to break the simulation soundness by shuffling the discrete logarithms σ_δ and $-\sigma_q$. Now, we set $\widetilde{\sigma}_\delta = \sigma_\delta + w$ and $\widetilde{\sigma}_q = \sigma_q + 1$ where the values can be computed from the issuing key and a given valid proof. Then, the proof $\widetilde{\sigma}_{\mathsf{set}} = \{\sigma_x, \sigma_y, \widetilde{\sigma}_\delta, \widetilde{\sigma}_q, \sigma_r\}$ also satisfies the above equations. The first equation holds since it holds that $e(K, G_2)^{\widetilde{\sigma}_\delta} \cdot e(K, Y)^{-\widetilde{\sigma}_q} = e(K, G_2)^{\widetilde{\sigma}_\delta - \widetilde{\sigma}_q \cdot w} = e(K, G_2)^{\sigma_\delta + w - (\sigma_q + 1) \cdot w} = e(K, G_2)^{\sigma_\delta - \sigma_q \cdot w} = e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q}$, and the other equations hold trivially. Therefore, the forgery $\widetilde{\sigma}_{\mathsf{set}}$ is valid as an attack against the simulation soundness of the underlying proof system in the sense that it can be generated without a witness after seeing some valid proofs.

Secondly, we show that the above forgery against the simulation soundness derives an attack against the anonymity of Mechanism 6. In the anonymity game of the BSZ model, the adversary is allowed to corrupt the issuer. Thus, the adversary can compute a re-randomized signature $\widetilde{\Sigma}$ for the challenge signature Σ^* as above. Also, since the adversary can access the the opening oracle, and the re-randomized signature is not the same as the challenge signature (that is, $\widetilde{\Sigma} \neq \Sigma^*$ holds), the adversary can query the signature $\widetilde{\Sigma}$ to the opening oracle. Here, the signer's information hidden in the re-randomized signature is the same as that of the challenge signature since the difference between them is only the proof part. Thus, the adversary obtains the signer's ID of the challenge signature by this query. In this way, the anonymity of Mechanism 6 can be broken.

Countermeasures for Our Attack. We can consider the following three countermeasures for our attack: (1) to remove Mechanism 6 from the standards and use alternative schemes, (2) to patch Mechanism 6 and update the document, and (3) to analyze the security properties offered by Mechanism 6 and restrict its use in a way that ensures that its anonymity is preserved. In the following, we provide more details of each countermeasure.

- The countermeasure (1) seems easy but is not desirable. In fact, Mechanism 5 and 7 in the ISO/IEC 20008-2 standard are also group signature schemes in a broad sense. In addition to the functionality of group signatures, Mechanism 5 (the

original paper [72]) introduces a special authority called a user-revocation manager, and Mechanism 7 has a functionality called controllable linkability [66]. Therefore, at a first glance, Mechanism 5 and 7 might be considered reasonable substitutes for Mechanism 6. However, it is not always the case since Mechanism 5 and 7 have some drawbacks. Concretely, Mechanism 5 is significantly less efficient than Mechanism 6 due to the fact that Mechanism 5 is based on an RSA-type algebraic structure. Furthermore, Mechanism 7 provides only weaker security notion of anonymity, CPA-anonymity. This indicates that in Mechanism 7, once the opening result of at least one signature is revealed to the public, the anonymity of signatures is no more ensured. Therefore, the countermeasure (1) is not very appropriate because of these drawbacks.

- The countermeasure (2) is ideal and should be taken if possible. However, it cannot be carried out immediately since it takes much work and time to standardize a new scheme even though it is just an updated to an existing one. For example, in the case of the ISO/IEC 9796-2 standard [1] that specifies digital signature schemes for smart cards, one of the standardized schemes (denoted as Scheme 1) was attacked by Coron et al. [42] in 1999,[1] but the final revised version was not published before 2002. Specifically in this case, when it was seen that Scheme 1 is not secure, RSA-PSS [16] was known to be an adequate scheme to replace Scheme 1. That is, it took three long years to finally update the document even though there already existed such a candidate for an alternative scheme. (By the way, due to this delay of the update, Scheme 1 had populated a lot of commercial products (e.g., e-passports [3] and EMV cards [4]).) Therefore, the countermeasure (2) is not immediate countermeasure for the attack.

- The countermeasure (3) seems most realistic among the possible countermeasures. Although we see that Mechanism 6 does not satisfy the expected security level by our attack, it is premature to rule out Mechanism 6 as a useful scheme. Specifically, it might be that Mechanism 6 is still secure to use in practice since the BSZ model considers a relatively strong level of security, e.g., dynamic model, double authority, and CCA-anonymity. For example, since the BSZ model considers double authority, all of the entities except for the opener can corrupt in the anonymity game of this model. However, this seems not necessarily a real threat. Therefore, the countermeasure (3) seems most reasonable among the possible countermeasures.

From the above discussion, we investigate the countermeasure (3) as we consider that

[1] More precisely, Coron, Naccache and Stern [42] discovered that Scheme 1 is existentially forgeable in theory. After that, Coron, Naccache, Tibouchi, and Weinmann [43] showed a practical forgery for Scheme 1 in 2009.

this is the most appropriate one and analyze the security of Mechanism 6 in the next section.

8.3 Security Evaluation

In the previous section, we see that Mechanism 6 does not satisfy anonymity in the BSZ model, that is, it does not satisfy the expected security level in the ISO/IEC document. Here, as the most appropriate countermeasure, we analyze the security properties offered by Mechanism 6 and characterize the conditions under which its anonymity is preserved.

As we mentioned, the flaw of Mechanism 6 is that the underlying proof system does not satisfy simulation soundness, and this property allows to break the anonymity by re-randomizing the challenge signature. In fact, it seems that such an attack is the only way to break the anonymity of Mechanism 6 since the scheme is well structured except for the proof part.

Therefore, we analyze the security of Mechanism 6 in the following way: Firstly, we prove that Mechanism 6 satisfies anonymity under the restricted condition that the adversary does not make such a type of attack (in Section 8.3.1). Secondly, we provide further analysis of the attack by classifying some cases depending on the types of the adversary's queries (in Section 8.3.2). From the result of this analysis, we can characterize the conditions under which the anonymity of Mechanism 6 is preserved. Finally, we formalize these conditions and formally prove the strict security of Mechanism 6 under these (in Section 8.3.3 and 8.3.4, respectively).

8.3.1 Proof of the Anonymity under the Restricted Condition

In this section, we formalize the attack to re-randomize the challenge signature by forging its proof part and query it to the open oracle, and show that Mechanism 6 is secure if the adversary does not make this type of attack. More precisely, we formalize a query of a re-randomized signature generated by forging the proof part (called "related query" in the following), and then prove that Mechanism 6 satisfies anonymity against the adversary who does not generate any such a type of queries.

Firstly, we define a related query. Intuitively, a related query is a query which is obtained by re-randomizing the challenge signature through changing only the proof part. Let m^* and $\Sigma^* = (\{T_i^*\}_{i\in[1,4]}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$ be the challenge message and the challenge signature, respectively. Formally, a related query is defined as follows.

A Related Query: We say that a query $(\widetilde{m}, \widetilde{\Sigma} = (\{\widetilde{T}_i\}_{i\in[1,4]}, \widetilde{c}, \widetilde{\sigma}_x, \widetilde{\sigma}_y, \widetilde{\sigma}_\delta, \widetilde{\sigma}_q, \widetilde{\sigma}_r))$ is a

related query if $(\widetilde{m}, \widetilde{\Sigma})$ is accepted by the verification algorithm, and it holds that

$$(\{\widetilde{T}_i\}_{i\in[1,4]}, \{\widetilde{R}_i\}_{i\in[1,4]}, \widetilde{m}) = (\{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*)$$

where $\{\widetilde{R}_i\}_{i\in[1,4]}$ and $\{R_i^*\}_{i\in[1,4]}$ are the intermediate values computed in the verification of pairs $(\widetilde{m}, \widetilde{\Sigma})$ and (m^*, Σ^*), respectively. However, we do not regard the pair (m^*, Σ^*) itself as a related query since it is not accepted by the opening oracle.

Then, we prove that Mechanism 6 satisfies anonymity if the adversary does not generate a related query. We provide the games Game from 0 to 7, and prove that for $0 \leq \ell \leq 6$, the advantages of the adversary in Game ℓ and Game $\ell + 1$ are almost the same (which we denote Game $\ell \approx$ Game $\ell + 1$). Game 0 is the original anonymity game and Game 7 is the game that the adversary wins with the probability 1/2. In fact for $\ell \neq 5$, it holds that Game $\ell \approx$ Game $\ell + 1$ for the adversary without restriction on querying. However, when proving Game 5 $\approx$ Game 6, we need the condition that the adversary who does not generate a related query. Formally, we prove the following theorem.

Theorem 8.3.1. *If the adversary does not generate a related query, Mechanism 6 satisfies anonymity under the DDH assumption in the group $\mathbb{G}$ in the random oracle model.*

Proof. Let $\mathcal{A}$ be an adversary that attacks the anonymity of Mechanism 6 Π_{FI}. We consider the following sequence of games. Let $\mathtt{Suc}_\ell$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game ℓ.

[Game 0]: This is the experiment $\mathsf{Exp}^{anon}_{\Pi_{\mathsf{FI}},\mathcal{A}}(\lambda)$ itself. The challenger manages an in-out/output pair of the random oracle in the list L. More precisely, when the adversary queries x to the random oracle, the challenger returns y if there is a pair (x, y) in L. On the other hand if there is no pair $(x, \cdot)$ in L, the challenger samples a value y uniform randomly and returns y to the adversary. Then, the challenger adds (x, y) to the list L. In the following, we denote $y = \mathsf{H}(x)$ if there exists a pair (x, y) in the list. For the sake of convenience, we assume that the adversary queries $(\mathsf{gpk}, \{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m)$ to the random oracle before he queries $(m, \Sigma = \{T_i\}_{i\in[1,4]}, c, \sigma_x, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r))$ to the Open oracle where $R_1 = e(H, G_2)^{\sigma_x} \cdot e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} \cdot e(T_1, G_2)^{\sigma_y} \cdot \left(\frac{e(G_1,G_2)}{e(T_1,Y)}\right)^{-c}$, $R_2 = G^{\sigma_x+\sigma_r} \cdot {T_2}^{-c}$, $R_3 = U^{\sigma_r} \cdot {T_3}^{-c}$, and $R_4 = V^{\sigma_r} \cdot {T_4}^{-c}$. Since we can construct the adversary who generates the involved random oracle query before querying to the Open oracle by using the adversary who does not generate the involved random oracle query before querying to the Open oracle, the condition can be assumed without loss of generality.

[Game 1]: We modify the way to generate the challenge signature in Game 1. More precisely, if there is already the pair $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), \cdot)$ in the list L when

computing the value $\mathsf{H}(\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*)$, the challenger sets $\Sigma^* = \perp$. If there is not such a value, the challenger generates the challenge signature as in Game 0.

[Game 2]: We further modify the way to generate the challenge signature. In this game, the challenge signature is generated as follows:

Step 1. Choose values $r^*, q^* \in \mathbb{Z}_p$ uniformly random and compute $T_1^*, T_2^*, T_3^*, T_4^*$ as in Game 1.

Step 2. Choose $\sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^* \in \mathbb{Z}_p$ and $c^* \in \mathbb{Z}_p$ uniformly random, and compute $R_1^* = e(H, G_2)^{\sigma_x^*} \cdot e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \cdot \left(\frac{e(G_1, G_2)}{e(T_1^*, Y)}\right)^{-c^*}$, $R_2^* = G^{\sigma_x^* + \sigma_r^*} \cdot T_2^{*-c^*}$, $R_3^* = U^{\sigma_r^*} \cdot T_3^{*-c^*}$, and $R_4^* = V^{\sigma_r^*} \cdot T_4^{*-c^*}$.

Step 3. If a value $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), \cdot)$ is not defined in the list L, the value $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), c^*)$ is added to L and the challenge signature Σ^* is set to be $(\{T_i^*\}_{i\in[1,4]}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$. On the other hand, if such a value is already defined, the challenge signature is set to be $\perp$.

[Game 3]: In this game, we modify the way to generate a proof τ in replying queries for the Open oracle. More precisely, if there is already the pair $((\mathsf{gpk}, Q, T_2, T_3, R), \cdot)$ in the list L when computing the value $\mathsf{H}(\mathsf{gpk}, Q, T_2, T_3, R)$ in the generation of τ, the challenger replies $\perp$ as the response of the query.

[Game 4]: We further modify the way to generate a proof τ in replying queries for the Open oracle. The challenger replies for a query $(m, \Sigma = (\{T_i\}_{i\in[1,4]}, c, \sigma_x, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r))$ as follows. We note that steps except for Step 3 are the same as Game 3.

Step 1. If $\mathsf{GVf}(\mathsf{gpk}, m, \Sigma) = 0$, return 0.

Step 2. Compute $Q = T_2 \cdot (T_3{}^{\frac{1}{u}})^{-1}$ and find the index i such that $\mathsf{reg}[i] = Q$ in the list reg. If there is no such i, return $(0, \perp)$.

Step 3. Choose $\sigma_u \in \mathbb{Z}_p$ and $d \in \mathbb{Z}_p$ uniformly random, and set $R = (Q \cdot T_2{}^{-1})^{\sigma_u} \cdot T_3^{-d}$.

Step 4. If the value $((\mathsf{gpk}, Q, T_2, T_3, R), \cdot)$ is not defined in the list L, the value $((\mathsf{gpk}, Q, T_2, T_3, R), d)$ is added to L and reply $(i, \tau = (d, \sigma_u))$ to the adversary. On the other hand, if such a value is already defined, the opening proof τ is set to be $\perp$.

[Game 5]: We modify the way to generate a factor T_4^* in the challenge signature. More precisely, in Game 5, the challenger newly samples a random value $r_2^* \in \mathbb{Z}$ and computes $T_2^* = G^{x_{i_b} + r^*}, T_4^* = G^{r_2^*}$ by comparing Game 4 in which he computes $T_2^* = G^{x_{i_b} + r^*}, T_4^* = V^{r^*}$ where $r^* \in \mathbb{Z}$ is a uniform random value.

[Game 6]: In this game, the key to open signatures is changed from u to v. More precisely, in Game 6, the challenger sets $Q = T_2 \cdot (T_4{}^{\frac{1}{v}})^{-1}$ by comparing Game 5 in which he sets $Q = T_2 \cdot (T_3{}^{\frac{1}{u}})^{-1}$.

[Game 7]: We modify the way to generate a factor T_3^* in the challenge signature. More precisely, in Game 7, the challenger newly samples a random value $r_1^* \in \mathbb{Z}$ and computes $T_2^* = G^{x_{i_b}+r^*}, T_3^* = G^{r_1^*}$ by comparing Game 6 in which he computes $T_2^* = G^{x_{i_b}+r^*}, T_3^* = U^{r^*}$ where $r^* \in \mathbb{Z}$ is a uniform random value.

For the advantage $\mathsf{Adv}^{anon}_{\Pi_{\mathsf{FI}},\mathcal{A}}(\lambda)$,

$$\mathsf{Adv}^{anon}_{\Pi_{\mathsf{FI}},\mathcal{A}}(\lambda) = \big|\Pr[\mathtt{Suc}_0] - 1/2\big| \leq \sum_{\ell=0}^{6} \big|\Pr[\mathtt{Suc}_\ell] - \Pr[\mathtt{Suc}_{\ell+1}]\big| + \big|\Pr[\mathtt{Suc}_7] - 1/2\big|$$

holds. Moreover, the following lemmas hold.

Lemma 8.3.1. *Let q_H be the number of $\mathcal{A}$'s random oracle queries. Then, it holds that $\big|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\big| \leq q_H/p$ for any PPT $\mathcal{A}$.*

Proof. We define the event $\mathtt{Bad}_\ell^{(1)}$ as follows.

$\mathtt{Bad}_\ell^{(1)}$**:** The event that there is already the pair $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), \cdot)$ in the list L when computing the value $\mathsf{H}(\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*)$ in Game ℓ.

Game 0 and Game 1 are identical unless the events $\mathtt{Bad}_0^{(1)}$ and $\mathtt{Bad}_1^{(1)}$ occur. That is, we get $\Pr[\mathtt{Suc}_0 \wedge \neg\mathtt{Bad}_0^{(1)}] = \Pr[\mathtt{Suc}_1 \wedge \neg\mathtt{Bad}_1^{(1)}]$. Therefore, it holds that $\big|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\big| = \big|\Pr[\mathtt{Suc}_0 \wedge \mathtt{Bad}_0^{(1)}] + \Pr[\mathtt{Suc}_0 \wedge \neg\mathtt{Bad}_0^{(1)}] - \Pr[\mathtt{Suc}_1 \wedge \mathtt{Bad}_1^{(1)}] - \Pr[\mathtt{Suc}_1 \wedge \neg\mathtt{Bad}_1^{(1)}]\big| = \big|\Pr[\mathtt{Suc}_0 \wedge \mathtt{Bad}_0^{(1)}] - \Pr[\mathtt{Suc}_1 \wedge \mathtt{Bad}_1^{(1)}]\big| \leq \Pr[\mathtt{Bad}_1^{(1)}]$.

Here, we estimate the probability $\Pr[\mathtt{Bad}_1^{(1)}]$. When the event $\mathtt{Bad}_1^{(1)}$ occurs, $\widetilde{T}_1 = T_1^*$ holds for some defined value $((\cdot, \widetilde{T}_1, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot), \cdot)$ in the list L. Since $q^* \in \mathbb{Z}_p$ is chosen uniform randomly in Game 1, $T_1^* = A_{i_b} \cdot K^{q^*} \in \mathbb{G}_1$ is also uniformly random. Also, the number of values in the list L is at least q_H. Therefore, the probability that $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), \cdot)$ is already stored in L when generating the challenge signature is at most q_H/p. That is, $\Pr[\mathtt{Bad}_1^{(1)}] \leq q_H/p$. Thus, we obtain $\big|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\big| \leq q_H/p$. □

Lemma 8.3.2. *It holds that $\Pr[\mathtt{Suc}_1] = \Pr[\mathtt{Suc}_2]$ for any PPT $\mathcal{A}$.*

Proof. For Game 2, we introduce new values $\rho_x^*, \rho_y^*, \rho_\delta^*, \rho_q^*, \rho_r^* \in \mathbb{Z}_p$, and set $\rho_x^* = \sigma_x^* - x_{i_b} \cdot c^*$, $\rho_y^* = \sigma_y^* - y_{i_b} \cdot c^*$, $\rho_\delta^* = \sigma_\delta^* - \delta^* \cdot c^*$, $\rho_q^* = \sigma_q^* - q^* \cdot c^*$, and $\rho_r^* = \sigma_r^* - r^* \cdot c^*$.

Then, the following equations hold:

$$R_1^* \;=\; e(H, G_2)^{\sigma_x^*} \cdot e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \cdot \Big(\frac{e(G_1, G_2)}{e(T_1^*, Y)}\Big)^{-c^*}$$

$$= \quad e(H, G_2)^{\rho_x^*} \cdot e(K, G_2)^{\rho_\delta^*} \cdot e(K, Y)^{-\rho_q^*} \cdot e({T_1}^*, G_2)^{\rho_y^*},$$

$R_2^* = G^{\sigma_x^* + \sigma_r^*} \cdot (T_2^*)^{-c^*} = G^{\rho_x^* + \rho_r^*}$, $R_3^* = U^{\sigma_r^*} \cdot (T_3^*)^{-c^*} = U^{\rho_r^*}$, $R_4^* = V^{\sigma_r^*} \cdot (T_4^*)^{-c^*} = V^{\rho_r^*}$.

Moreover, it holds that $\sigma_x^* = x_{i_b} \cdot c^* + \rho_x^*$, $\sigma_y^* = y_{i_b} \cdot c^* + \rho_y^*$, $\sigma_\delta^* = \delta^* \cdot c^* + \rho_\delta^*$, $\sigma_q^* = q^* \cdot c^* + \rho_q^*$, and $\sigma_r^* = r^* \cdot c^* + \rho_r^*$. Furthermore, $\rho_x^*, \rho_y^*, \rho_\delta^*, \rho_q^*, \rho_r^* \in \mathbb{Z}_p$ are uniformly random since σ_x^*, σ_y^*, $\sigma_\delta^*, \sigma_q^*, \sigma_r^* \in \mathbb{Z}_p$ are chosen uniform randomly. Therefore, Game 2 is identical to Game 1. That is, $\Pr[\mathtt{Suc}_1] = \Pr[\mathtt{Suc}_2]$. □

Lemma 8.3.3. *Let q_H and q_{open} be the number of $\mathcal{A}$'s random oracle queries and opening queries, respectively. Then, it holds that $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| \leq q_H \cdot q_{\mathsf{open}}/p$ for any PPT $\mathcal{A}$.*

Proof. We define the event $\mathtt{Bad}_\ell^{(2)}$ as follows.

$\mathtt{Bad}_\ell^{(2)}$**:** The event that there is already the pair $((\mathsf{gpk}, Q, T_2, T_3, R), \cdot)$ in the list L when computing the value $\mathsf{H}(\mathsf{gpk}, Q, T_2, T_3, R)$ during the generation of an opening proof τ in Game ℓ.

Game 2 and Game 3 are identical unless the events $\mathtt{Bad}_2^{(2)}$ and $\mathtt{Bad}_3^{(2)}$ occur. Therefore, we get $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| \leq \Pr[\mathtt{Bad}_3^{(2)}]$ same as in Lemma 8.3.1.

Here, we estimate the probability $\Pr[\mathtt{Bad}_3^{(2)}]$. When the event $\mathtt{Bad}_3^{(2)}$ occurs, $\widetilde{R} = R$ holds for the some defined value $((\cdot, \cdot, \cdot, \cdot, \widetilde{R}), \cdot)$ in the list L. Since $\rho_u \in \mathbb{Z}_p$ is chosen uniform randomly in Game 3, $R = (Q \cdot {T_2}^{-1})^{\rho_u} \in \mathbb{G}$ is also uniformly random. Also, the number of values in the list L is at least q_H. Therefore, the probability that $((\mathsf{gpk}, Q, T_2, T_3, R), \cdot)$ is already stored in L when generating an opening proof is at most q_H/p. By the union bound, $\Pr[\mathtt{Bad}_3^{(2)}] \leq q_H \cdot q_{\mathsf{open}}/p$ holds. Thus, we obtain $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| \leq q_H \cdot q_{\mathsf{open}}/p$. □

Lemma 8.3.4. *It holds that $\Pr[\mathtt{Suc}_3] = \Pr[\mathtt{Suc}_4]$ for any PPT $\mathcal{A}$.*

Proof. For Game 4, we introduce new values ρ_u, and sets $\rho_u = \sigma_u - u \cdot d$. Then, $R_1 = (Q \cdot {T_2}^{-1})^{\sigma_u} \cdot T_3^{-d} = (Q \cdot {T_2}^{-1})^{\rho_u}$ and $\sigma_u = u \cdot d + \rho_u$ hold. Moreover, $\rho_u \in \mathbb{Z}_p$ is uniformly random since $\sigma_u \in \mathbb{Z}_p$ is chosen uniform randomly. Therefore, Game 4 is identical to Game 3. That is, $\Pr[\mathtt{Suc}_3] = \Pr[\mathtt{Suc}_4]$. □

Lemma 8.3.5. *There exists a* PPT *algorithm $\mathcal{B}_1$ such that $\left|\Pr[\mathtt{Suc}_4] - \Pr[\mathtt{Suc}_5]\right| = \mathsf{Adv}_{\mathcal{B}_1}^{DDH}(\lambda)$ for any PPT $\mathcal{A}$.*

Proof. Let $\mathcal{B}_1$ be an adversary that tries to solve the DDH problem. First, $\mathcal{B}_1$ receives the DDH tuple $G, V, R, W \in \mathbb{G}$. Let $V = G^v$, and $R = G^r$. The element W is G^{vr} or a random value in $\mathbb{G}$. Next, $\mathcal{B}_1$ generates the instance of the anonymity game. Here for G

and V, he uses the ones in the DDH tuple. Other elements are generated by following the $\mathsf{GS.GGen}$ algorithm. Let $\mathsf{gpk} = (G_1, G_2, G, \mathsf{H}, H, K, Y, U, V)$, $\mathsf{ik} = w$, and $\mathsf{ok} = (u, v)$, $\mathcal{B}_1$ sends $(\mathsf{gpk}, \mathsf{ik})$ to the adversary $\mathcal{A}$. We note that $\mathcal{B}_1$ does not know the discrete logarithm v of the value V. Although v is the part of the opening key ok, the key that is used for opening in Game 4 and Game 5 is $u = \log_G U$. Therefore, $\mathcal{B}_1$ possesses all the keys which are needed to reply oracle queries, and can simulate the replies of all the queries. Especially, $\mathcal{B}_1$ generates the challenge signature as follows:

1. Choose $q^* \in \mathbb{Z}_p$ uniform randomly and compute $(T_1^*, T_2^*, T_3^*, T_4^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot R, R^u, W)$ where R and W are the part of the DDH tuple.

2. Choose $\sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^* \in \mathbb{Z}_p$ and $c^* \in \mathbb{Z}_p$ uniform randomly, and computes $R_1^* = e(H, G_2)^{\sigma_x^*} \cdot e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \cdot \left(\frac{e(G_1, G_2)}{e(T_1^*, Y)}\right)^{-c^*}$, $R_2^* = G^{\sigma_x^* + \sigma_r^*} \cdot T_2^{*-c^*}$, $R_3^* = U^{\sigma_r^*} \cdot T_3^{*-c^*}$, and $R_4^* = V^{\sigma_r^*} \cdot T_4^{*-c^*}$.

3. If the value $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), \cdot)$ is not defined in the list L, the value $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), c^*)$ is added to L and the challenge signature Σ^* is set to be $(\{T_i^*\}_{i\in[1,4]}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$. On the other hand, if such a value is already defined, the challenge signature is set to be $\perp$.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b} \in \{0, 1\}$, $\mathcal{B}_1$ outputs 1 if $b = \widetilde{b}$. Otherwise he outputs 0.

If the DDH tuple that $\mathcal{B}_1$ obtains is $(G, V, R, W) = (G, G^v, G^r, G^{vr})$, it holds that $(T_1^*, T_2^*, T_3^*, T_4^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot G^r, G^{ur}, G^{vr}) = (A_{i_b} \cdot K^{q^*}, G^{x_{i_b}+r}, U^r, V^r)$. Then, $\mathcal{B}_1$ perfectly simulates Game 4 for $\mathcal{A}$. On the other hand, if the element W is a random value in $\mathbb{G}$, it holds that $(T_1^*, T_2^*, T_3^*, T_4^*) = (A_{i_b} \cdot K^{q^*}, G^{x_{i_b}+r}, U^r, W)$. Then, $\mathcal{B}_1$ perfectly simulates Game 5 for $\mathcal{A}$. Therefore, it holds that $\mathsf{Adv}_{\mathcal{B}_1}^{DDH}(\lambda) = |\Pr[1 \leftarrow \mathcal{B}_1(G, G^v, G^r, G^{vr})] - \Pr[1 \leftarrow \mathcal{B}_1(G, G^v, G^r, W)]| = |\Pr[b = \widetilde{b} \text{ in Game 4}] - \Pr[b = \widetilde{b} \text{ in Game 5}]| = \left|\Pr[\mathtt{Suc}_4] - \Pr[\mathtt{Suc}_5]\right|$. □

Lemma 8.3.6. *If the adversary does not generate a related query, it holds that* $\left|\Pr[\mathtt{Suc}_5] - \Pr[\mathtt{Suc}_6]\right| \leq 1/p$ *for any PPT* $\mathcal{A}$.

Proof. We define the event $\mathtt{Bad}_\ell^{(3)}$ as follows.

$\mathtt{Bad}_\ell^{(3)}$**:** The event that the adversary $\mathcal{A}$ sends the opening query $(m, \Sigma = (\{T_i\}_{i\in[1,4]}, c, \sigma_x, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r))$ such that $\mathsf{GS.Verify}(\mathsf{gpk}, m, \Sigma) = 1$ and $\log_U T_3 \neq \log_V T_4$ in Game ℓ.

Game 5 and Game 6 are identical unless the events $\mathtt{Bad}_5^{(3)}$ and $\mathtt{Bad}_6^{(3)}$ occur. Therefore, we get $\left|\Pr[\mathtt{Suc}_5] - \Pr[\mathtt{Suc}_6]\right| \leq \Pr[\mathtt{Bad}_6^{(3)}]$ same as in Lemma 8.3.1. Moreover, we define the event $\overline{\mathtt{Bad}}_6^{(3)}$ as follows.

$\overline{\mathtt{Bad}_6^{(3)}}$**:** The event that in Game 6, the adversary $\mathcal{A}$ sends the random oracle query (gpk, $\{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m$) such that $\log_U T_3 \neq \log_V T_4$ and $(\{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m) \neq (\{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*)$, and there exists σ_r such that

$$\begin{pmatrix} \log_U R_3 \\ \log_V R_4 \end{pmatrix} = \begin{pmatrix} 1 & -\log_U T_3 \\ 1 & -\log_V T_4 \end{pmatrix} \begin{pmatrix} \sigma_r \\ \widetilde{c} \end{pmatrix} \tag{8.1}$$

where $\widetilde{c}$ is the reply of the query ($\mathsf{gpk}, \{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m$).

When $\log_U T_3 \neq \log_V T_4$ holds, it holds that

$$\left| \begin{pmatrix} 1 & -\log_U T_3 \\ 1 & -\log_V T_4 \end{pmatrix} \right| = \left| \log_U T_3 - \log_V T_4 \right| \neq 0.$$

Therefore, the simultaneous equation (8.1) has the unique solution $(\sigma_r, \widetilde{c})$. Since it holds that $(\{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m) \neq (\{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*)$, $\widetilde{c}$ is chosen uniform randomly for $(\{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m)$. Thus, the probability that there exists σ_r such that the equation (8.1) holds for $\widetilde{c}$ is $1/p$. That is, $\Pr[\overline{\mathtt{Bad}_6^{(3)}}] = 1/p$.

In the following, we prove $\left|\Pr[\mathtt{Suc}_5] - \Pr[\mathtt{Suc}_6]\right| \leq 1/p$ by showing $\mathtt{Bad}_6^{(3)} \subseteq \overline{\mathtt{Bad}_6^{(3)}}$. We consider that the event $\mathtt{Bad}_6^{(3)}$ happens, that is, the situation that $\mathcal{A}$ sends the opening query $(m, \Sigma = (\{T_i\}_{i\in[1,4]}, c, \sigma_x, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r))$ such that $\mathsf{GS.Verify}(\mathsf{gpk}, m, \Sigma) = 1$ and $\log_U T_3 \neq \log_V T_4$. Since we assume that the adversary $\mathcal{A}$ does not generate related queries, it holds that $(\{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m) \neq (\{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*)$. Also from the condition which is made in Game 0, the random oracle query $X = (\mathsf{gpk}, \{T_i\}_{i\in[1,4]}, \{R_i\}_{i\in[1,4]}, m)$ is generated before (m, Σ) is queried to the Open oracle where $R_1 = e(H, G_2)^{\sigma_x} \cdot e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} \cdot e(T_1, G_2)^{\sigma_y} \cdot \left(\frac{e(G_1,G_2)}{e(T_1,Y)}\right)^{-c}$, $R_2 = G^{\sigma_x+\sigma_r} \cdot {T_2}^{-c}$, $R_3 = U^{\sigma_r} \cdot {T_3}^{-c}$, and $R_4 = V^{\sigma_r} \cdot {T_4}^{-c}$. Let $\widetilde{c}$ be the reply of X. Since $c = \widetilde{c}$ holds when $\mathsf{GS.Verify}(\mathsf{gpk}, m, \Sigma) = 1$, it holds that $R_3 = U^{\sigma_r} \cdot {T_3}^{-\widetilde{c}}$ and $R_4 = V^{\sigma_r} \cdot {T_4}^{-\widetilde{c}}$. For the two equations, we consider the discrete logarithm by considering the base as U and V, and then the simultaneous equation

$$\begin{pmatrix} \log_U R_3 \\ \log_V R_4 \end{pmatrix} = \begin{pmatrix} 1 & -\log_U T_3 \\ 1 & -\log_V T_4 \end{pmatrix} \begin{pmatrix} \sigma_r \\ \widetilde{c} \end{pmatrix}$$

holds. Therefore, the query X satisfies two conditions of the event $\overline{\mathtt{Bad}_6^{(3)}}$, and there exists σ_r which satisfies the equation (8.1) for the reply $\widetilde{c}$. Thus, $\mathtt{Bad}_6^{(3)} \subseteq \overline{\mathtt{Bad}_6^{(3)}}$ holds and we obtain $\left|\Pr[\mathtt{Suc}_5] - \Pr[\mathtt{Suc}_6]\right| \leq \Pr[\mathtt{Bad}_6^{(3)}] \leq \Pr[\overline{\mathtt{Bad}_6^{(3)}}] = 1/p$. □

Lemma 8.3.7. *There exists a* PPT *algorithm* $\mathcal{B}_2$ *such that* $\left|\Pr[\mathtt{Suc}_6] - \Pr[\mathtt{Suc}_7]\right| = \mathsf{Adv}_{\mathcal{B}_2}^{DDH}(\lambda)$ *for any PPT* $\mathcal{A}$.

Proof. Let $\mathcal{B}_2$ be an adversary that tries to solve the DDH problem. First, $\mathcal{B}_2$ receives the DDH tuple $G, U, R, W \in \mathbb{G}$. Let $U = G^u$ and $R = G^r$. The element W is G^{ur} or a random value in $\mathbb{G}$. Next, $\mathcal{B}_2$ generates the instance of the anonymity game. Here for G and U, he uses the ones in the DDH tuple. Other elements are generated by following the $\mathsf{GS.GGen}$ algorithm. Let $\mathsf{gpk} = (G_1, G_2, G, \mathsf{H}, H, K, Y, U, V)$, $\mathsf{ik} = w$, and $\mathsf{ok} = (u, v)$, $\mathcal{B}_2$ sends $(\mathsf{gpk}, \mathsf{ik})$ to the adversary $\mathcal{A}$. We note that $\mathcal{B}_2$ does not know the discrete logarithm u of the value U. Although u is the part of the opening key ok, the key that is used for opening in Game 6 and Game 7 is $v = \log_G V$. Therefore, $\mathcal{B}_2$ possesses all the keys which are needed to reply oracle queries, and can simulate the replies of all the queries. Especially, $\mathcal{B}_2$ generates the challenge signature as follows:

1. Choose $q^*, r_2^* \in \mathbb{Z}_p$ uniform randomly and compute $(T_1^*, T_2^*, T_3^*, T_4^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot R, W, G^{r_2^*})$ where R and W are the part of the DDH tuple.

2. Choose $\sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^* \in \mathbb{Z}_p$ and $c^* \in \mathbb{Z}_p$ uniform randomly, and computes $R_1^* = e(H, G_2)^{\sigma_x^*} \cdot e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \cdot \left(\frac{e(G_1, G_2)}{e(T_1^*, Y)}\right)^{-c^*}$, $R_2^* = G^{\sigma_x^* + \sigma_r^*} \cdot T_2^{*-c^*}$, $R_3^* = U^{\sigma_r^*} \cdot T_3^{*-c^*}$, and $R_4^* = V^{\sigma_r^*} \cdot T_4^{*-c^*}$.

3. If the value $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), \cdot)$ is not defined in the list L, the value $((\mathsf{gpk}, \{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*), c^*)$ is added to L and the challenge signature Σ^* is set to be $(\{T_i^*\}_{i\in[1,4]}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$. On the other hand, if such a value is already defined, the challenge signature is set to be $\perp$.

Finally, when $\mathcal{A}$ terminates with $\widetilde{b} \in \{0, 1\}$, $\mathcal{B}_2$ outputs 1 if $b = \widetilde{b}$. Otherwise he outputs 0.

If the DDH tuple that $\mathcal{B}_2$ obtains is $(G, U, R, W) = (G, G^u, G^r, G^{ur})$, it holds that $(T_1^*, T_2^*, T_3^*, T_4^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot G^r, G^{ur}, G^{r_2^*})$. Then, $\mathcal{B}_2$ perfectly simulates Game 6 for $\mathcal{A}$. On the other hand, if the element W is a random value in $\mathbb{G}$, it holds that $(T_1^*, T_2^*, T_3^*, T_4^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot G^r, W, G^{r_2^*})$. Then, $\mathcal{B}_2$ perfectly simulates Game 7 for $\mathcal{A}$. Therefore, it holds that $\mathsf{Adv}_{\mathcal{B}_2}^{DDH}(\lambda) = |\Pr[1 \leftarrow \mathcal{B}_2(G, G^u, G^r, G^{ur})] - \Pr[1 \leftarrow \mathcal{B}_2(G, G^u, G^r, W)]| = |\Pr[b = \widetilde{b} \text{ in Game 6}] - \Pr[b = \widetilde{b} \text{ in Game 7}]| = \big|\Pr[\mathtt{Suc}_6] - \Pr[\mathtt{Suc}_7]\big|$. □

For random values $q^*, r^*, r_1^*, r_2^* \in \mathbb{Z}_p$, the challenge signature in Game 7 is denoted by $\Sigma^* = (\{T_i^*\}_{i\in[1,4]}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot G^{r^*}, U^{r_1^*}, V^{r_2^*}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$. Therefore, the choice of the challenge bit b and the distribution of the challenge signature Σ^* are independent. Thus, we can say that $\Pr[\mathtt{Suc}_7] = 1/2$. From this fact and Lemma 8.3.1 to Lemma 8.3.7, we get

$$\begin{aligned}\mathsf{Adv}_{\Pi_{\mathsf{FI}},\mathcal{A}}^{anon}(\lambda) &\leq \sum_{\ell=0}^{6} \big|\Pr[\mathtt{Suc}_\ell] - \Pr[\mathtt{Suc}_{\ell+1}]\big| + \big|\Pr[\mathtt{Suc}_7] - 1/2\big| \\ &\leq \mathsf{Adv}_{\mathcal{B}_1}^{DDH}(\lambda) + \mathsf{Adv}_{\mathcal{B}_2}^{DDH}(\lambda) + \frac{q_H(1 + q_{\mathsf{open}}) + 1}{p}.\end{aligned}$$

Since q_H and q_{open} are polynomial in λ and p is exponential in λ, we see that $(q_H(1+q_{\mathsf{open}})+1)/p$ is negligible in λ. Therefore, if the adversary does not generate related queries, Mechanism 6 satisfies anonymity under the DDH assumption in the random oracle model. □

8.3.2 Analysis of Possible Attacks

From the result of the previous section, we see that the only way to break the anonymity of Mechanism 6 is generating a related query. Therefore in this section, we analyze all the cases of a related query and find out the cases in which the adversary might generate it.

Let m^* and $\Sigma^* = (\{T_i^*\}_{i\in[1,4]}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$ be the challenge message and the challenge signature, respectively. Let $(\widetilde{m}, \widetilde{\Sigma} = (\{\widetilde{T_i}^*\}_{i\in[1,4]}, \widetilde{c}, \widetilde{\sigma}_x, \widetilde{\sigma}_y, \widetilde{\sigma}_\delta, \widetilde{\sigma}_q, \widetilde{\sigma}_r))$ be a related query. From the definition of a related query it holds that $(\{\widetilde{T_i}\}_{i\in[1,4]}, \{\widetilde{R_i}\}_{i\in[1,4]}, \widetilde{m}) = (\{T_i^*\}_{i\in[1,4]}, \{R_i^*\}_{i\in[1,4]}, m^*)$. Moreover, since $(\widetilde{m}, \widetilde{\Sigma}) \neq (m^*, \Sigma^*)$ holds, it is required that $(\widetilde{\sigma}_x, \widetilde{\sigma}_y, \widetilde{\sigma}_\delta, \widetilde{\sigma}_q, \widetilde{\sigma}_r) \neq (\sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$. That is,

$$\widetilde{\sigma}_x \neq \sigma_x^* \ \vee\ \widetilde{\sigma}_y \neq \sigma_y^* \ \vee\ \widetilde{\sigma}_\delta \neq \sigma_\delta^* \ \vee\ \widetilde{\sigma}_q \neq \sigma_q^* \ \vee\ \widetilde{\sigma}_r \neq \sigma_r^*$$

holds. Therefore, we have 31 (= {the first part is changed or not} × {the second part is changed or not}×···×{the last part is changed or not}−{any parts are not changed} = $2^5 - 1$) cases of a related query.

Although there are many cases, we can narrow down to seven cases. From the equation $\widetilde{R}_3 = R_3^*$, it holds that $\widetilde{R}_3 = R_3^* \Leftrightarrow U^{\widetilde{\sigma}_r} \cdot {T_3}^{-c} = U^{\sigma_r^*} \cdot (T_3^*)^{-c^*} \Leftrightarrow U^{\widetilde{\sigma}_r} \cdot (T_3^*)^{-c^*} = U^{\sigma_r^*} \cdot (T_3^*)^{-c^*} \Leftrightarrow U^{\widetilde{\sigma}_r} = U^{\sigma_r^*} \Leftrightarrow u\widetilde{\sigma}_r = u\sigma_r^*$. Since $u \in \mathbb{Z}_p^*$, we get $\widetilde{\sigma}_r = \sigma_r^*$. In a similar way, we get $\widetilde{\sigma}_x = \sigma_x^*$ from the equation $\widetilde{R}_2 = R_2^*$. That is, it ultimately holds that

$$\widetilde{\sigma}_y \neq \sigma_y^* \ \vee\ \widetilde{\sigma}_\delta \neq \sigma_\delta^* \ \vee\ \widetilde{\sigma}_q \neq \sigma_q^*.$$

Thus, we can narrow down to seven ($=2^3 - 1$) cases of a related query described in Table 8.1. Here, we classify these cases into the following types:

(a) $\widetilde{\sigma}_y \neq \sigma_y^*$ ($\widetilde{\sigma}_\delta$ and $\widetilde{\sigma}_q$ are arbitrary), (b) $\widetilde{\sigma}_y = \sigma_y^* \ \wedge\ \widetilde{\sigma}_\delta \neq \sigma_\delta^* \ \wedge\ \widetilde{\sigma}_q = \sigma_q^*$,
(c) $\widetilde{\sigma}_y = \sigma_y^* \ \wedge\ \widetilde{\sigma}_\delta = \sigma_\delta^* \ \wedge\ \widetilde{\sigma}_q \neq \sigma_q^*$, ($\star$) $\widetilde{\sigma}_y = \sigma_y^* \ \wedge\ \widetilde{\sigma}_\delta \neq \sigma_\delta^* \ \wedge\ \widetilde{\sigma}_q \neq \sigma_q^*$.

Then, we analyze each type. Specifically, the query described in Section 8.2 as an attack for Mechanism 6 is in Type ($\star$).

Now, we examine the related queries in Type (a), (b), and (c). In fact, the adversary can generate the these types of queries with only negligible probability. In the following,

$\widetilde{\sigma}_y \stackrel{?}{=} \sigma_y^*$	$\widetilde{\sigma}_\delta \stackrel{?}{=} \sigma_\delta^*$	$\widetilde{\sigma}_q \stackrel{?}{=} \sigma_q^*$	**Type**
No	Yes	Yes	(a)
No	Yes	No	(a)
No	No	Yes	(a)
No	No	No	(a)
Yes	No	Yes	(b)
Yes	Yes	No	(c)
Yes	No	No	($\star$)

Table 8.1: Possible Cases of Related Queries

we explain the intuition of this fact.

Let $\mathcal{A}$ be the adversary who attacks the anonymity of Mechanism 6. We note that for any related query, it holds that

$$e(K, G_2)^{\widetilde{\sigma}_\delta} \cdot e(K, Y)^{-\widetilde{\sigma}_q} \cdot e(T_1^*, G_2)^{\widetilde{\sigma}_y} = e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \tag{8.2}$$

since the equation $\widetilde{R}_1 = R_1^*$ holds. From this equation, we can get the following observations on the related queries in Type (a), (b), and (c).

Type (a): We consider the situation that $\mathcal{A}$ generates a related query $(m^*, \Sigma = (\{T_i^*\}_{i\in[1,4]}, c^*, \sigma_x^*, \widetilde{\sigma}_y, \widetilde{\sigma}_\delta, \widetilde{\sigma}_q, \sigma_r^*))$ in Type (a). That is, $\widetilde{\sigma}_y \neq \sigma_y^*$ holds (here, we say nothing whether $\widetilde{\sigma}_\delta \neq \sigma_\delta^*$ and $\widetilde{\sigma}_q \neq \sigma_q^*$). Let $T_1^* = {G_1}^t$, $K = {G_1}^k$, and $H = {G_1}^h$. From Equation (8.2), it holds that

$$\begin{aligned}
&e(K, G_2)^{\widetilde{\sigma}_\delta} \cdot e(K, Y)^{-\widetilde{\sigma}_q} \cdot e(T_1^*, G_2)^{\widetilde{\sigma}_y} = e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \\
&\Leftrightarrow e({G_1}^k, G_2)^{\widetilde{\sigma}_\delta} \cdot e({G_1}^k, {G_2}^w)^{-\widetilde{\sigma}_q} \cdot e({G_1}^t, G_2)^{\widetilde{\sigma}_y} \\
&\qquad\qquad = e({G_1}^k, G_2)^{\sigma_\delta^*} \cdot e({G_1}^k, {G_2}^w)^{-\sigma_q^*} \cdot e({G_1}^t, G_2)^{\sigma_y^*} \\
&\Leftrightarrow e(G_1, G_2)^{k\widetilde{\sigma}_\delta - kw\widetilde{\sigma}_q + t\widetilde{\sigma}_y} = e(G_1, G_2)^{k\sigma_\delta^* - kw\sigma_q^* + t\sigma_y^*} \\
&\Leftrightarrow k\widetilde{\sigma}_\delta - kw\widetilde{\sigma}_q + t\widetilde{\sigma}_y = k\sigma_\delta^* - kw\sigma_q^* + t\sigma_y^* \\
&\Leftrightarrow t = k\frac{w\Delta\sigma_q - \Delta\sigma_\delta}{\Delta\sigma_y} \quad \left(\because \widetilde{\sigma}_y \neq \sigma_y^*\right)
\end{aligned}$$

where $\Delta\sigma_\delta = \widetilde{\sigma}_\delta - \sigma_\delta^*$, $\Delta\sigma_q = \widetilde{\sigma}_q - \sigma_q^*$, and $\Delta\sigma_y = \widetilde{\sigma}_y - \sigma_y^*$. Moreover, since $T_1^* = A_{i_b} \cdot K^{q^*} = \left(\frac{G_1}{H^{x_{i_b}} \cdot K^{z_{i_b}}}\right)^{\frac{1}{w+y_{i_b}}} \cdot K^{q^*} = \left(\frac{G_1}{{G_1}^{hx_{i_b}} \cdot {G_1}^{kz_{i_b}}}\right)^{\frac{1}{w+y_{i_b}}} \cdot {G_1}^{kq^*}$ holds, it holds that

$$t = \log_{G_1} T_1^* = \frac{1}{w + y_{i_b}}\left(1 - hx_{i_b} - kz_{i_b}\right) + kq^*.$$

From these two equations, we get

$$k\frac{w\Delta\sigma_q - \Delta\sigma_\delta}{\Delta\sigma_y} = \frac{1}{w + y_{i_b}}\big(1 - hx_{i_b} - kz_{i_b}\big) + kq^*. \tag{8.3}$$

From a viewpoint of the challenger who executes the anonymity game with $\mathcal{A}$, the challenger knows the values w and $(y_{i_b}, x_{i_b}, z_{i_b})$ since he generates the issuing key and all the signing keys of honest users by himself. Also, q^* is chosen by the challenger. Moreover, the challenger can compute $\Delta\sigma_\delta = \widetilde{\sigma}_\delta - \sigma^*_\delta$, $\Delta\sigma_q = \widetilde{\sigma}_q - \sigma^*_q$, and $\Delta\sigma_y = \widetilde{\sigma}_y - \sigma^*_y$ from the values $\widetilde{\sigma}_\delta$, $\widetilde{\sigma}_q$, and $\widetilde{\sigma}_y$ which are the part of the related query, and the values σ^*_δ, σ^*_q, and σ^*_y which are the part of the challenge signature. The challenger does not know the discrete logarithm of K in usual since the value K is randomly chosen from $\mathbb{G}_1$ in the GKg algorithm. However, if the challenger chooses $k \in \mathbb{Z}_p$ uniform randomly and sets $K = {G_1}^k$, he can know the discrete logarithm k. Now, the challenger knows all the values in Equation (8.3) except for h. This means that the challenger can compute the discrete logarithm h of $H \in \mathbb{G}_1$ from the values he knows. Thus, when $\mathcal{A}$ generates a related query in Type (a), the challenger can solve the DL problem in $\mathbb{G}_1$. That is, if the DL assumption holds in $\mathbb{G}_1$, the probability that $\mathcal{A}$ generates a related query in Type (a) is negligible.

Type (b): Let $K = {G_1}^k$. When the conditions $\widetilde{\sigma}_y = \sigma^*_y$ and $\widetilde{\sigma}_q = \sigma^*_q$ are put in Equation (8.2), we get $e(K, G_2)^{\widetilde{\sigma}_\delta} = e(K, G_2)^{\sigma^*_\delta} \Leftrightarrow e(G_1, G_2)^{k\widetilde{\sigma}_\delta} = e(G_1, G_2)^{k\sigma^*_\delta} \Leftrightarrow k\widetilde{\sigma}_\delta = k\sigma^*_\delta$. If $k \neq 0$, $\widetilde{\sigma}_\delta = \sigma^*_\delta$ holds. However, since this contradicts $\widetilde{\sigma}_\delta \neq \sigma^*_\delta$ that is the condition of Type (b), a related query in Type (b) does not exist if $k \neq 0$. On the other hand, the probability that $k = 0$ holds is $1/p$ since $K \in \mathbb{G}_1$ is chosen uniform randomly. Therefore, the probability that $\mathcal{A}$ generates a related query in Type (b) is at most $1/p$ which is negligible.

Type (c): Let $K = {G_1}^k$. When the conditions $\widetilde{\sigma}_y = \sigma^*_y$ and $\widetilde{\sigma}_\delta = \sigma^*_\delta$ are put in Equation (8.2), we get $e(K, Y)^{-\widetilde{\sigma}_q} = e(K, Y)^{-\sigma^*_q} \Leftrightarrow e(G_1, G_2)^{-kw\widetilde{\sigma}_q} = e(G_1, G_2)^{-kw\sigma^*_q} \Leftrightarrow kw\widetilde{\sigma}_q = kw\sigma^*_q$. If $k \neq 0$ and $w \neq 0$, $\widetilde{\sigma}_q = \sigma^*_q$ holds. However, since this contradicts $\widetilde{\sigma}_q \neq \sigma^*_q$ that is the condition of Type (c), a related query in Type (c) does not exist if $k \neq 0$ and $w \neq 0$. On the other hand, the probability that $k = 0$ or $w = 0$ satisfies $\Pr[k = 0 \vee w = 0] \leq \Pr[k = 0] + \Pr[w = 0] = 2/p$ since $K \in \mathbb{G}_1$ and $w \in \mathbb{Z}_p$ are chosen uniform randomly. Therefore, the probability that $\mathcal{A}$ generates a related query in Type (c) is at most $2/p$ which is negligible.

Therefore, we see that the probability that $\mathcal{A}$ generates the related queries in Type (a), (b), and (c) is negligible if the DL assumption holds in $\mathbb{G}_1$.

On the other hand, we cannot rule out the possibility that the adversary generates a related query in Type ($\star$) since our attack is in this type. Now, we further analyze this type of query. This type of query satisfies $\widetilde{\sigma}_y = \sigma^*_y$. When this equation is put in Equation (8.2), we get

$$\begin{aligned} e(K,G_2)^{\widetilde{\sigma}_\delta} \cdot e(K,Y)^{-\widetilde{\sigma}_q} &= e(K,G_2)^{\sigma^*_\delta} \cdot e(K,Y)^{-\sigma^*_q} \\ &\Leftrightarrow e(G_1,G_2)^{k\widetilde{\sigma}_\delta} \cdot e(G_1,G_2)^{-kw\widetilde{\sigma}_q} = e(G_1,G_2)^{k\sigma^*_\delta} \cdot e(G_1,G_2)^{-kw\sigma^*_q} \\ &\Leftrightarrow k(\widetilde{\sigma}_\delta - w\widetilde{\sigma}_q) = k(\sigma^*_\delta - w\sigma^*_q). \end{aligned}$$

Since the probability that $k = 0$ holds is $1/p$, it holds that $k \neq 0$ with high probability. If $k \neq 0$, we get

$$\widetilde{\sigma}_\delta - w\widetilde{\sigma}_q = \sigma^*_\delta - w\sigma^*_q \;\Leftrightarrow\; w = (\widetilde{\sigma}_\delta - \sigma^*_\delta)/(\widetilde{\sigma}_q - \sigma^*_q).$$

That is, the issuing key w can be computed from the values σ^*_δ and σ^*_q in the challenge signature Σ^* and the values $\widetilde{\sigma}_\delta$ and $\widetilde{\sigma}_q$ in the related query. Therefore, this indicates that the adversary who can generate a related query in Type ($\star$) knows the issuing key.

From the above observations, we see that only a related query in Type ($\star$) might be generated by the adversary. Furthermore, the adversary generating this type of query knows the issuing key. Therefore, the minimum condition of breaking the anonymity of Mechanism 6 seems to be that the adversary knows the issuing key. Thus, we can expect that *if the adversary does not possess the issuing key, Mechanism 6 satisfies anonymity.*

8.3.3 Definition of Weak Anonymity

In this section, we formalize the conditions which we characterized under which the anonymity of Mechanism 6 is preserved in the previous section. Concretely, we introduce a new security definition of anonymity called "weak anonymity", where the adversary is not allowed to corrupt the issuer.

We give the definition of weak anonymity by using the oracles in Section 4.2. Intuitively, weak anonymity ensures that the adversary who corrupts all the users but not the issuer cannot extract the signer's information from a signature. Formally, it is defined as follows.

Definition 8.3.1 (Weak Anonymity)**.** *Let $\mathcal{A}$ be an adversary for weak anonymity. We define the experiment $\mathsf{Exp}^{w\text{-}anon}_{\mathcal{GS},\mathcal{A}}(\lambda)$ as follows.*

$$\begin{aligned} \mathsf{Exp}^{w\text{-}anon}_{\mathcal{GS},\mathcal{A}}(\lambda): \quad & b \leftarrow \{0,1\};\ (\mathsf{gpk},\mathsf{ik},\mathsf{ok}) \leftarrow \mathsf{GS.GGen}(1^k);\ \mathsf{CU} \leftarrow \emptyset;\ \mathsf{HU} \leftarrow \emptyset \\ & \widetilde{b} \leftarrow \mathcal{A}^{\mathsf{AddU}(\cdot),\mathsf{Corrupt}(\cdot,\cdot),\mathsf{SndToI}(\cdot,\cdot),\mathsf{USK}(\cdot),\mathsf{RReg}(\cdot),\mathsf{Ch}(b,\cdot,\cdot,\cdot),\mathsf{Open}(\cdot,\cdot)}(\mathsf{gpk}) \end{aligned}$$

Return 1 if $\widetilde{b} = b$, otherwise return 0

We say that $\mathcal{GS}$ satisfies weak anonymity if the advantage

$$\mathsf{Adv}^{w\text{-}anon}_{\mathcal{GS},\mathcal{A}} := \left| \Pr[\mathsf{Exp}^{w\text{-}anon}_{\mathcal{GS},\mathcal{A}}(\lambda) = 1] - 1/2 \right|$$

is negligible for any PPT adversary $\mathcal{A}$.

8.3.4 Proof of the Weak Anonymity

Mechanism 6 satisfies weak anonymity as shown in Theorem 8.3.2. This theorem implies that *Mechanism 6 is still secure under the condition that the issuer does not join the attack*. Such a condition is reasonable if a single authority plays roles of both the opener and the issuer.

We note that most of the proof is the same as that of the anonymity under the restricted condition (given in Section 8.3.1) since anonymity in Definition 4.2.2 implies weak anonymity. However, since it is not assumed that the adversary does not generate a related query in the proof of the weak anonymity, we cannot straightforwardly prove the part corresponding to Game 5 $\approx$ Game 6 in the proof of the anonymity.

In the proof of the weak anonymity, we rule out the possibility that the adversary generates a related query by the computational assumptions. As we observe in Section 8.3.2, the adversary cannot generate related queries in Type (a), (b), and (c) under the DL assumption. Also in the proof, we prove that the adversary who does not possess the issuing key cannot generate related queries in Type ($\star$) under the q-SDH assumption. This part is the most difficult in this proof since the reduction algorithm needs to deal with generating user signing keys without the issuing key. To overcome this problem, we apply the rewinding technique as in the forking lemma [96] in our security proof.

Theorem 8.3.2. *Mechanism 6 satisfies weak anonymity under the DL assumption in the group $\mathbb{G}_1$, the DDH assumption in the group $\mathbb{G}$, and the q-SDH assumption in the groups $(\mathbb{G}_1, \mathbb{G}_2)$ in the random oracle model.*

Proof. Let $\mathcal{A}$ be an adversary that attacks the weak anonymity of Mechanism 6. We consider the following sequence of games. Let b be the challenge bit, m^* be the challenge message, i_0, i_1 be the challenge users, and $\Sigma^* = (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$ be the challenge signature. Let $\mathtt{Suc}_\ell$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game ℓ. Also, we specify the random tape of $\mathcal{A}$ in the proof when we use the rewinding technique.

[Game 0]: This is the experiment $\mathsf{Exp}^{w\text{-}anon}_{\mathcal{GS},\mathcal{A}}(\lambda)$ itself. As in Game 0 of Theorem 8.3.1, The challenger manages an inout/output pair of the random oracle in the list L, and we

assume that the adversary generates the involved random oracle query before he queries to the Open oracle.

[Game 1 - Game 5]: The modification of each game is the same as that of the game in Theorem 8.3.1.

[Game 6]: We change the way to generate $H, K \in \mathbb{G}_1$ in the group public key gpk. More precisely, in Game 6, the challenger samples $h, k \in \mathbb{Z}_p$ uniform randomly and sets $H \leftarrow {G_1}^h, K \leftarrow {G_1}^k$ by comparing Game 5 in which H, K are chosen uniform randomly in $\mathbb{G}_1$.

[Game 7]: In Game 7, if the adversary generates the opening query $(m^*, (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r^*))$ such that $\sigma_y \neq \sigma_y^*$, the challenger returns $\perp$ where σ_y and σ_q are arbitrary.

[Game 8]: In Game 8, if the adversary generates the opening query $(m^*, (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta, \sigma_q^*, \sigma_r^*))$ such that $\sigma_\delta \neq \sigma_\delta^*$, the challenger returns $\perp$.

[Game 9]: In Game 9, if the adversary generates the opening query $(m^*, (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q, \sigma_r^*))$ such that $\sigma_q \neq \sigma_q^*$, the challenger returns $\perp$.

[Game 10]: We modify the way to reply queries for the SndToI oracle. More precisely, the challenger replies the ℓ-th query $(i, (R_1, R_2))$ for the SndToI oracle as follows. Let N be a constant number.

Step 1 (Practice Phase). Execute other N anonymity games with the adversaries $\mathcal{A}_j$ who are the same as the original $\mathcal{A}$ in parallel where $1 \leq j \leq N$. Specifically, for $1 \leq j \leq N$, perform the following operations.

1. Sample $\widehat{c}_i^{(j)} \xleftarrow{r} \mathbb{Z}_p$.
2. Execute $\mathcal{A}_j(\mathsf{gpk}; \rho)$ where ρ is the random tape of the original $\mathcal{A}$. Then, the challenger makes exactly the same replies as those for the original $\mathcal{A}$ until the ℓ-th query $(i, (R_1, R_2))$ is generated. We note that the query is also the same as that of the original $\mathcal{A}$ since the challenger makes the same replies those for the original $\mathcal{A}$ until the ℓ-th query is generated.
3. Send $\widehat{c}_i^{(j)}$ to $\mathcal{A}_j$ as the first reply of the ℓ-th query $(i, (R_1, R_2))$, and obtain $(\widehat{\sigma}_{x_i}^{(j)}, \widehat{\sigma}_{z_i'}^{(j)})$.

Step 2 (First Reply in the Original Game with $\mathcal{A}$). If $i \notin \mathsf{CU}$, return $\perp$. If $i \in \mathsf{CU}$, sample $c_i \xleftarrow{r} \mathbb{Z}_p$ and return c_i as the first reply of the send-to-issuer query $(i, (R_1, R_2))$. Then, obtain $(\sigma_{x_i}, \sigma_{z_i'})$ from $\mathcal{A}$.

Step 3 (Second Reply in the Original Game with $\mathcal{A}$). If $(\sigma_{x_i}, \sigma_{z_i'})$ is invalid, return $\perp$ as the second reply. If $(\sigma_{x_i}, \sigma_{z_i'})$ is valid, find the index $j^* \in [1, N]$ in

the replies from $\mathcal{A}_j$ in Step 1 such that $(\widehat{\sigma}_{x_i}^{(j^*)}, \widehat{\sigma}_{z_i'}^{(j^*)})$ is valid and $c_i \neq \widehat{c}_i^{(j^*)}$. If there is no such index j^*, return $\perp$. On the other hand if there exists such index j^*, compute the second reply as follows.

1. Compute $\Delta\sigma_{x_i} \leftarrow \widehat{\sigma}_{x_i}^{(j^*)} - \sigma_{x_i}$, $\Delta\sigma_{z_i'} \leftarrow \widehat{\sigma}_{z_i'}^{(j^*)} - \sigma_{z_i'}$, and $\Delta c_i \leftarrow \widehat{c}_i^{(j^*)} - c_i$, and set $\widetilde{x}_i \leftarrow \Delta\sigma_{x_i}/\Delta c_i$ and $\widetilde{z}_i' \leftarrow \Delta\sigma_{z_i'}/\Delta c_i$.
2. Sample $y_i, z_i'' \in \mathbb{Z}_p$ uniform randomly and compute $C_i \leftarrow G_1^{\frac{1}{w+y_i}}$.
3. Compute $A_i \leftarrow C_i^{1-h\widetilde{x}_i - k(\widetilde{z}_i' + z_i'')}$. Then, set $\mathsf{cert}_i \leftarrow (A_i, y_i, z_i'')$ and reply cert_i as the second reply. Register $\mathsf{reg}[i] \leftarrow Q_i$.

[Game 11]: We modify the way to compute the element A_i in the simulation of the AddU oracle. In Game 11, A_i is computed as follows. The challenger chooses a random value y_i and sets $C_i \leftarrow G_1^{\frac{1}{w+y_i}}$ where w is the issuing key. Then, he also samples a random value z_i'' and sets $A_i \leftarrow C_i^{1-hx_i-k(z_i'+z_i'')}$ where $\mathsf{usk}_i = (x_i, z_i')$, $h = \log_{G_1} H$, and $k = \log_{G_1} K$.

[Game 12]: In Game 12, if the adversary generates the opening query $(m^*, (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta, \sigma_q, \sigma_r^*))$ such that $\sigma_y = \sigma_y^*$, $\sigma_\delta \neq \sigma_\delta^*$, and $\sigma_q \neq \sigma_q^*$, the challenger returns $\perp$.

[Game 13, Game 14]: The modification of each game is the same as that of Game 6 and Game 7 in Theorem 8.3.1, respectively.

For the advantage $\mathsf{Adv}_{\Pi_{\mathsf{FI}},\mathcal{A}}^{w\text{-}anon}(\lambda)$,

$$\mathsf{Adv}_{\Pi_{\mathsf{FI}},\mathcal{A}}^{w\text{-}anon}(\lambda) = \left|\Pr[\mathtt{Suc}_0] - 1/2\right| \leq \sum_{\ell=0}^{13} \left|\Pr[\mathtt{Suc}_\ell] - \Pr[\mathtt{Suc}_{\ell+1}]\right| + \left|\Pr[\mathtt{Suc}_{14}] - 1/2\right|$$

holds. Moreover, the following lemmas hold.

Lemma 8.3.8. *Let q_H be the number of $\mathcal{A}$'s random oracle queries. Then, it holds that $\left|\Pr[\mathtt{Suc}_0] - \Pr[\mathtt{Suc}_1]\right| \leq q_H/p$ for any PPT $\mathcal{A}$.*

Lemma 8.3.9. *It holds that $\Pr[\mathtt{Suc}_1] = \Pr[\mathtt{Suc}_2]$ for any PPT $\mathcal{A}$.*

Lemma 8.3.10. *Let q_H and q_{open} be the number of $\mathcal{A}$'s random oracle queries and opening queries, respectively. Then, it holds that $\left|\Pr[\mathtt{Suc}_2] - \Pr[\mathtt{Suc}_3]\right| \leq q_H \cdot q_{\mathsf{open}}/p$ for any PPT $\mathcal{A}$.*

Lemma 8.3.11. *It holds that $\Pr[\mathtt{Suc}_3] = \Pr[\mathtt{Suc}_4]$ for any PPT $\mathcal{A}$.*

Lemma 8.3.12. *There exists a* PPT *algorithm $\mathcal{B}_1$ such that $\left|\Pr[\mathtt{Suc}_4] - \Pr[\mathtt{Suc}_5]\right| = \mathsf{Adv}_{\mathcal{B}_1}^{DDH}(\lambda)$*

Lemma 8.3.8 to 8.3.12 can be proved as in the case of the anonymity (given in Section 8.3.1) since the modification of each game is the same as that of the game in Theorem 8.3.1. Therefore, we omit these proofs.

Lemma 8.3.13. *It holds that* $\Pr[\mathtt{Suc}_5] = \Pr[\mathtt{Suc}_6]$ *for any PPT* $\mathcal{A}$.

Proof. The difference between Game 5 and Game 6 is the way to generate the values $H, K \in \mathbb{G}_1$. More precisely, H, K are chosen from $\mathbb{G}_1$ uniform randomly in Game 5. On the other hand in Game 6, the challenger chooses $h, k \in \mathbb{Z}_p$ uniform randomly and sets $H \leftarrow {G_1}^h, K \leftarrow {G_1}^k$. However, since the distribution of H, K is uniform in $\mathbb{G}_1$ in each game, Game 5 and Game 6 are identical. Thus, it holds that $\Pr[\mathtt{Suc}_5] = \Pr[\mathtt{Suc}_6]$. □

Lemma 8.3.14. *There exists a* PPT *algorithm* $\mathcal{B}_2$ *such that* $\left|\Pr[\mathtt{Suc}_6] - \Pr[\mathtt{Suc}_7]\right| \leq \mathsf{Adv}^{DL}_{\mathcal{B}_2}(\lambda)$ *for any PPT* $\mathcal{A}$.

Proof. We define the event $\mathtt{Bad}^{(\mathsf{a})}_\ell$ as follows.

$\mathtt{Bad}^{(\mathsf{a})}_\ell$**:** The event that the adversary $\mathcal{A}$ generates the related query in Type (a) to the Open oracle in Game ℓ.

Game 6 and Game 7 are identical unless the events $\mathtt{Bad}^{(\mathsf{a})}_6$ and $\mathtt{Bad}^{(\mathsf{a})}_7$ occur. Therefore, we get $\left|\Pr[\mathtt{Suc}_6] - \Pr[\mathtt{Suc}_7]\right| \leq \Pr[\mathtt{Bad}^{(\mathsf{a})}_7]$ same as in Lemma 8.3.1.

In the following, we construct the algorithm $\mathcal{B}_2$ who tries to solve the DL problem in $\mathbb{G}_1$ by using $\mathcal{A}$ and estimate the probability $\Pr[\mathtt{Bad}^{(\mathsf{a})}_7]$. First, $\mathcal{B}_2$ receives the DL tuple $G_1, H \in \mathbb{G}_1$ and $G_2 \in \mathbb{G}_2$. Now, $\mathcal{B}_2$'s goal is to compute the value $\log_{G_1} H$. Next, $\mathcal{B}_2$ generates the instance of the weak anonymity game. Here for $G_1 \in \mathbb{G}_1, G_2 \in \mathbb{G}_2$ and $H \in \mathbb{G}_1$, he uses the ones in the instance of the DL problem. Also, $\mathcal{B}_2$ samples $k \in \mathbb{Z}_p$ uniform randomly and sets $K = {G_1}^k \in \mathbb{G}_1$. Other elements are generated by following the GKg algorithm. Let $\mathsf{gpk} = (G_1, G_2, G, \mathsf{H}, H, K, Y, U, V)$, $\mathsf{ik} = w$, and $\mathsf{ok} = (u, v)$, $\mathcal{B}_2$ sends gpk to the adversary $\mathcal{A}$. Since possessing all the keys which are needed to reply oracle queries, $\mathcal{B}_2$ can simulate the replies of all the queries. Especially, $\mathcal{B}_2$ generates the challenge signature as follows:

1. Choose $q^*, r^*, r_2^* \in \mathbb{Z}_p$ uniform randomly and compute $(T_1^*, T_2^*, T_3^*, T_4^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot G^{r^*}, U^{r^*}, G^{r_2^*})$.

2. Choose $\sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^* \in \mathbb{Z}_p$ and $c^* \in \mathbb{Z}_p$ uniform randomly, and computes $R_1^* = e(H, G_2)^{\sigma_x^*} \cdot e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \cdot \left(\frac{e(G_1, G_2)}{e(T_1^*, Y)}\right)^{-c^*}$, $R_2^* = G^{\sigma_x^* + \sigma_r^*} \cdot {T_2^*}^{-c^*}$, $R_3^* = U^{\sigma_r^*} \cdot {T_3^*}^{-c^*}$, and $R_4^* = V^{\sigma_r^*} \cdot {T_4^*}^{-c^*}$.

3. If the value $((\mathsf{gpk}, T_1^*, T_2^*, T_3^*, T_4^*, R_1^*, R_2^*, R_3^*, R_4^*, m^*), \cdot)$ is not defined in the list L, the value $((\mathsf{gpk}, T_1^*, T_2^*, T_3^*, T_4^*, R_1^*, R_2^*, R_3^*, R_4^*, m^*), c^*)$ is added to L and the challenge signature Σ^* is set to be $(T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$. On the other hand, if such a value is already defined, the challenge signature is set to be $\perp$.

Finally, $\mathcal{A}$ terminates with $\widetilde{b} \in \{0,1\}$.

When $\mathcal{A}$ generated the related query in Type (a) to the Open oracle, there is the Open query $(m^*, \Sigma = (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r^*))$ such that $\sigma_y \neq \sigma_y^*$ and $\mathsf{GVf}(\mathsf{gpk}, m^*, \Sigma) = 1$. For this query, $\mathcal{B}_2$ computes

$$h = \frac{1}{x_{i_b}}\Big(1 - kz_{i_b} + (w + y_{i_b})(kq^* + k\frac{\Delta\sigma_\delta - w\Delta\sigma_q}{\Delta\sigma_y})\Big)$$

where $\Delta\sigma_\delta = \sigma_\delta - \sigma_\delta^*$, $\Delta\sigma_q = \sigma_q - \sigma_q^*$, and $\Delta\sigma_y = \sigma_y - \sigma_y^*$. Then, he outputs h as the solution of the DL problem. Since i_b is an honest user, that is, $i_b \in \mathsf{HU}$ holds by the condition of the challenge query, $\mathcal{B}_2$ knows the i_b's signing key $\mathsf{gsk}_{i_b} = (A_{i_b}, y_{i_b}, z_{i_b}, x_{i_b}, Q_{i_b})$ and can compute the above h.

Now, we show the value h is the discrete logarithm of H in the following. Let ${T_1^*} = {G_1}^t$. Since a related query satisfies Equation (8.2), it holds that

$$\begin{aligned}
&e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} \cdot e(T_1^*, G_2)^{\sigma_y} = e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \\
&\quad \Leftrightarrow e({G_1}^k, G_2)^{\sigma_\delta} \cdot e({G_1}^k, {G_2}^w)^{-\sigma_q} \cdot e({G_1}^t, G_2)^{\sigma_y} \\
&\qquad\qquad = e({G_1}^k, G_2)^{\sigma_\delta^*} \cdot e({G_1}^k, {G_2}^w)^{-\sigma_q^*} \cdot e({G_1}^t, G_2)^{\sigma_y^*} \\
&\quad \Leftrightarrow e(G_1, G_2)^{k\sigma_\delta - kw\sigma_q + t\sigma_y} = e(G_1, G_2)^{k\sigma_\delta^* - kw\sigma_q^* + t\sigma_y^*} \\
&\quad \Leftrightarrow k\sigma_\delta - kw\sigma_q + t\sigma_y = k\sigma_\delta^* - kw\sigma_q^* + t\sigma_y^* \\
&\quad \Leftrightarrow t = k\frac{w\Delta\sigma_q - \Delta\sigma_\delta}{\Delta\sigma_y}.
\end{aligned}$$

Therefore, we get

$$\begin{aligned}
{G_1}^h &= {G_1}^{\frac{1}{x_{i_b}}\left(1 - kz_{i_b} + (w+y_{i_b})(kq^* + k\frac{\Delta\sigma_\delta - w\Delta\sigma_q}{\Delta\sigma_y})\right)} \\
&= {G_1}^{\frac{1}{x_{i_b}}\left(1 - kz_{i_b} + (w+y_{i_b})(kq^* - t)\right)} \\
&= \Big(G_1 \cdot K^{-z_{i_b}} \cdot (K^{q^*} \cdot (T_1^*)^{-1})^{(w+y_{i_b})}\Big)^{\frac{1}{x_{i_b}}} \\
&= \Big(\frac{G_1 \cdot K^{q^*(w+y_{i_b})}}{K^{z_{i_b}} \cdot (A_{i_b} \cdot K^{q^*})^{(w+y_{i_b})}}\Big)^{\frac{1}{x_{i_b}}} \quad (\because T_1^* = A_{i_b} \cdot K^{q^*}) \\
&= \left(\frac{G_1}{K^{z_{i_b}} \cdot \Big(\big(\frac{G_1}{H^{x_{i_b}} \cdot K^{z_{i_b}}}\big)^{\frac{1}{w+y_{i_b}}}\Big)^{(w+y_{i_b})}}\right)^{\frac{1}{x_{i_b}}} \quad \left(\because A_{i_b} = \big(\frac{G_1}{H^{x_{i_b}} \cdot K^{z_{i_b}}}\big)^{\frac{1}{w+y_{i_b}}}\right) \\
&= H.
\end{aligned}$$

Thus, h is the discrete logarithm of H. That is, if the event $\mathtt{Bad}_7^{(\mathsf{a})}$ occurs, $\mathcal{B}_2$ can solve the DL problem, and we get $\Pr[\mathtt{Bad}_7^{(\mathsf{a})}] \leq \mathsf{Adv}_{\mathcal{B}_2}^{DL}(\lambda)$. Finally, we get $\Big|\Pr[\mathtt{Suc}_6] - \Pr[\mathtt{Suc}_7]\Big| \leq \Pr[\mathtt{Bad}_7^{(\mathsf{a})}] \leq \mathsf{Adv}_{\mathcal{B}_2}^{DL}(\lambda)$. □

Lemma 8.3.15. *It holds that* $\left|\Pr[\mathtt{Suc}_7] - \Pr[\mathtt{Suc}_8]\right| \leq 1/p$ *for any PPT* $\mathcal{A}$.

Proof. We define the event $\mathtt{Bad}_\ell^{(\mathsf{b})}$ as follows.

$\mathtt{Bad}_\ell^{(\mathsf{b})}$**:** The event that the adversary $\mathcal{A}$ generates the related query in Type (b) to the Open oracle in Game ℓ.

Game 7 and Game 8 are identical unless the events $\mathtt{Bad}_7^{(\mathsf{b})}$ and $\mathtt{Bad}_8^{(\mathsf{b})}$ occur. Therefore, we get $\left|\Pr[\mathtt{Suc}_7] - \Pr[\mathtt{Suc}_8]\right| \leq \Pr[\mathtt{Bad}_8^{(\mathsf{b})}]$ same as in Lemma 8.3.1.

We estimate the probability $\Pr[\mathtt{Bad}_8^{(\mathsf{b})}]$ in the following. Let $K = {G_1}^k$. When the conditions $\sigma_y = \sigma_y^*$ and $\sigma_q = \sigma_q^*$ are put in Equation (8.2), we get $e(K, G_2)^{\sigma_\delta} = e(K, G_2)^{\sigma_\delta^*} \Leftrightarrow e(G_1, G_2)^{k\sigma_\delta} = e(G_1, G_2)^{k\sigma_\delta^*} \Leftrightarrow k\sigma_\delta = k\sigma_\delta^*$. Therefore, a related query in Type (b) satisfies the equation $k\sigma_\delta = k\sigma_\delta^*$, and then $\sigma_\delta = \sigma_\delta^*$ holds if $k \neq 0$. However, this contradicts the condition of Type (b) (i.e., $\sigma_\delta \neq \sigma_\delta^*$). Thus, a related query in Type (b) does not exist if $k \neq 0$. On the other hand, the probability that $k = 0$ holds is $1/p$ since $K \in \mathbb{G}_1$ is chosen uniform randomly. Thus, the probability that $\mathcal{A}$ generates a related query in Type (b) is at most $1/p$, and $\Pr[\mathtt{Bad}_8^{(\mathsf{b})}] \leq 1/p$ holds. That is, we get $\left|\Pr[\mathtt{Suc}_7] - \Pr[\mathtt{Suc}_8]\right| \leq 1/p$. □

Lemma 8.3.16. *It holds that* $\left|\Pr[\mathtt{Suc}_8] - \Pr[\mathtt{Suc}_9]\right| \leq 2/p$ *for any PPT* $\mathcal{A}$.

Proof. We define the event $\mathtt{Bad}_\ell^{(\mathsf{c})}$ as follows.

$\mathtt{Bad}_\ell^{(\mathsf{c})}$**:** The event that the adversary $\mathcal{A}$ generates a related query in Type (c) to the Open oracle in Game ℓ.

Game 8 and Game 9 are identical unless the events $\mathtt{Bad}_8^{(\mathsf{c})}$ and $\mathtt{Bad}_9^{(\mathsf{c})}$ occur. Therefore, we get $\left|\Pr[\mathtt{Suc}_8] - \Pr[\mathtt{Suc}_9]\right| \leq \Pr[\mathtt{Bad}_9^{(\mathsf{c})}]$ in the same way as in Lemma 8.3.1.

We estimate the probability $\Pr[\mathtt{Bad}_9^{(\mathsf{c})}]$ in the following. Let $K = {G_1}^k$. When the conditions $\sigma_y = \sigma_y^*$ and $\sigma_\delta = \sigma_\delta^*$ are put in Equation (8.2), we get $e(K, Y)^{-\sigma_q} = e(K, Y)^{-\sigma_q^*} \Leftrightarrow e(G_1, G_2)^{-kw\sigma_q} = e(G_1, G_2)^{-kw\sigma_q^*} \Leftrightarrow kw\sigma_q = kw\sigma_q^*$. Therefore, a related query in Type (c) satisfies the equation $kw\sigma_q = kw\sigma_q^*$, and then $\sigma_q = \sigma_q^*$ holds if $k \neq 0$ and $w \neq 0$. However, this contradicts the condition of Type (c) (i.e., $\sigma_q \neq \sigma_q^*$). Thus, a related query in Type (c) does not exist if $k \neq 0$ and $w \neq 0$. On the other hand, the probability that $k = 0$ or $w = 0$ hold is at most $\Pr[k = 0 \vee w = 0] \leq \Pr[k = 0] + \Pr[w = 0] = 2/p$ since $K \in \mathbb{G}_1$ and $w \in \mathbb{Z}_p$ are chosen uniform randomly. Therefore, the probability that $\mathcal{A}$ generates a related query in Type (c) is at most $2/p$, and $\Pr[\mathtt{Bad}_9^{(\mathsf{c})}] \leq 2/p$ holds. That is, we get $\left|\Pr[\mathtt{Suc}_8] - \Pr[\mathtt{Suc}_9]\right| \leq 2/p$. □

Lemma 8.3.17. *Let* q_{iss} *be the number of* $\mathcal{A}$*'s send-to-issuer queries. Then, it holds that* $\left|\Pr[\mathtt{Suc}_9] - \Pr[\mathtt{Suc}_{10}]\right| \leq \sum_{\ell=1}^{q_{\mathsf{iss}}} \min\{(1 - \mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$ *for any PPT* $\mathcal{A}$.

Proof. We consider the following intermediate games $\overline{\text{Game}}\ 0, \ldots, \overline{\text{Game}}\ q_{\mathsf{iss}}$ to estimate the probability $\left|\Pr[\mathtt{Suc}_9] - \Pr[\mathtt{Suc}_{10}]\right|$.

[$\overline{\text{Game}}\ 0$]: This game is identical to Game 9.

[$\overline{\text{Game}}\ 1$]: We modify the way to reply the first send-to-issuer query. More precisely, the challenger replies the first send-to-issuer query by rewinding $\mathcal{A}$ as denoted in Game 10. After the first, the challenger replies send-to-issuer queries by following the Issue algorithm as in Game 9.

$\vdots$

[$\overline{\text{Game}}\ \ell$]: In this game, the challenger replies the first to ℓ-th send-to-issuer queries by rewinding $\mathcal{A}$. On the other hand, for the $(\ell+1)$-th to q_{iss}-th send-to-issuer queries, he replies by following the Issue algorithm.

$\vdots$

[$\overline{\text{Game}}\ q_{\mathsf{iss}}$]: In this game, the challenger replies all the send-to-issuer queries by rewinding $\mathcal{A}$. Thus, this game is identical to Game 10.

Let $\overline{\mathtt{Suc}}_\ell$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit in $\overline{\text{Game}}\ \ell$. Then, the following inequality holds:

$$\left|\Pr[\mathtt{Suc}_9] - \Pr[\mathtt{Suc}_{10}]\right| = \left|\Pr[\overline{\mathtt{Suc}}_0] - \Pr[\overline{\mathtt{Suc}}_{q_{\mathsf{iss}}}]\right| \leq \sum_{\ell=1}^{q_{\mathsf{iss}}} \left|\Pr[\overline{\mathtt{Suc}}_{\ell-1}] - \Pr[\overline{\mathtt{Suc}}_\ell]\right|. \tag{8.4}$$

Now, we prove that $\left|\Pr[\overline{\mathtt{Suc}}_{\ell-1}] - \Pr[\overline{\mathtt{Suc}}_\ell]\right| \leq \min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$ holds for $1 \leq \ell \leq q_{\mathsf{iss}}$ where N is the number of the parallel anonymity games with $\mathcal{A}_j$. The difference between $\overline{\text{Game}}\ \ell-1$ and $\overline{\text{Game}}\ \ell$ is the way to reply the ℓ-th SndToI query. In both games, the first reply of the ℓ-th SndToI query is chosen uniform randomly. Moreover, when $\mathcal{A}$'s output $(\sigma_{x_i}, \sigma_{z_i'})$ for the first reply is invalid, the second reply of the ℓ-th SndToI query will be $\perp$ in both games. Therefore, only when $\mathcal{A}$'s output $(\sigma_{x_i}, \sigma_{z_i'})$ for the first reply is valid, the challengers in $\overline{\text{Game}}\ \ell-1$ and $\overline{\text{Game}}\ \ell$ behave differently. In the following, we consider the case that $(\sigma_{x_i}, \sigma_{z_i'})$ is valid.

In $\overline{\text{Game}}\ \ell$, when $(\sigma_{x_i}, \sigma_{z_i'})$ is valid, the second reply is decided whether there exists the valid output $(\widehat{\sigma}_{x_i}^{(j^*)}, \widehat{\sigma}_{z_i'}^{(j^*)})$ which satisfies $\widehat{c}_i^{(j^*)} \neq c_i$ in the N executions of $\mathcal{A}$. More precisely, if there exists such an output, a certificate $\mathsf{cert}_i = (A_i, y_i, z_i'')$ will be returned. On the other hand, if there is no such an output, the second reply will be $\perp$. Now, we show that in the former case, the second reply cert_i in $\overline{\text{Game}}\ \ell$ is the same as that in $\overline{\text{Game}}\ \ell-1$. First of all, y_i and z_i'' are chosen uniform randomly in both games. Thus, the distribution of y_i and z_i'' in $\overline{\text{Game}}\ \ell$ is the same as that in $\overline{\text{Game}}\ \ell-1$. Next, we prove the value A_i in $\overline{\text{Game}}\ \ell$ is the same as that in $\overline{\text{Game}}\ \ell-1$. Since $(\sigma_{x_i}, \sigma_{z_i'})$ and

$(\widehat{\sigma}_{x_i}^{(j^*)}, \widehat{\sigma}_{z'_i}^{(j^*)})$ are valid, it holds that

$$R_2 = H^{\sigma_{x_i}} K^{\sigma_{z'_i}} / H_i^{c_i} \ \wedge \ R_2 = H^{\widehat{\sigma}_{x_i}^{(j^*)}} K^{\widehat{\sigma}_{z'_i}^{(j^*)}} / H_i^{\widehat{c}_i^{(j^*)}}.$$

Therefore, we get

$$H_i = H^{\frac{\Delta\sigma_{x_i}}{\Delta c_i}} \cdot K^{\frac{\Delta\sigma_{z'_i}}{\Delta c_i}} \tag{8.5}$$

where $\Delta\sigma_{x_i} = \widehat{\sigma}_{x_i}^{(j^*)} - \sigma_{x_i}$, $\Delta\sigma_{z'_i} = \widehat{\sigma}_{z'_i}^{(j^*)} - \sigma_{z'_i}$, and $\Delta c_i = \widehat{c}_i^{(j^*)} - c_i$. We note that $\Delta c_i \neq 0$ holds since $\widehat{c}_i^{(j^*)} \neq c_i$. From this equation, the value A_i in $\overline{\text{Game}}\ \ell$ satisfies that

$$\begin{aligned}
A_i &= C_i^{1-h\widetilde{x}_i - k(\widetilde{z}'_i + z''_i)} \\
&= \left(G_1^{\frac{1}{w+y_i}}\right)^{1-h\frac{\Delta\sigma_{x_i}}{\Delta c_i} - k\left(\frac{\Delta\sigma_{z'_i}}{\Delta c_i} + z''_i\right)} \\
&= \left(G_1^{1-h\frac{\Delta\sigma_{x_i}}{\Delta c_i} - k\left(\frac{\Delta\sigma_{z'_i}}{\Delta c_i} + z''_i\right)}\right)^{\frac{1}{w+y_i}} \\
&= \left(\frac{G_1}{G_i^{h\frac{\Delta\sigma_{x_i}}{\Delta c_i}} \cdot G_i^{k\left(\frac{\Delta\sigma_{z'_i}}{\Delta c_i} + z''_i\right)}}\right)^{\frac{1}{w+y_i}} \\
&= \left(\frac{G_1}{H^{\frac{\Delta\sigma_{x_i}}{\Delta c_i}} \cdot K^{\frac{\Delta\sigma_{z'_i}}{\Delta c_i}} \cdot K^{z''_i}}\right)^{\frac{1}{w+y_i}} \\
&= \left(\frac{G_1}{H_i \cdot K^{z''_i}}\right)^{\frac{1}{w+y_i}}. \qquad (\because \text{Equation(8.5)})
\end{aligned}$$

This is the same as the value A_i in $\overline{\text{Game}}\ \ell - 1$. Therefore, in the condition that there exists the valid output $(\widehat{\sigma}_{x_i}^{(j^*)}, \widehat{\sigma}_{z'_i}^{(j^*)})$ such that $\widehat{c}_i^{(j^*)} \neq c_i$, the second reply for the ℓ-th SndTol query in $\overline{\text{Game}}\ \ell$ is identical to that in $\overline{\text{Game}}\ \ell - 1$.

Now, we define the events $\mathtt{Bad}$ and $\widehat{\mathtt{Bad}}$ as follows.

$\mathtt{Bad}$: The event that there is no valid output $(\widehat{\sigma}_{x_i}^{(j^*)}, \widehat{\sigma}_{z'_i}^{(j^*)})$ which satisfies $\widehat{c}_i^{(j^*)} \neq c_i$ in the N executions of $\mathcal{A}$ in $\overline{\text{Game}}\ \ell$.

$\widehat{\mathtt{Bad}}$: The event that $\mathcal{A}$'s output $(\sigma_{x_i}, \sigma_{z'_i})$ for the first reply is valid in $\overline{\text{Game}}\ \ell$.

In the above discussion, when $(\sigma_{x_i}, \sigma_{z'_i})$ is invalid, the replies of the ℓ-th SndTol query in $\overline{\text{Game}}\ \ell - 1$ and $\overline{\text{Game}}\ \ell$ are identical. Also, when $(\sigma_{x_i}, \sigma_{z'_i})$ is valid and there exists the valid output $(\widehat{\sigma}_{x_i}^{(j^*)}, \widehat{\sigma}_{z'_i}^{(j^*)})$ which satisfies $\widehat{c}_i^{(j^*)} \neq c_i$ in the N executions of $\mathcal{A}_j$, the replies of the ℓ-th SndTol query in both games are identical. That is, $\overline{\text{Game}}\ \ell$ is identical to $\overline{\text{Game}}\ \ell - 1$ unless the events $\mathtt{Bad}$ and $\widehat{\mathtt{Bad}}$ occur, and $\Pr[\overline{\mathtt{Suc}}_{\ell-1}] = \Pr[\overline{\mathtt{Suc}}_{\ell} \wedge (\neg\mathtt{Bad} \vee \neg\widehat{\mathtt{Bad}})]$ holds. Therefore, we get $\left|\Pr[\overline{\mathtt{Suc}}_{\ell-1}] - \Pr[\overline{\mathtt{Suc}}_{\ell}]\right| = \left|\Pr[\overline{\mathtt{Suc}}_{\ell-1}] - (\Pr[\overline{\mathtt{Suc}}_{\ell} \wedge (\neg\mathtt{Bad} \vee$

$\neg\widehat{\mathtt{Bad}})] + \Pr[\overline{\mathtt{Suc}}_\ell \wedge (\mathtt{Bad} \wedge \widehat{\mathtt{Bad}})])\big| \leq \Pr[\overline{\mathtt{Suc}}_\ell \wedge (\mathtt{Bad} \wedge \widehat{\mathtt{Bad}})] \leq \Pr[\mathtt{Bad} \wedge \widehat{\mathtt{Bad}}]$. Since $\Pr[\mathtt{Bad} \wedge \widehat{\mathtt{Bad}}] \leq \Pr[\mathtt{Bad}]$ and $\Pr[\mathtt{Bad} \wedge \widehat{\mathtt{Bad}}] \leq \Pr[\widehat{\mathtt{Bad}}]$ hold, we get $\Pr[\mathtt{Bad} \wedge \widehat{\mathtt{Bad}}] \leq \min\{\Pr[\mathtt{Bad}], \Pr[\widehat{\mathtt{Bad}}]\}$. Since $\Pr[\mathtt{Bad}] = (1 - \mathsf{prob}_\ell)^N$ and $\Pr[\widehat{\mathtt{Bad}}] = \mathsf{prob}_\ell$ hold by the definition of the events, it holds that $\Pr[\mathtt{Bad} \wedge \widehat{\mathtt{Bad}}] \leq \min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$. Therefore, we get $\big|\Pr[\overline{\mathtt{Suc}}_{\ell-1}] - \Pr[\overline{\mathtt{Suc}}_\ell]\big| \leq \min\{(1 - \mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$.

From the above discussion and Equation (8.4),

$$\big|\Pr[\mathtt{Suc}_9] - \Pr[\mathtt{Suc}_{10}]\big| \leq \sum_{\ell=1}^{q_{\mathsf{iss}}} \big|\Pr[\overline{\mathtt{Suc}}_{\ell-1}] - \Pr[\overline{\mathtt{Suc}}_\ell]\big| = \sum_{\ell=1}^{q_{\mathsf{iss}}} \min\{(1 - \mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$$

holds. □

Lemma 8.3.18. *It holds that* $\Pr[\mathtt{Suc}_{10}] = \Pr[\mathtt{Suc}_{11}]$ *for any PPT* $\mathcal{A}$.

Proof. The difference between Game 10 and Game 11 is the way to compute the element $\mathcal{A}_i$ in the simulation of the AddU oracle. However, the value $\mathcal{A}_i$ generated in Game 11 is identical to that in Game 10 since it holds that $A_i = {C_i}^{1-hx_i-k(z'_i+z''_i)} = ({G_1}^{\frac{1}{w+y_i}})^{1-hx_i-k(z'_i+z''_i)} = ({G_1}^{1-hx_i-k(z'_i+z''_i)})^{\frac{1}{w+y_i}} = (G_1 \cdot H^{-x_i} \cdot K^{-z'_i} \cdot K^{-z''_i})^{\frac{1}{w+y_i}} = (G_1 \cdot {H_i}^{-1} \cdot K^{-z''_i})^{\frac{1}{w+y_i}}$. That is, $\Pr[\mathtt{Suc}_{10}] = \Pr[\mathtt{Suc}_{11}]$. □

Lemma 8.3.19. *Let* q_{add} *and* q_{iss} *be the number of* $\mathcal{A}$*'s add-user queries and send-to-issuer queries, respectively. Then, there exists a* PPT *algorithm* $\mathcal{B}_3$ *such that* $\big|\Pr[\mathtt{Suc}_{11}] - \Pr[\mathtt{Suc}_{12}]\big| \leq \mathsf{Adv}_{\mathcal{B}_3}^{(q_{\mathsf{add}} + q_{\mathsf{iss}} + 1)\text{-}SDH}(\lambda) + 1/p$ *for any PPT* $\mathcal{A}$.

Proof. We define the event $\mathtt{Bad}_\ell^{(\star)}$ as follows.

$\mathtt{Bad}_\ell^{(\star)}$**:** The event that the adversary $\mathcal{A}$ sends the related query in Type $(\star)$ to the Open oracle in Game ℓ.

Game 11 and Game 12 are identical unless the events $\mathtt{Bad}_{11}^{(\star)}$ and $\mathtt{Bad}_{12}^{(\star)}$ occur. Therefore, we get $\big|\Pr[\mathtt{Suc}_{11}] - \Pr[\mathtt{Suc}_{12}]\big| \leq \Pr[\mathtt{Bad}_{12}^{(\star)}]$ in the same way as in Lemma 8.3.1.

In the following, we construct the algorithm $\mathcal{B}_3$ who tries to solve the simplified $(q_{\mathsf{add}} + q_{\mathsf{iss}})$-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ by using $\mathcal{A}$ and estimate the probability $\Pr[\mathtt{Bad}_{12}^{(\star)}]$. First, $\mathcal{B}_3$ receives a tuple $(G_1, G_2, Y, \{C_i, y_i\}_{i=1}^{q_{\mathsf{add}}+q_{\mathsf{iss}}})$ as the input of the simplified $(q_{\mathsf{add}} + q_{\mathsf{iss}})$-SDH problem. Let $Y = {G_2}^w$, it holds that $C_i = {G_1}^{\frac{1}{w+y_i}}$. Next, $\mathcal{B}_3$ generates the instance of the weak anonymity game. Here for $G_1 \in \mathbb{G}_1, G_2 \in \mathbb{G}_2$ and $Y \in \mathbb{G}_2$, he uses the ones in the tuple of the simplified $(q_{\mathsf{add}} + q_{\mathsf{iss}})$-SDH problem. Also, $\mathcal{B}_3$ samples $h, k \in \mathbb{Z}_p$ uniform randomly and sets $H = {G_1}^h, K = {G_1}^k$. Other elements are generated by following the $\mathsf{GS.GGen}$ algorithm. Let $\mathsf{gpk} = (G_1, G_2, G, \mathsf{H}, H, K, Y, U, V)$, $\mathsf{ik} = w$, and $\mathsf{ok} = (u, v)$, $\mathcal{B}_3$ sends gpk to the adversary $\mathcal{A}$. We note that $\mathcal{B}_3$ does not know the discrete logarithm w of the value Y. The CrptU oracle and the RReg oracle are easily simulated since they just

set the user public key and retrieve the register, respectively. $\mathcal{B}_3$ simulates other oracles (AddU, USK, and SndToI) as follows.

[The AddU oracle] For an input i, the algorithm $\mathcal{B}_3$ runs the UKg algorithm and obtains $(\mathsf{upk}_i, \mathsf{usk}_i) = ((Q_i, H_i), (x_i, z_i'))$. Also, $\mathcal{B}_3$ samples a random value z_i'' and sets $A_i \leftarrow {C_i}^{1-hx_i-k(z_i'+z_i'')}$. Then, he sets $\mathsf{gsk}_i \leftarrow (A_i, y_i, z_i, x_i, Q_i)$. Finally, the user public key upk_i is returned to the adversary, and i is added to HU.

[The USK oracle] For an input i, $\mathcal{B}_3$ returns $\perp$ if $i \notin \mathsf{HU}$. If $i \in \mathsf{HU}$, he returns the secret keys usk_i and gsk_i. Since i is queried to the AddU oracle if $i \in \mathsf{HU}$, $\mathcal{B}_3$ knows the secret keys of the user i.

[The SndToI oracle] For a query $(i, (R_1, R_2))$, $\mathcal{B}_3$ replies as follows:

Step 1 (Practice Phase). Execute other N games with the same adversary $\mathcal{A}$ in parallel with the game of the original $\mathcal{A}$. For $1 \leq j \leq N$, perform the following operations.

1. Sample $\widehat{c}_i^{(j)} \xleftarrow{r} \mathbb{Z}_p$.
2. Execute $\mathcal{A}(\mathsf{gpk}; \rho)$ where the challenger makes the same replies for the original $\mathcal{A}$ until the ℓ-th SndToI query is generated.
3. Send $\widehat{c}_i^{(j)}$ to $\mathcal{A}$ as the first reply of the ℓ-th SndToI query $(i, (R_1, R_2))$, and obtain $(\widehat{\sigma}_{x_i}^{(j)}, \widehat{\sigma}_{z_i'}^{(j)})$. We note that the query $(i, (R_1, R_2))$ is the same as that in the game of the original $\mathcal{A}$ since the challenger makes the same replies until the ℓ-th SndToI query is generated.

Step 2 (First Reply in the Original Game with $\mathcal{A}$). If $i \notin \mathsf{CU}$, return $\perp$. If $i \in \mathsf{CU}$, sample $c_i \xleftarrow{r} \mathbb{Z}_p$ and return c_i as the first reply of the ℓ-th SndToI query $(i, (R_1, R_2))$. Then, obtain $(\sigma_{x_i}, \sigma_{z_i'})$ from $\mathcal{A}$.

Step 3 (Second Reply in the Original Game with $\mathcal{A}$). If $(\sigma_{x_i}, \sigma_{z_i'})$ is invalid, return $\perp$ as the second reply. If $(\sigma_{x_i}, \sigma_{z_i'})$ is valid, find the index $j^* \in [1, N]$ for the replies from $\mathcal{A}$ in Step 1 such that $(\widehat{\sigma}_{x_i}^{(j^*)}, \widehat{\sigma}_{z_i'}^{(j^*)})$ is valid and $c_i \neq \widehat{c}_i^{(j^*)}$. If there is no such index j^*, return $\perp$ as the second reply. On the other hand if there is such index j^*, compute the second reply as follows.

1. Compute $\Delta\sigma_{x_i} \leftarrow \widehat{\sigma}_{x_i}^{(j^*)} - \sigma_{x_i}$, $\Delta\sigma_{z_i'} \leftarrow \widehat{\sigma}_{z_i'}^{(j^*)} - \sigma_{z_i'}$, and $\Delta c_i \leftarrow \widehat{c}_i^{(j^*)} - c_i$, and set $\widetilde{x}_i \leftarrow \Delta\sigma_{x_i}/\Delta c_i$ and $\widetilde{z}_i' \leftarrow \Delta\sigma_{z_i'}/\Delta c_i$.
2. Sample $z_i'' \in \mathbb{Z}_p$ uniform randomly.
3. Compute $A_i \leftarrow {C_i}^{1-h\widetilde{x}_i-k(\widetilde{z}_i'+z_i'')}$, and set $\mathsf{cert}_i \leftarrow (A_i, y_i, z_i'')$ where (C_i, y_i) is the part of the input of the simplified $(q_{\mathsf{add}} + q_{\mathsf{iss}})$-SDH problem. Then, reply cert_i as the second reply, and register $\mathsf{reg}[i] \leftarrow Q_i$.

Finally, $\mathcal{A}$ terminates with $\widetilde{b} \in \{0,1\}$.

When $\mathcal{A}$ generated a related query in Type $(\star)$ to the Open oracle, there is the Open query $(m^*, \Sigma = (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y, \sigma_\delta, \sigma_q, \sigma_r^*))$ such that $\sigma_y = \sigma_y^*$, $\sigma_\delta \neq \sigma_\delta^*$, $\sigma_q \neq \sigma_q^*$, and $\mathsf{GS.Verify}(\mathsf{gpk}, m^*, \Sigma) = 1$. For this query, $\mathcal{B}_3$ computes $\widetilde{w} = (\sigma_\delta - \sigma_\delta^*)/(\sigma_q - \sigma_q^*)$. Moreover, he chooses a value $y \notin \{y_1, \ldots . y_q\}$ and computes $C = G_2{}^{\frac{1}{\widetilde{w}+y}}$. Then, $\mathcal{B}_3$ finally outputs (C, y).

In the following, we show (C, y) is the solution of the simplified $(q_{\mathsf{add}} + q_{\mathsf{iss}})$-SDH problem, that is, $e(C, Y \cdot G_2{}^y) = e(G_1, G_2)$ holds. Since a related query satisfies Equation (8.2), it holds that

$$\begin{aligned}
&e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} \cdot e(T_1^*, G_2)^{\sigma_y} = e(K, G_2)^{\sigma_\delta^*} \cdot e(K, Y)^{-\sigma_q^*} \cdot e(T_1^*, G_2)^{\sigma_y^*} \\
&\quad \Leftrightarrow e(G_1{}^k, G_2)^{\sigma_\delta} \cdot e(G_1{}^k, G_2{}^w)^{-\sigma_q} = e(G_1{}^k, G_2)^{\sigma_\delta^*} \cdot e(G_1{}^k, G_2{}^w)^{-\sigma_q^*} \\
&\quad \Leftrightarrow e(G_1, G_2)^{k\sigma_\delta - kw\sigma_q + t\sigma_y} = e(G_1, G_2)^{k\sigma_\delta^* - kw\sigma_q^* + t\sigma_y^*} \\
&\quad \Leftrightarrow k(\sigma_\delta - w\sigma_q) = k(\sigma_\delta^* - w\sigma_q^*) \ \ (\because \sigma_y = \sigma_y^*).
\end{aligned}$$

From this, if $k \neq 0$, we get $\sigma_\delta - w\sigma_q = \sigma_\delta^* - w\sigma_q^* \Leftrightarrow w = (\sigma_\delta - \sigma_\delta^*)/(\sigma_q - \sigma_q^*)$. Therefore, $C = G_1{}^{\frac{1}{\widetilde{w}+y}} = G_1{}^{\frac{1}{w+y}}$, and (C, y) is the solution of the simplified $(q_{\mathsf{add}}+q_{\mathsf{iss}})$-SDH problem. On the other hand, the probability that $k = 0$ holds is $1/p$ since $k \in \mathbb{Z}_p$ is chosen uniform randomly. Let $\mathtt{BAD}$ be the event that $k = 0$ holds in Game 12. Then, it holds that

$$\begin{aligned}
\Pr[\mathtt{Bad}_{12}^{(\star)}] &= \Pr[\mathtt{Bad}_{12}^{(\star)} \wedge \neg\mathtt{BAD}] + \Pr[\mathtt{Bad}_{12}^{(\star)} \wedge \mathtt{BAD}] \\
&\leq \Pr[\mathtt{Bad}_{12}^{(\star)} \wedge \neg\mathtt{BAD}] + \Pr[\mathtt{BAD}] \\
&\leq \mathsf{Adv}_{\mathcal{B}_3}^{sim\text{-}(q_{\mathsf{add}} + q_{\mathsf{iss}})\text{-}SDH}(\lambda) + 1/p.
\end{aligned}$$

Moreover, since $\mathsf{Adv}_{\mathcal{B}_3}^{sim\text{-}(q_{\mathsf{add}} + q_{\mathsf{iss}})\text{-}SDH}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}_3}^{(q_{\mathsf{add}} + q_{\mathsf{iss}} + 1)\text{-}SDH}(\lambda)$ holds by Theorem 2.1.1, we get $\left|\Pr[\mathtt{Suc}_{11}] - \Pr[\mathtt{Suc}_{12}]\right| \leq \Pr[\mathtt{Bad}_{12}^{(\star)}] \leq \mathsf{Adv}_{\mathcal{B}_3}^{(q_{\mathsf{add}} + q_{\mathsf{iss}} + 1)\text{-}SDH}(\lambda) + 1/p$. □

Also, the following lemmas hold. Since we can show these lemmas as in the case of Theorem 8.3.1, we omit the proofs of the lemmas.

Lemma 8.3.20. *It holds that* $\left|\Pr[\mathtt{Suc}_{12}] - \Pr[\mathtt{Suc}_{13}]\right| \leq 1/p$ *for any PPT* $\mathcal{A}$.

Lemma 8.3.21. *There exists a* PPT *algorithm* $\mathcal{B}_4$ *such that* $\left|\Pr[\mathtt{Suc}_{13}] - \Pr[\mathtt{Suc}_{14}]\right| \leq \mathsf{Adv}_{\mathcal{B}_5}^{DDH}(\lambda)$ *for any PPT* $\mathcal{A}$.

For random values $q^*, r^*, r_1^*, r_2^* \in \mathbb{Z}_p$, the challenge signature in Game 14 is denoted by $\Sigma^* = (T_1^*, T_2^*, T_3^*, T_4^*, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*) = (A_{i_b} \cdot K^{q^*}, Q_{i_b} \cdot G^{r^*}, U^{r_1^*}, V^{r_2^*}, c^*, \sigma_x^*, \sigma_y^*, \sigma_\delta^*, \sigma_q^*, \sigma_r^*)$. Therefore, the choice of the challenge bit b and the distribution of the challenge signature Σ^* are independent. Thus, we can say that $\Pr[\mathtt{Suc}_{14}] = 1/2$. From this fact

and Lemma 8.3.8 to Lemma 8.3.21, we get

$$\begin{aligned}\mathsf{Adv}^{w\text{-}anon}_{\Pi_{\mathsf{FI}},\mathcal{A}}(\lambda) = \left|\Pr[\mathtt{Suc}_0] - 1/2\right| &\leq \sum_{\ell=0}^{13}\left|\Pr[\mathtt{Suc}_\ell] - \Pr[\mathtt{Suc}_{\ell+1}]\right| + \left|\Pr[\mathtt{Suc}_{14}] - 1/2\right| \\ &\leq \mathsf{Adv}^{DDH}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{DDH}_{\mathcal{B}_4}(\lambda) + \mathsf{Adv}^{DL}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{(q_{\mathsf{add}}+q_{\mathsf{iss}}+1)\text{-}SDH}_{\mathcal{B}_3}(\lambda) \\ &\quad + \frac{q_H(1+q_{\mathsf{open}})+5}{p} + \sum_{\ell=1}^{q_{\mathsf{iss}}} \min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}.\end{aligned}$$

Now, we prove that the last term is negligible when setting N to an appropriate value. Let $N = \lambda^{\mathsf{cnst}+1}$ for an arbitrary constant cnst. If $\mathsf{prob}_\ell \geq 1/\lambda^{\mathsf{cnst}}$, we get $(1-\mathsf{prob}_\ell)^N \leq (1-1/\lambda^{\mathsf{cnst}})^{\lambda^{\mathsf{cnst}+1}} = (1-1/\lambda^{\mathsf{cnst}})^{\lambda^{\mathsf{cnst}}\cdot\lambda} < 1/e^\lambda$. Therefore for the sufficiently large λ, it holds that $\min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\} \leq (1-\mathsf{prob}_\ell)^N < 1/\lambda^{\mathsf{cnst}}$. Thus, when $\mathsf{prob}_\ell \geq 1/\lambda^{\mathsf{cnst}}$ holds, $\min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$ is negligible in λ. On the other hand if $\mathsf{prob}_\ell < 1/\lambda^{\mathsf{cnst}}$, it holds that $\min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\} \leq \mathsf{prob}_\ell < 1/\lambda^{\mathsf{cnst}}$. Therefore, also when $\mathsf{prob}_\ell < 1/\lambda^{\mathsf{cnst}}$ holds, $\min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$ is negligible. Since q_{iss} is polynomial in λ, $\sum_{\ell=1}^{q_{\mathsf{iss}}} \min\{(1-\mathsf{prob}_\ell)^N, \mathsf{prob}_\ell\}$ is negligible in λ.

Also, since q_H and q_{open} are polynomial in λ and p is exponential in λ, we get $(q_H(1+q_{\mathsf{open}})+5)/p = \mathsf{negl}(\lambda)$. Therefore, Mechanism 6 satisfies weak anonymity under the DDH assumption, the DL assumption, and the q-SDH assumption in the random oracle model. □

8.4 A Patched Scheme

From our analysis of the security of Mechanism 6, we also derive a simple patch for the scheme. Our patched scheme can be a candidate for the new standardized scheme when ISO/IEC 20008-2 is revised in the future.

In the patched scheme, only the signing and verification algorithms are changed, and the signature size increases by only one element in the group $\mathbb{G}_1$ where $\mathbb{G}_1$ is a source group in the used asymmetric bilinear group. More precisely, a signature in the patched scheme consists of two elements from $\mathbb{G}_1$, three elements from $\mathbb{G}$, and six elements from $\mathbb{Z}_p$ (where $\mathbb{G}$ is the group in which the decisional Diffie-Hellman assumption holds). Also, we need to introduce the external Diffie-Hellman assumption in $\mathbb{G}_1$ to prove the anonymity of the patched scheme, but the other security requirements can be shown under the same assumptions as those of Mechanism 6.

8.4.1 Description

In this section, we give a description of a patched scheme. As we explained before, the flaw of Mechanism 6 is that the underlying proof system does not satisfy simulation soundness. More precisely, for commitments $\{R_i\}_{i\in[1,4]}$ and a challenge value c, the elements σ_x and σ_r are uniquely determined but the other elements σ_y, σ_δ, and σ_q are redundant. By this redundancy, the adversary can re-randomize the challenge signature, and then Mechanism 6 can be broken.

To achieve that Mechanism 6 satisfies anonymity in the BSZ model, we need to remove this redundancy. A simple way to do this is to make the underlying proof system have unique responses [52, 104] (defined as "strict soundness" in the later paper). That is, for commitments $\{R_i\}_i$ and a challenge value c, there exists only one valid proof. By doing so, the adversary cannot re-randomize a signature since there is no candidate of such a signature. However, when we employ the proof system with unique responses, the resulting group signature scheme becomes to be inefficient. This is because many equations need to be proved/verified in such a proof system, and then the signature size and the signing/verifying costs in the group signature scheme also increase.

In the proposed patched scheme, we *reduce* the redundancy to prevent from re-randomizing the signature. Concretely, we add an equation to prove about the witness q and also fix the element σ_q. That is, the parts σ_y and σ_δ are still redundant also in the patched scheme. However, from the analysis of related queries in Section 8.3.2, we see that it is hard to generate related queries in such a situation. When the element σ_q is fixed, possible cases of related queries are " $\widetilde{\sigma}_y \neq \sigma_y^* \ \wedge \ \widetilde{\sigma}_\delta \neq \sigma_\delta^*$ " or " $\widetilde{\sigma}_y \neq \sigma_y^* \ \wedge \ \widetilde{\sigma}_\delta = \sigma_\delta^*$ " or " $\widetilde{\sigma}_y = \sigma_y^* \ \wedge \ \widetilde{\sigma}_\delta \neq \sigma_\delta^*$ ". In Table 8.1, the former two cases are in Type (a), and the later case is in Type (b). As we proved, the probability that the adversary generates the related queries in Type (a) and (b) is negligible. Therefore, the adversary cannot re-randomize a signature when the element σ_q is fixed.

The description of the patched scheme is given in Figure 8.3. Changed parts from Mechanism 6 are underlined. In the patched scheme, only the signing and the verification algorithms are changed whereas the other algorithms (GS.GGen, GS.UGen, GS.Join/GS.Issue, GS.Open, and GS.Judge) are the same as those of Mechanism 6. Concretely, to fix the value σ_q, the element $T_0 = {G_1}^q$ is added as a part of a signature, and a signer also proves this equation when generating a signature. That is, the signature size increases by only one element in $\mathbb{G}_1$ from Mechanism 6.

One concern is that the signer's information may leak by adding a new element to a signature. In Mechanism 6, the randomness $q \in \mathbb{Z}_p$ is used to mask the certificate A_i such that $T_1 = A_i \cdot K^q$. Thus, the tuple (T_0, T_1) is an ElGamal encryption of a certificate A_i. Since Type II pairing is considered, the XDH assumption holds. Thus, the ElGamal

scheme is secure in $\mathbb{G}_1$, and then the additional element T_0 does not leak the signer's information.

$\mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m)$:
$r, q \xleftarrow{r} \mathbb{Z}_p$; $\underline{T_0 \leftarrow {G_1}^q}$; $T_1 \leftarrow A_i \cdot K^q$; $T_2 \leftarrow G^{x_i+r}$; $T_3 \leftarrow U^r$; $T_4 \leftarrow V^r$
$\rho_{x_i}, \rho_{y_i}, \rho_\delta, \rho_q, \rho_r \xleftarrow{r} \mathbb{Z}_p$
$R_1 \leftarrow e(H, G_2)^{\rho_{x_i}} \cdot e(K, G_2)^{\rho_\delta} \cdot e(K, Y)^{-\rho_q} \cdot e(T_1, G_2)^{\rho_{y_i}}$
$R_2 \leftarrow G^{\rho_{x_i}+\rho_r}$; $R_3 \leftarrow U^{\rho_r}$; $R_4 \leftarrow V^{\rho_r}$; $\underline{R_5 \leftarrow {G_1}^{\rho_q}}$
$c \leftarrow \mathsf{H}(\mathsf{gpk}, \{T_i\}_{i\in[0,4]}, \{R_i\}_{i\in[1,5]}, m)$; $\delta \leftarrow z_i - q y_i$
$\sigma_{x_i} \leftarrow x_i \cdot c + \rho_{x_i}$; $\sigma_{y_i} \leftarrow y_i \cdot c + \rho_{y_i}$; $\sigma_\delta \leftarrow \delta \cdot c + \rho_\delta$
$\sigma_q \leftarrow q \cdot c + \rho_q$; $\sigma_r \leftarrow r \cdot c + \rho_r$
Return $\Sigma = (\{T_i\}_{i\in[0,4]}, c, \sigma_{x_i}, \sigma_{y_i}, \sigma_\delta, \sigma_q, \sigma_r)$

$\mathsf{GS.Verify}(\mathsf{gpk}, m, \Sigma)$:
$R_1' \leftarrow e(H, G_2)^{\sigma_x} \cdot e(K, G_2)^{\sigma_\delta} \cdot e(K, Y)^{-\sigma_q} \cdot e(T_1, G_2)^{\sigma_y} \cdot \left(\frac{e(G_1,G_2)}{e(T_1,Y)}\right)^{-c}$
$R_2' \leftarrow G^{\sigma_x+\sigma_r} \cdot {T_2}^{-c}$; $R_3' \leftarrow U^{\sigma_r} \cdot {T_3}^{-c}$; $R_4' \leftarrow V^{\sigma_r} \cdot {T_4}^{-c}$; $\underline{R_5' \leftarrow {G_1}^{\sigma_q} \cdot {T_0}^{-c}}$
Return 1 if $c = \mathsf{H}(\mathsf{gpk}, \{T_i\}_{i\in[0,4]}, \{R_i'\}_{i\in[1,5]}, m)$, else return 0

Figure 8.3: The GS.Sign and the GS.Verify Algorithm of the Patched Scheme

8.4.2 Security

By the above modification, the patched scheme satisfies anonymity in the BSZ model. We give only its intuition here.

A signature in the patched scheme consists of two ElGamal encryptions (specifically, one is a double encryption) and a zero-knowledge proof. Intuitively, the signer's information is hidden from the adversary by the security of the encryption schemes and the zero-knowledge property of the underlying proof system. Therefore, we can easily see that the patched scheme satisfies anonymity if the adversary does not generate a related query as in Theorem 8.3.1. Also, there are three cases of a related query in the patched scheme as we mentioned above, but all the cases be eliminated by the analysis in Section 8.3.2. Thus, the patched scheme satisfies anonymity. Formally, the following theorem holds.

Theorem 8.4.1. *The patched scheme satisfies anonymity under the DL assumption in the group $\mathbb{G}_1$, the XDH assumption in the group $\mathbb{G}_1$, and the DDH assumption in the group $\mathbb{G}$ in the random oracle model.*

Moreover, the patched scheme satisfies the other security requirements, that is, traceability and non-frameability [17]. Since our modification does not affect these security proofs, we can prove the traceability and non-frameability of the patched scheme in the same way as those of the original scheme. Formally, the following theorems hold.

Theorem 8.4.2. *The patched scheme satisfies traceability under the q-SDH assumption in the groups* $(\mathbb{G}_1, \mathbb{G}_2)$ *in the random oracle model.*

Theorem 8.4.3. *The patched scheme satisfies non-frameability under the DL assumption in the group* $\mathbb{G}_1$ *in the random oracle model.*

Chapter 9

Conclusion

This thesis have focused on the cryptographic primitive, group signature, and have shown four contributions in the field of this area.

In Chapter 5, we have discussedthe minimum assumptions for the existence of group signature. Specifically, we showed a construction of a selfless anonymous group signature scheme without any public key encryption scheme. Since we see that a fully anonymous group signature scheme implies a public key encryption scheme from the previous work, we found out that the minimal assumptions are depends on whether the target group signature is fully anonymous or selfless anonymous.

In Chapter 6, we have focused on group signature with verifier-local revocation (VLR-GS), that is one of methodologies to achieve revocation functionality in group signature. Especially, we have showen the first construction of a fully anonymous VLR-GS scheme. Concretely, the scheme is constructed from a digital signature scheme, a key-private public key encryption scheme, and a non-interactive zero-knowledge proof system. Also, we have given VLR-GS schemes with backward unlinkability, which ensures that even after a user is revoked, signatures produced by the user before the revocation remain anonymous.

In Chapter 7, we have introduced the notion of deniable group signature where, with respect to a signature and a user, the authority can issue a proof showing that the specified user is not the signer of the signature, without revealing the actual signer. Also we have shown a concrete construction of deniable group signature.

In Chapter 8, we have given a cryptanalysis of Mechanism 6 that is listed in the ISO/IEC 20008-2 standard. Specifically, we have shown that Mechanism 6 does not reach the expect security revel, but we also showed that Mechanism 6 is still secure under the condition that the issuer does not join the attack. Such a condition is reasonable if a single authority plays roles of both the opener and the issuer. Furthermore, we have provided a simple patched scheme which achieves the expected security level.

As future works, we leave efficient instantiations of some group signature schemes. Concretely, we will construct a fully anonymous VLR-GS scheme based on number-theoretic assumptions even we have provided generic constructions of it. Also, we leave the construction of an efficient deniable group signature scheme, especially in the random oracle model, as one of future work. Furthermore, it is an interesting question whether a NIZK proof system is an inevitable assumption to construct group signature schemes.

References

[1] ISO/IEC 9796-2:2010 information technology – security techniques – digital signature schemes giving message recovery – part 2: Integer factorization based mechanisms.

[2] ISO/IEC 20008-2:2013 information technology – security techniques – anonymous digital signatures – part 2: Mechanisms using a group public key.

[3] ICAO NTWG, PKI for Machine Readable Travel Documents Offering ICC Read-Only Access, Technical Report, Version 1.1, 2004.

[4] EMV, Integrated Circuit Card Specifications for Payment Systems, Book 2, Security and Key Management, Version 4.2, 2008.

[5] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In *TCC*, pages 480–497, 2010.

[6] Michel Abdalla and Bogdan Warinschi. On the minimal assumptions of group signature schemes. In *ICICS*, pages 1–13, 2004.

[7] Masayuki Abe, Sherman S. M. Chow, Kristiyan Haralambiev, and Miyako Ohkubo. Double-trapdoor anonymous tags for traceable signatures. In *ACNS*, pages 183–200, 2011.

[8] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In *CRYPTO*, pages 209–236, 2010.

[9] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO*, pages 255–270, 2000.

[10] Giuseppe Ateniese, Dawn Xiaodong Song, and Gene Tsudik. Quasi-efficient revocation in group signatures. In *FC*, pages 183–197, 2002.

[11] Giuseppe Ateniese and Gene Tsudik. Some open issues and new directions in group signatures. In *FC*, pages 196–211, 1999.

[12] Nuttapong Attrapadung, Keita Emura, Goichiro Hanaoka, and Yusuke Sakai. A revocable group signature scheme from identity-based revocation techniques: Achieving constant-size revocation list. In *ACNS*, pages 419–437, 2014.

[13] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In *ASIACRYPT*, pages 566–582, 2001.

[14] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT*, pages 614–629, 2003.

[15] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS*, pages 62–73, 1993.

[16] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures - how to sign with RSA and rabin. In *EUROCRYPT*, pages 399–416, 1996.

[17] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA*, pages 136–153, 2005.

[18] Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In *EUROCRYPT*, pages 274–285, 1993.

[19] David Bernhard, Georg Fuchsbauer, Essam Ghadafi, Nigel P. Smart, and Bogdan Warinschi. Anonymous attestation with user-controlled linkability. *Int. J. Inf. Sec.*, 12(3):219–249, 2013.

[20] Olivier Blazy, Céline Chevalier, and Damien Vergnaud. Non-interactive zero-knowledge proofs of non-membership. In *CT-RSA*, pages 145–164, 2015.

[21] Olivier Blazy, David Derler, Daniel Slamanig, and Raphael Spreitzer. Non-interactive plaintext (in-)equality proofs and group signatures with verifiable controllable linkability. In *CT-RSA*, pages 127–143, 2016.

[22] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *STOC*, pages 103–112, 1988.

[23] Alexandra Boldyreva, Marc Fischlin, Adriana Palacio, and Bogdan Warinschi. A closer look at PKI: security and efficiency. In *PKC*, pages 458–475, 2007.

[24] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *EUROCRYPT*, pages 56–73, 2004.

[25] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the sdh assumption in bilinear groups. *J. Cryptology*, 21(2):149–177, 2008.

[26] Dan Boneh, Xavier Boyen, and Shai Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. In *CT-RSA*, pages 226–243, 2006.

[27] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *CRYPTO*, pages 41–55, 2004.

[28] Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In *CCS*, pages 168–177, 2004.

[29] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In *EUROCRYPT*, pages 427–444, 2006.

[30] Xavier Boyen and Brent Waters. Full-domain subgroup hiding and constant-size group signatures. In *PKC*, pages 1–15, 2007.

[31] Emmanuel Bresson and Jacques Stern. Efficient revocation in group signatures. In *PKC*, pages 190–206, 2001.

[32] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *CCS*, pages 132–145, 2004.

[33] Jan Camenisch. Efficient and generalized group signatures. In *EUROCRYPT*, pages 465–479, 1997.

[34] Jan Camenisch and Jens Groth. Group signatures: Better efficiency and new theoretical aspects. In *SCN*, pages 120–133, 2004.

[35] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In *Public Key Cryptography*, pages 481–500, 2009.

[36] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO*, pages 61–76, 2002.

[37] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups (extended abstract). In *CRYPTO*, pages 410–424, 1997.

[38] Sébastien Canard, Berry Schoenmakers, Martijn Stam, and Jacques Traoré. List signature schemes. *Discrete Applied Mathematics*, 154(2):189–201, 2006.

[39] David Chaum and Eugène van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.

[40] Sherman S. M. Chow. Real traceable signatures. In *SAC*, pages 92–107, 2009.

[41] Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. Civitas: Toward a secure voting system. In *S&P*, pages 354–368, 2008.

[42] Jean-Sébastien Coron, David Naccache, and Julien P. Stern. On the security of RSA padding. In *CRYPTO*, pages 1–18, 1999.

[43] Jean-Sébastien Coron, David Naccache, Mehdi Tibouchi, and Ralf-Philipp Weinmann. Practical cryptanalysis of ISO/IEC 9796-2 and EMV signatures. In *CRYPTO*, pages 428–444, 2009.

[44] Ivan Damgård, Dennis Hofheinz, Eike Kiltz, and Rune Thorbek. Public-key encryption with non-interactive opening. In *CT-RSA*, pages 239–255, 2008.

[45] Ivan Damgård and Nikos Triandopoulos. Supporting non-membership proofs with bilinear-map accumulators. *IACR Cryptology ePrint Archive*, 2008:538, 2008.

[46] Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In *VIETCRYPT*, pages 193–210, 2006.

[47] Nicolas Desmoulins, Roch Lescuyer, Olivier Sanders, and Jacques Traoré. Direct anonymous attestations with dependent basename opening. In *CANS*, pages 206–221, 2014.

[48] Keita Emura, Goichiro Hanaoka, and Yusuke Sakai. Group signature implies PKE with non-interactive opening and threshold PKE. In *IWSEC*, pages 181–198, 2010.

[49] Keita Emura, Goichiro Hanaoka, Yusuke Sakai, and Jacob C. N. Schuldt. Group signature implies public-key encryption with non-interactive opening. *Int. J. Inf. Sec.*, 13(1):51–62, 2014.

[50] Keita Emura, Takuya Hayashi, and Ai Ishida. Group signatures with time-bound keys revisited: A new model and an efficient construction. In *ASIACCS*, pages 777–788, 2017.

[51] Keita Emura, Takuya Hayashi, and Ai Ishida. Group signatures with time-bound keys revisited: A new model, an efficient construction, and its implementation. *IEEE Transactions on Dependable and Secure Computing*, 2017. To appear.

[52] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the fiat-shamir transform. In *INDOCRYPT*, pages 60–79, 2012.

[53] Uriel Feige and Adi Shamir. Zero knowledge proofs of knowledge in two rounds. In *CRYPTO*, pages 526–544, 1989.

[54] Jun Furukawa and Hideki Imai. An efficient group signature scheme from bilinear maps. In *ACISP*, pages 455–467, 2005.

[55] Jun Furukawa and Hideki Imai. An efficient group signature scheme from bilinear maps. *IEICE Transactions*, 89-A(5):1328–1338, 2006.

[56] Steven D. Galbraith, Florian Hess, and Frederik Vercauteren. Aspects of pairing inversion. *IEEE Transactions on Information Theory*, 54(12):5719–5728, 2008.

[57] David Galindo. Breaking and repairing damgård et al. public key encryption scheme with non-interactive opening. In *CT-RSA*, pages 389–398, 2009.

[58] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In *CRYPTO*, pages 276–288, 1984.

[59] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

[60] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT*, pages 444–459, 2006.

[61] Jens Groth. Fully anonymous group signatures without random oracles. In *ASIACRYPT*, pages 164–180, 2007.

[62] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In *EUROCRYPT*, pages 339–358, 2006.

[63] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, pages 415–432, 2008.

[64] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[65] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust authenticated-encryption AEZ and the problem that it solves. In *EUROCRYPT*, pages 15–44, 2015.

[66] Jung Yeon Hwang, Sokjoon Lee, Byung-Ho Chung, Hyun Sook Cho, and DaeHun Nyang. Group signatures with controllable linkability for dynamic membership. *Inf. Sci.*, 222:761–778, 2013.

[67] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *STOC*, pages 44–61, 1989.

[68] Ai Ishida, Keita Emura, Goichiro Hanaoka, Yusuke Sakai, and Keisuke Tanaka. Group signature with deniability: How to disavow a signature. In *CANS*, pages 228–244, 2016.

[69] Ai Ishida, Keita Emura, Goichiro Hanaoka, Yusuke Sakai, and Keisuke Tanaka. Group signature with deniability: How to disavow a signature. *IEICE Transactions*, 100-A(9):1825–1837, 2017.

[70] Ai Ishida, Yusuke Sakai, Keita Emura, Goichiro Hanaoka, and Keisuke Tanaka. Breaking the group signature scheme in ISO/IEC 20008-2. Submitted to a conference.

[71] Ai Ishida, Yusuke Sakai, Keita Emura, Goichiro Hanaoka, and Keisuke Tanaka. Fully anonymous group signature with verifier-local revocation. Submitted to a conference.

[72] Toshiyuki Isshiki, Kengo Mori, Kazue Sako, Isamu Teranishi, and Shoko Yonezawa. Using group signatures for identity management and its implementation. In *DIM*, pages 73–78, 2006.

[73] Markus Jakobsson and Ari Juels. Mix and match: Secure function evaluation via ciphertexts. In *ASIACRYPT*, pages 162–177, 2000.

[74] Aggelos Kiayias and Moti Yung. Group signatures: Provable security, efficient constructions and anonymity from trapdoor-holders. *IACR Cryptology ePrint Archive*, 2004:76, 2004.

[75] Aggelos Kiayias and Moti Yung. Secure scalable group signature with dynamic joins and separable authorities. *IJSN*, 1(1/2):24–45, 2006.

[76] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC*, pages 581–600, 2006.

[77] Yuichi Komano, Kazuo Ohta, Atsushi Shimbo, and Shinichi Kawamura. Toward the fair anonymous signatures: Deniable ring signatures. In *CT-RSA*, pages 174–191, 2006.

[78] Adeline Langlois, San Ling, Khoa Nguyen, and Huaxiong Wang. Lattice-based group signature scheme with verifier-local revocation. In *PKC*, pages 345–361, 2014.

[79] Jiangtao Li, Ninghui Li, and Rui Xue. Universal accumulators with efficient non-membership proofs. In *ACNS*, pages 253–269, 2007.

[80] Benoît Libert, Thomas Peters, and Moti Yung. Group signatures with almost-for-free revocation. In *CRYPTO*, pages 571–589, 2012.

[81] Benoît Libert, Thomas Peters, and Moti Yung. Scalable group signatures with revocation. In *EUROCRYPT*, pages 609–627, 2012.

[82] Benoît Libert, Thomas Peters, and Moti Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In *CRYPTO*, pages 296–316, 2015.

[83] Benoît Libert and Damien Vergnaud. Group signatures with verifier-local revocation and backward unlinkability in the standard model. In *CANS*, pages 498–517, 2009.

[84] Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.*, 17(2):373–386, 1988.

[85] Yuh-Dauh Lyuu and Ming-Luen Wu. Convertible group undeniable signatures. In *ICISC*, pages 48–61, 2002.

[86] Toru Nakanishi, Hiroki Fujii, Yuta Hira, and Nobuo Funabiki. Revocable group signature schemes with constant costs for signing and verifying. In *PKC*, pages 463–480, 2009.

[87] Toru Nakanishi and Nobuo Funabiki. Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps. In *ASIACRYPT*, pages 533–548, 2005.

[88] Toru Nakanishi and Nobuo Funabiki. A short verifier-local revocation group signature scheme with backward unlinkability. In *IWSEC*, pages 17–32, 2006.

[89] Toru Nakanishi and Nobuo Funabiki. Revocable group signatures with compact revocation list using accumulators. In *ICISC*, pages 435–451, 2013.

[90] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In *CRYPTO*, pages 41–62, 2001.

[91] Moni Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[92] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43, 1989.

[93] Lan Nguyen. Accumulators from bilinear pairings and applications. In *CT-RSA*, pages 275–292, 2005.

[94] Lan Nguyen and Reihaneh Safavi-Naini. Efficient and provably secure trapdoor-free group signature schemes from bilinear pairings. In *ASIACRYPT*, pages 372–386, 2004.

[95] Go Ohtake, Arisa Fujii, Goichiro Hanaoka, and Kazuto Ogawa. On the theoretical gap between group signatures with and without unlinkability. In *AFRICACRYPT*, pages 149–166, 2009.

[96] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.

[97] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *ASIACRYPT*, pages 552–565, 2001.

[98] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.

[99] Yusuke Sakai, Keita Emura, Goichiro Hanaoka, Yutaka Kawai, Takahiro Matsuda, and Kazumasa Omote. Group signatures with message-dependent opening. In *Pairing*, pages 270–294, 2012.

[100] Yusuke Sakai, Jacob C. N. Schuldt, Keita Emura, Goichiro Hanaoka, and Kazuo Ohta. On the security of dynamic group signatures: Preventing signature hijacking. In *PKC*, pages 715–732, 2012.

[101] Jacob C. N. Schuldt and Kanta Matsuura. Efficient convertible undeniable signatures with delegatable verification. *IEICE Transactions*, 94-A(1):71–83, 2011.

[102] Dawn Xiaodong Song. Practical forward secure group signature schemes. In *CCS*, pages 225–234, 2001.

[103] Gene Tsudik and Shouhuai Xu. Accumulating composites and improved group signing. In *ASIACRYPT*, pages 269–286, 2003.

[104] Dominique Unruh. Quantum proofs of knowledge. In *EUROCRYPT*, pages 135–152, 2012.

[105] Shengke Zeng and Shaoquan Jiang. A new framework for conditionally anonymous ring signature. *Comput. J.*, 57(4):567–578, 2014.

[106] Sujing Zhou and Dongdai Lin. Shorter verifier-local revocation group signatures from bilinear maps. In *CANS*, pages 126–143, 2006.