On the Security of Dynamic Group Signatures: Preventing Signature Hijacking*

Yusuke Sakai† Jacob C. N. Schuldt‡ Keita Emura§ Goichiro Hanaoka¶
Kazuo Ohta‖

July 31, 2012

Abstract

We identify a potential weakness in the standard security model for dynamic group signatures which appears to have been overlooked previously. More specifically, we highlight that even if a scheme provably meets the security requirements of the model, a malicious group member can potentially claim ownership of a group signature produced by an honest group member by forging a proof of ownership. This property leads to a number of vulnerabilities in scenarios in which dynamic group signatures are likely to be used. We furthermore show that the dynamic group signature scheme by Groth (ASIACRYPT 2007) does not provide protection against this type of malicious behavior.

To address this, we introduce the notion of *opening soundness* for group signatures which essentially requires that it is infeasible to produce a proof of ownership of a valid group signature for any user except the original signer. We then show a relatively simple modification of the scheme by Groth which allows us to prove opening soundness for the modified scheme without introducing any additional assumptions.

We believe that opening soundness is an important and natural security requirement for group signatures, and hope that future schemes will adopt this type of security.

1 Introduction

Group signatures, introduced by Chaum and van Heyst [12], allow a group member to anonymously sign a message on behalf of the group. More specifically, anyone will be able to verify that a signature originates from a group member, but the signature does not reveal the identity of the signer, not

*An extended abstract appears in The 15th International Conference on Practice and Theory in Public Key Cryptography (PKC 2012).

†The University of Electro-Communications, Japan. The first author is supported by a JSPS Fellowship for Young Scientists. yusuke.sakai@uec.ac.jp

‡National Institute of Advanced Industrial Science and Technology, Japan. The second author is supported by a JSPS Fellowship for Young Scientists. jacob.schuldt@aist.go.jp

§National Institute of Information and Communications Technology, Japan. This work was done when the third author was a postdoctoral researcher at Center for Highly Dependable Embedded Systems Technology, Japan Advanced Institute of Science and Technology. k-emura@nict.go.jp

¶National Institute of Advanced Industrial Science and Technology, Japan. hanaoka-goichiro@aist.go.jp

‖The University of Electro-Communications, Japan. kazuo.ohta@uec.ac.jp

even to other members of the group. Group membership is controlled by an authority called the issuer, who handles enrollment of users through an interactive join protocol. To prevent misuse of the signing capabilities obtained by group members, another authority called the opener can revoke the anonymity of a signature and identify the signer of the message.

Following the introduction of group signatures, a series of different security requirements were proposed for this primitive, each of which aims at addressing a specific security concern by augmenting or refining previous notions, e.g. unforgeability, exculpability, traceability, coalition resistance, framing resistance, anonymity and unlinkability. These security notions were later consolidated in the security model proposed by Bellare, Micciancio, and Warinschi [2] who introduce two strong security requirements, full-anonymity and full-traceability, which imply all of the previously proposed notions of security.

However, a drawback of the model by Bellare, Micciancio, and Warinschi [2] is that only *static* group signature schemes are considered i.e. the set of group members is fixed, and the private key material of each group member is generated in the setup phase of the scheme. Furthermore, the authority controlling the group (which acts as both the issuer and opener) is considered to be fully trusted. To address this, Bellare, Shi, and Zhang [3] extended the model of [2] to capture *dynamic* group signature schemes in which a user can dynamically join the group by engaging in a join protocol with the issuer. Furthermore, to reduce trust in the opener, the model adopts the approach by Camenisch and Michels [11], and requires that the opener produces a non-interactive and publicly verifiable proof that a given signature was produced by a given signer. The model introduces three formal security notions: anonymity, traceability, and non-frameability. The former two notions are adaptations of the full-anonymity and full-traceability notions to the dynamic group signature setting. The latter notion, non-frameability, requires that even if a malicious opener and issuer collude, they cannot frame an honest user by producing a signature and corresponding opening which identify the honest user as the signer, when the honest user did not produce the signature in question.

Limitations of Non-Frameability. While non-frameability is a strong security notion, it only partly covers the security properties one would intuitively expect to gain when the opener is required to produce a non-interactive and publicly verifiable proof of an opening. More specifically, the non-frameability notion only ensures that the opener cannot frame an *uncorrupted* user by constructing a proof that the user is the signer of a signature he did not produce. However, no guarantee is given regarding an opening involving a *corrupted* user. This leaves open the possibility that an opening showing that a malicious or corrupted user is the signer of a signature produced by an honest user, can be constructed. Furthermore, this might not require the opener to be corrupted or malicious, in which case a malicious user might be able to independently forge a proof showing that he is the signer of any signature of his choice.

Depending on the concrete scenario in which a dynamic group signature scheme is used, the ability to forge an opening proof might become a real security concern. We highlight several potential threats that this ability gives rise to:

- ***Signer impersonation.*** The most obvious threat is signer impersonation. This is a problem if a group signature scheme is used for an anonymous auction as suggested in [1]. In this scenario, the bidders correspond to group members, and when submitting a bid, a group member will attach a group signature on his bid. The opener serves as the auctioneer, and

will make the opening of the signature on the highest bid public. This will enable anyone to verify who the winner of the auction is. However, a malicious bidder may forge a proof of ownership of the signature on the highest bid and may insist that he/she is the winner.

A similar situation occurs if a dynamic group signature scheme is used to implement an authentication scheme with identity escrow [25]. In this case, a malicious group member can claim to be the user who authenticated himself to a server (and provide a proof thereof) when this is not the case.

- ***Proxy confession.*** The ability to open a group signature is introduced to keep the group members accountable of the messages signed on behalf of the group. However, assume that a signature on some message causes a dispute, but the real signer wants to avoid being blamed for this. Then the real signer asks (or intimidates) another group member to forge a proof of ownership of the signature and take the blame.

- ***Key exposure.*** Consider the case in which a group member's private key is exposed and falls into the hands of a malicious user. This will not only allow the malicious user to construct future signatures on any message of this choice, but will furthermore allow him to claim (and prove) that the original user is the signer of any *previously generated* signature.

Our Contribution. We highlight the above described potential weakness of the security guarantee provided by the formal model of Bellare, Shi, and Zhang [3]. Furthermore, we show that this is not only a property of the security model, but that the most efficient dynamic group signature schemes enable a malicious group member to forge a proof of ownership of a signature.

To address this, we propose a new security notion for dynamic group signatures which we denote *opening soundness*. We consider two variants of this notion, weak opening soundness and (ordinary) opening soundness. The former is intended to address the above highlighted security threats in an intuitive and straightforward manner, and will rule out the possibility that a malicious group member can produce a proof of ownership of a signature generated by an honest user. The latter considers a stronger adversary who has access to the private key of the opener, and who is only required to produce two different openings of a maliciously constructed signature. The notion of opening soundness implies the notion of weak opening soundness.

As a positive result, we prove that the generic construction of a dynamic group signature scheme by Bellare, Shi, and Zhang [3] achieves opening soundness. We furthermore propose a modification of the scheme by Groth [21] which allows us to prove opening soundness of the modified scheme. In contrast, we show that the original scheme does not provide weak opening soundness. In addition, we briefly discuss opening soundness of the random oracle scheme [16, 4]. A summary of our results regarding opening soundness of the above mentioned schemes can be seen in Table 1.

Related Work. Since the first proposal of group signature by Chaum and van Heyst, many efficient constructions have been proposed, most of which are relying on the random oracle model [1, 7, 10, 24, 16, 13, 4]. Many initial schemes were based on the strong-RSA assumption. The first group signature schemes based on assumptions of the discrete-logarithm type were achieved independently by Camenisch and Lysyanskaya [10], and Boneh, Boyen, and Shacham [7]. The former scheme is based on the LRSW assumption, while the latter is based on the q-strong Diffie-Hellman assumption. Kiayias, Tsiounis, and Yung proposed the notion of traceable signature [23], which

Table 1: Summary of the results. The mark "?" means it is an open question whether the scheme has the given property or not. The rightmost column denotes the section in which the security of the corresponding scheme is discussed.

	Opening Soundness	Weak Opening Soundness	
Our Variant of [21]	Yes	Yes	(§5.1)
Bellare-Shi-Zhang [3]	Yes	Yes	(§4)
Furukawa-Imai [16]	No	?	(§4)
Bichsel et al. [4]	No	?	(§4)
Groth (full version) [21]	No	No	(§4)

can be seen as an extension of group signature with additional anonymity-revocation functionalities. One of these functionalities is that of allowing a group member to claim the authorship of a signature, however, its security requirement does not care about the possibility in which a malicious member falsely claims the authorship of an honestly generated signature by another.

Constructions which are provably secure without random oracles were only recently achieved. Besides the generic construction relying on non-interactive zero-knowledge (NIZK) proofs for general NP languages, Groth constructed the first concrete group signature scheme with constant signature size by exploiting the properties of bilinear groups [19], though signatures are extremely large. Boyen and Waters proposed group signature schemes [8, 9] whose signature sizes are quite compact. In particular the latter scheme has signatures consisting only of six group elements of a composite order group. The drawback of these schemes is that they only achieve weaker security guarantees, that is, they only provide so called CPA-anonymity in the security model of Bellare, Micciancio, and Warinschi [2]. Groth proposed another group signature scheme [20, 21] which has constant signature size (roughly one or two kilobytes) and which is provably secure in the dynamic group signature model of Bellare, Shi, and Zhang [3] without relying on random oracles.

2 Preliminaries

2.1 Group Signatures

In this section, we briefly review the model and the security notions of group signatures, presented by Bellare, Shi, and Zhang [3]. A group signature scheme consists of the following seven algorithms:

GKg: This is the group key generation algorithm which, on input 1^k, returns the keys (gpk, ik, ok), where gpk is a group public key, ik is an issuing key, and ok is an opening key.

UKg: This is the user key generation algorithm which, on input gpk, returns a personal public and private key pair (upk, usk). Each user i will generate a personal key pair (upk_i, usk_i) before engaging in the joining protocol which is described below. It is assumed that anyone can obtain an authentic copy of the public key of any user. (This might be implemented via a standard public key infrastructure.)

Join/Issue: This is the pair of interactive algorithms which implement the joining protocol run by a user i and the issuer. The algorithm Join, which is run by the user, takes (gpk, upk, usk) as

input, whereas Issue, which is run by the issuer, takes (gpk, upk, ik) as input. Upon successful completion of the protocol, Join outputs a private signing key gsk_i for user i, and Issue outputs the registration information of user i which is stored in $reg[i]$, where reg is a registration table maintained by the issuer.

GSig: This is the group signing algorithm run by a user i, which, on input gpk, a signing key gsk_i, and a message m, returns a group signature Σ.

GVf: This is the group signature verification algorithm which, on input (gpk, m, Σ), returns 1 to indicate that Σ is a valid signature on m, or 0 otherwise.

Open: This is the opening algorithm run by the opener, which, on input $(gpk, ok, reg, m, \Sigma)$, returns (i, τ), where i specifies that the originator of the signature Σ is the user i, and τ is a non-interactive proof of this. In case the algorithm fails to identify the originator of the signature, it outputs $i = 0$. Note that Open requires access to the registration table reg.

Judge: This is the judge algorithm which, on input $(gpk, i, upk_i, m, \Sigma, \tau)$, outputs either 1 or 0 indicating that the proof τ is accepted as valid or invalid, respectively.

The model in [3] introduces four requirements for a group signature scheme, namely, correctness, anonymity, non-frameability, and traceability. The correctness notion requires that honestly generated signatures will be accepted as valid by the verification algorithm, can be opened by the opening algorithm, and that the judging algorithm will accept the resulting proof as valid. The anonymity notion requires that no information about the identity of a signer is leaked from a group signature, even if the signing keys of all group members and the issuer are exposed. The non-frameability notion requires that no adversary corrupting both the opener and the issuer, can produce a signature and an opening proof that identify an uncorrupted group member as the signer, when the uncorrupted group member did not produce the signature in question. The traceability notion requires that an adversary corrupting the opener and controlling a group of malicious group members, cannot produce a valid signature that cannot be opened correctly.

The formal definitions of the four notions are given as follows. We first define several oracles needed for security notions:

AddU(i): The add-user oracle runs UKg(gpk) and Join/Issue protocol to add an honest user. It returns the user public key upk of the user. The oracle add i to the set HU.

RReg(i): The read-registration-table oracle reveals the content of the registration table $reg[i]$.

SndToU(i, M): The send-to-user oracle at first sets up a user public/secret key pair by $(upk_i, usk_i) \leftarrow$ UKg(gpk) and add i to the set HU. The oracle then allows the adversary to engage a group-joining protocol of the user i on the behalf of the corrupted issuer. The message M is sent to the user i who follows the protocol Join(gpk, upk_i, usk_i). The response of the user is returned to the adversary.

WReg(i, M): The write-registration-table oracle updates $reg[i]$ to M.

USK(i): The user-secret-keys oracle reveals the secret keys (usk_i, gsk_i) of the user i to the adversary.

CrptU(i, M): The corrupt-user oracle sets the user public key of the user i to M and add i to the set CU.

$\mathsf{Open}(m, \Sigma)$**:** The open oracle returns the opening $(i, \tau) \leftarrow \mathsf{Open}(gpk, ok, m, \Sigma)$ of the signature Σ under the message m.

$\mathsf{Ch}_b(m, i_0, i_1)$**:** The challenge oracle returns a challenge $\Sigma^* \leftarrow \mathsf{GSig}(gpk, gsk_{i_b}, m)$. The users i_0 and i_1 needs to be in the set HU.

$\mathsf{GSig}(i, m)$**:** The signing oracle returns a signature $\Sigma \leftarrow \mathsf{GSig}(gpk, gsk_i, m)$ on the message m of the user i, who needs to be in the set HU.

$\mathsf{SndToI}(i, M)$**:** The send-to-issuer oracle allows the adversary to engage a group-joining protocol on behalf of the corrupted user i. The message M is sent to the issuer who follows the protocol $\mathsf{Issue}(gpk, upk_i, ik)$. The response of the issuer is returned to the adversary. The user i needs to be in the set CU.

The correctness and security requirements for a group signature scheme are as follows:

Definition 1. *A group signature scheme is said to have* correctness *if*

$$\Pr[(gpk, ik, ok) \leftarrow \mathsf{GKg}(1^k); (i, m) \leftarrow \mathcal{A}^{\mathsf{AddU},\mathsf{RReg}}(gpk); \Sigma \leftarrow \mathsf{GSig}(gpk, gsk_i, m); (j, \tau) \leftarrow \mathsf{Open}(gpk, ok, reg, m, \Sigma) : \mathsf{GVf}(gpk, m, \Sigma) = 0 \vee i \neq j \vee \mathsf{Judge}(gpk, i, upk_i, m, \Sigma, \tau) = 0]$$

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

Definition 2. *A group signature scheme is said to have* anonymity *if*

$$\Pr[b \leftarrow \{0,1\}; (gpk, ik, ok) \leftarrow \mathsf{GKg}(1^k); b' \leftarrow \mathcal{A}^{\mathsf{SndToU},\mathsf{WReg},\mathsf{USK},\mathsf{CrptU},\mathsf{Open},\mathsf{Ch}_b}(gpk, ik) : b = b'] - \frac{1}{2}$$

is negligible in k for any probabilistic polynomial-time adversary $\mathcal{A}$.

Definition 3. *A group signature scheme is said to have* non-frameability *if*

$$\Pr[(gpk, ik, ok) \leftarrow \mathsf{GKg}(1^k); (m, \Sigma, i, \tau) \leftarrow \mathcal{A}^{\mathsf{SndToU},\mathsf{WReg},\mathsf{USK},\mathsf{CrptU},\mathsf{GSig}}(gpk, ok, ik) : \mathsf{GVf}(gpk, m, \Sigma) = 1 \wedge i \in HU \wedge \mathsf{Judge}(gpk, i, upk_i, m, \Sigma, \tau) = 1 \wedge \mathcal{A} \text{ queried neither } \mathsf{USK}(i) \text{ nor } \mathsf{GSig}(i, m)]$$

is negligible in k for any probabilistic polynomial-time adversary $\mathcal{A}$.

Definition 4. *A group signature scheme is said to have* traceability *if*

$$\Pr[(gpk, ik, ok) \leftarrow \mathsf{GKg}(1^k); (m, \Sigma) \leftarrow \mathcal{A}^{\mathsf{CrptU},\mathsf{SndToI},\mathsf{AddU},\mathsf{USK},\mathsf{RReg}}(gpk, ok); (i, \tau) \leftarrow \mathsf{Open}(gpk, ok, reg, m, \Sigma) : \mathsf{GVf}(gpk, m, \Sigma) = 1 \wedge (i = 0 \vee \mathsf{Judge}(gpk, i, upk_i, m, \Sigma, \tau) = 0)]$$

is negligible in k for any probabilistic polynomial-time adversary $\mathcal{A}$.

2.2 Other Primitives

Public-Key Encryption. A public key encryption scheme consists of three algorithms (EKg, $\mathsf{Enc}, \mathsf{Dec}$), which satisfy the following correctness condition: For any security parameter $\ell \in \mathbb{N}$, any plaintext $m \in \{0,1\}^*$, any random tape r for EKg, and any random tape s for Enc, the condition $\mathsf{Dec}(dk, \mathsf{Enc}(pk, m; s)) = m$ holds, where pk and dk are output from EKg as $(pk, dk) \leftarrow \mathsf{EKg}(1^\ell; r)$. In this paper we require a public key encryption scheme to satisfy the security notion of indistinguishability under chosen-ciphertext attack (IND-CCA) [27].

Digital Signature. A digital signature scheme consists of three algorithms ($\mathsf{SKg}, \mathsf{Sign}, \mathsf{Ver}$), which satisfy the following correctness condition: For any security parameter $\ell \in \mathbb{N}$, any message $m \in \{0,1\}^*$, any random tape r for SKg, and any random tape s for Sign, the condition $\mathsf{Ver}(vk, m, \mathsf{Sign}(sk, m; s)) = \top$ holds, where vk and sk are output from SKg as $(vk, sk) \leftarrow \mathsf{SKg}(1^\ell; r)$. In this paper we use two types of security for digital signature schemes. One is the standard security notion of unforgeability under adaptive chosen message attack (EUF-CMA), and the other is strong one-time signatures. See [18] for exact definitions.

Target Collision-Resistant Hash Functions. A family of functions is called target collision-resistant if no algorithms, which firstly chooses an input and then is given a description of a function in the family, can find another input that produces the same output to the first input. The formal definition we need is as follows: A function generator $\mathrm{HashGen}(1^\ell)$ takes as input a security parameter and outputs a function $\mathcal{H}$. The family of functions is said to be target collision-resistant when $\Pr[(x, s) \leftarrow \mathcal{A}; \mathcal{H} \leftarrow \mathrm{HashGen}(1^\ell); x' \leftarrow \mathcal{A}(\mathcal{H}, s) : \mathcal{H}(x) = \mathcal{H}(x') \wedge x \neq x']$ is negligible for any polynomial-time algorithm $\mathcal{A}$.

Non-interactive Proofs. A non-interactive proof system for an NP-relation $R \in \{0,1\}^* \times \{0,1\}^*$ defining $L = \{x | (x, w) \in R \text{ for some } w\}$ consists of three algorithms (K, P, V), which satisfy the following correctness and soundness conditions: For correctness, it is required that for any security parameter $\ell \in \mathbb{N}$, any common reference string $crs \leftarrow K(1^\ell)$, and any pair $(x, w) \in R$, it holds that $V(1^\ell, crs, x, P(1^\ell, crs, x, w)) = \top$; for soundness, it is required that for any $\ell \in \mathbb{N}$ and any probabilistic polynomial-time algorithm $\mathcal{A}$, the probability $\Pr[crs \leftarrow K(1^\ell); (x, \pi) \leftarrow \mathcal{A}(1^\ell, crs) : V(1^\ell, crs, x, \pi) = \top \wedge x \notin L]$ is negligible. We will later use three types of proof systems, one which is witness indistinguishable [15], one which is zero-knowledge [5, 14] and one which is simulation-sound zero-knowledge [28].

Bilinear Maps and Groth-Sahai Proofs. Bilinear groups are groups $\mathbb{G}$ and $\mathbb{G}_T$ with the same order that have an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $\mathcal{G}$ be a probabilistic polynomial-time algorithm that outputs a group parameter $gk = (p, \mathbb{G}, \mathbb{G}_T, e, g)$ where p is the order of $\mathbb{G}$ and $\mathbb{G}_T$, e is a non-degenerates bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and g is a generator of $\mathbb{G}$.

Groth and Sahai [22] introduced a framework for very efficient non-interactive proof for the satisfiability of relations in bilinear groups, including pairing product equations. The proof system consists of algorithms $(K_{\mathrm{NI}}, P, V, X)$. The algorithm $K_{\mathrm{NI}}(gk)$ takes a group parameter gk as input and outputs (crs, xk), where crs is a common reference string and xk is a trapdoor extraction key for extracting a witness from a proof. The algorithm $P(crs, x, w)$ outputs a proof π for an equation

described by x whose witness is w. A proof π is verified by running $V(crs, x, \pi)$. The algorithm $X_{xk}(x, \pi)$ extracts a witness from the proof π which passes the verification algorithm.

There are two types of Groth-Sahai proof systems, $(K_{\mathrm{NI}}, P_{\mathrm{NIWI}}, V_{\mathrm{NIWI}}, X)$ and $(K_{\mathrm{NI}}, P_{\mathrm{NIZK}}, V_{\mathrm{NIZK}}, X)$, which respectively provide witness-indistinguishability and zero-knowledge. The two types of proof systems have identical common reference string generation algorithms, and can share a single common reference string. Furthermore, there are two types of reference strings: one yields perfect soundness, and the other yields perfect witness indistinguishability or perfect zero-knowledge, depending on the type of proof system. For further details see [22].

Tag-based Encryption. Tag-based encryption is an extension of public key encryption, which associates an additional "tag" with a ciphertext. The exact syntax is as follows: A key generation algorithm $G(1^\ell)$ generates a public key pk and a secret key dk; an encryption algorithm $E_{pk}(t, m)$ takes as input a public key pk, a tag t, and a plaintext m, and outputs a ciphertext c; a decryption algorithm $D_{dk}(t, c)$ takes as input a decryption key dk, a tag t, and a ciphertext c and outputs a plaintext m or a special symbol $\perp$ indicating the decryption failed. The correctness condition only ensures that the plaintext is recovered when the tags used in the encryption and the decryption are identical.

In this paper we use Kiltz's construction of tag-based encryption [26], which is explained below. The scheme can be built on bilinear groups. Let $gk = (p, \mathbb{G}, \mathbb{G}_T, e, g)$ be a group description. The key generation algorithm chooses random integers $\phi, \eta \leftarrow \mathbb{Z}_p$ and random elements $K, L \leftarrow \mathbb{G}$, and sets $pk = (F, H, K, L)$ where $F = g^\phi$ and $H = g^\eta$ and $dk = (\phi, \eta)$. A ciphertext of a plaintext m under a tag t is computed as $y = (y_1, y_2, y_3, y_4, y_5) = (F^r, H^s, mg^{r+s}, (g^tK)^r, (g^tL)^s)$. The decryption algorithm decrypts a ciphertext $(y_1, y_2, y_3, y_4, y_5)$ under a tag t by checking $e(F, y_4) = e(y_1, g^tK)$ and $e(H, y_5) = e(y_2, g^tL)$ and outputs $y_3/y_1^\phi y_2^\eta$ if the two equations hold, otherwise outputs $\perp$. This encryption scheme is secure against selective-tag weak chosen-ciphertext attacks if the decisional linear assumption holds [26]. Another interesting property is that the scheme has public verifiability in the sense that it can be efficiently checked whether a given five-tuple $(y_1, y_2, y_3, y_4, y_5)$ lies in the range of the encryption algorithm under a given public key pk and a given tag t by checking the two equations $e(F, y_4) = e(y_1, g^tK)$ and $e(H, y_5) = e(y_2, g^tL)$.

3 Opening Soundness

In this section we give a formal definition of opening soundness. Specifically, we introduce two variants of opening soundness, weaker and stronger definitions.

The weaker definition, named weak opening soundness, is intended to address the security concerns discussed in the introduction in a straightforward manner, and will rule out the possibility that a malicious user can claim ownership of a signature produced by an honest user by forging an opening proof. The definition is as follows:

Definition 5. *A group signature scheme is said to have* weak opening soundness *if*

$$\begin{aligned}\Pr[(gpk, ik, ok) \leftarrow \mathsf{GKg}(1^k); (m, i, i^*, s) \leftarrow \mathcal{A}^{\mathsf{AddU}(\cdot)}(gpk);\\ \Sigma \leftarrow \mathsf{GSig}(gpk, gsk_i, m); \tau^* \leftarrow \mathcal{A}^{\mathsf{AddU}(\cdot)}(s, \Sigma, gsk_{i^*})\\ : i \neq i^* \wedge i, i^* \in \mathsf{HU} \wedge \mathsf{Judge}(gpk, i^*, upk_{i^*}, m, \Sigma, \tau^*) = 1]\end{aligned}$$

is negligible for all polynomial time adversaries $\mathcal{A}$, where the oracle AddU *is defined as follows:*

AddU**:** *On a query $i \in \mathbb{N}$, the oracle runs $(upk_i, usk_i) \leftarrow \mathsf{UKg}(gpk)$, then executes the protocol $(gsk_i, reg_i) \leftarrow \langle \mathsf{Join}(gpk, upk_i, usk_i), \mathsf{Issue}(gpk, ik) \rangle$, adds i to a set $\mathcal{HU}$, and lastly returns upk_i.*

Note that the adversary is only allowed to receive the secret signing key of a single user i^*. Hence, this definition will not rule out attacks involving a corrupted opener, and therefore cannot contribute towards reducing trust in this entity.

In contrast, the stronger definition, named opening soundness, is intended to rule out the possibility that an adversary can produce two different openings of a signature, even if he is allowed to corrupt the opener and all the users in the system, and furthermore generate the signature in question maliciously. The definition is as follows:

Definition 6. *A group signature scheme is said to have* opening soundness *if*

$$\Pr[(gpk, ik, ok) \leftarrow \mathsf{GKg}(1^k); (m, \Sigma, i_1, \tau_1, i_2, \tau_2) \leftarrow \mathcal{A}^{\mathsf{CrptU},\mathsf{WReg}}(gpk, ok, ik) : \mathsf{GVf}(gpk, m, \Sigma) = 1 \wedge i_1 \neq i_2 \wedge \mathsf{Judge}(gpk, i_1, upk_{i_1}, m, \Sigma, \tau_1) = 1 \wedge \mathsf{Judge}(gpk, i_2, upk_{i_2}, m, \Sigma, \tau_2) = 1]$$

is negligible for all polynomial time adversaries $\mathcal{A}$, where the oracle $\mathsf{CrptU}(i, M)$ sets the user public key of the user i to be M, and the oracle $\mathsf{WReg}(i, M)$ sets $reg[i]$ to M.

While the weaker definition provides a minimum level of protection against the type of attacks described in the introduction, we believe that, when applied to the scenarios mentioned in the introduction, any dynamic group signature scheme should provide (ordinary) opening soundness to prevent any type of attack which exploits ambiguity of openings, or involves a corrupted opener. Furthermore, we will show that this level of security can be achieved efficiently by showing that our modified version of the scheme by Groth provides opening soundness (See Sect. 5 for details).

4 Opening Soundness of Existing Schemes

We will now take a closer look at some of the existing dynamic group signature schemes, and highlight the level of opening soundness (ordinary, weak or none) achieved by these. Note that since the Bellare-Shi-Zhang security model for dynamic group signatures does not considers opening soundness, a security proof in this model will not allow us to make any conclusions regarding the opening soundness of existing schemes.

In this section, we will focus on the standard model scheme by Groth described in [21] (note that the updated scheme in [21] is slightly different from the scheme described in [20]) and the generic construction of a dynamic group signature scheme by Bellare, Shi, and Zhang [3]. More specifically, we will show that the scheme by Groth does not have weak opening soundness whereas the generic construction by Bellare, Shi and Zhang has opening soundness. We further show that the random oracle model schemes by Furukawa and Imai [16] and Bichsel et al. [4] do not have opening soundness. Interestingly, while these schemes do not provide opening soundness, there seems to be no obvious attack against the weak opening soundness of these.

$\mathsf{GKg}(1^k)$:
$gk \leftarrow \mathcal{G}(1^k)$; $\mathcal{H} \leftarrow \mathrm{HashGen}(1^k)$
$(f, h, z) \leftarrow \mathbb{G}$; $T = e(f, z)$
$(crs, xk) \leftarrow K_{\mathrm{NI}}(gk)$
$(F, H, U, V, W, U', V', W') \leftarrow crs$
$K, L \leftarrow G$; $pk \leftarrow (F, H, K, L)$
(gpk, ik, ok)
$\leftarrow ((gk, \mathcal{H}, f, h, T, crs, pk), z, xk)$

$\mathsf{Join}/\mathsf{Issue}$(User i: gpk; Issuer: gpk, ik):
Run the coin-flipping protocol in [21]
The user obtains $v_i = g^{x_i}$ and x_i
and the issuer obtains v_i
Issuer: $r \leftarrow \mathbb{Z}_p$
$(a_i, b_i) \leftarrow (f^{-r}, (v_i h)^r z)$
set $reg[i] \leftarrow v_i$
send (a_i, b_i) to the user
User: If $e(a_i, hv_i)e(f, b_i) = T$,
set $gsk_i \leftarrow (x_i, a_i, b_i)$

$\mathsf{Open}(gpk, ok, reg, m, \Sigma)$:
(b, v, σ)
$\leftarrow X_{xk}(crs, (gpk, a, \mathcal{H}(vk_{\mathrm{sots}})), \pi)$
Return (i, σ) if there is i so $v = reg[i]$,
else return $(0, \sigma)$

$\mathsf{GSig}(gpk, gsk_i, m)$:
$(vk_{\mathrm{sots}}, sk_{\mathrm{sots}}) \leftarrow \mathrm{KeyGen}_{\mathrm{sots}}(1^k)$
(Repeat until $\mathcal{H}(vk_{\mathrm{sots}}) \neq -x_i$)
$\rho \leftarrow \mathbb{Z}_p$; $a \leftarrow a_i f^{-\rho}$; $b \leftarrow b_i (hv_i)^{\rho}$
$\sigma \leftarrow g^{1/(x_i + \mathcal{H}(vk_{\mathrm{sots}}))}$
$\pi \leftarrow P_{\mathrm{NIWI}}(crs, (gpk, a, \mathcal{H}(vk_{\mathrm{sots}})), (b, v_i, \sigma))$
$y \leftarrow E_{pk}(\mathcal{H}(vk_{\mathrm{sots}}), \sigma)$
$\psi \leftarrow P_{\mathrm{NIZK}}(crs, (gpk, y, \pi), (r, s, t))$
$\sigma_{\mathrm{sots}} \leftarrow \mathrm{Sign}_{sk_{\mathrm{sots}}}(vk_{\mathrm{sots}}, m, a, \pi, y, \psi)$
Return $\Sigma = (vk_{\mathrm{sots}}, a, \pi, y, \psi, \sigma_{\mathrm{sots}})$

$\mathsf{GVf}(gpk, m, \Sigma)$:
Return 1 if the following holds:
$1 = \mathrm{Ver}_{vk_{\mathrm{sots}}}((vk_{\mathrm{sots}}, m, a, \pi, y, \psi), \sigma_{\mathrm{sots}})$,
$1 = V_{\mathrm{NIWI}}(crs, (gpk, a, \mathcal{H}(vk_{\mathrm{sots}})), \pi)$,
$1 = V_{\mathrm{NIZK}}(crs, (gpk, \pi, y), \psi)$, and
$1 = \mathrm{ValidCiphertext}(pk, \mathcal{H}(vk_{\mathrm{sots}}), y)$,
else return 0

$\mathsf{Judge}(gpk, i, reg[i], m, \Sigma, \sigma)$:
Return 1 if
$i \neq 0 \wedge e(\sigma, v_i g^{\mathcal{H}(vk_{\mathrm{sots}})}) = e(g, g)$,
else return 0

Figure 1: The Groth group signature scheme [21].

The Groth Scheme. Figure 1 shows a description of the Groth scheme. Below, we will expand on the description given in the figure. However, before discussing the implementation details of the Groth scheme, we note that the scheme diverge slightly from the description of a dynamic group signature scheme given in the Bellare-Shi-Zhang model [3]. Specifically, in [3], a user is assumed to independently generate a public/private key pair (upk_i, usk_i), and then afterwards obtain a group signing key gsk_i by interacting with the issuer in the Join protocol. In the Groth scheme [21], on the other hand, a user generates a public/private key pair *jointly* with the issuer in the Join protocol. The public key for user i will be stored in $reg[i]$ by the issuer, and the corresponding private key will be the group signing key gsk_i of user i. This intuitively corresponds to a scheme in which the user key generation algorithm UKg is merged with the Join protocol. Note that in this setup it is assumed that user i and the issuer agree upon the content of $reg[i]$. To ensure this, it is suggested in [21] that the user signs the content of $reg[i]$, using a separate signing key, and that an entry in reg is only considered to be valid if the content is signed by the corresponding user.

To model the security of this type of scheme, a few minor changes are required to the security model presented in Sect. 2.1. Specifically, the public key upk_i of user i is simply defined as the

content of $reg[i]$, and we no longer consider the write-registration-table oracle WReg in the security definitions, but only the corrupt oracle CrptU which allows the adversary to set the public key upk_i (i.e., the content of $reg[i]$) to a given value, i.e., the WReg oracle is simply removed from the security definitions. Furthermore, since the issuer is the only party which can insert the public key of a user in reg, and the issuer will only do so upon successful completion of the Join protocol, we no longer consider the corrupt oracle CrptU in the traceability security definition, but only SndToI which allows the adversary to interact with the (honest) issuer in the Join protocol, i.e., the CrptU oracle is removed from the definition. Lastly, the oracles AddU and SendToU will no longer run the algorithm UKg since this algorithm is not defined for the scheme. With these changes, we obtain a security model equivalent to the model presented by Groth [21].

We will now return to the implementation details of the Groth scheme. In the group key generation algorithm GKg, the elements f, h, T correspond to a verification key of the Zhou-Lin signature scheme [29], whereas z corresponds to the signing key. Furthermore, pk is a public key of Kiltz's tag-based encryption scheme. Note that the first two elements of pk and the common reference string crs for the non-interactive Groth-Sahai proofs are identical.

In the group signing algorithm GSig, a group member constructs two non-interactive Groth-Sahai proofs. The first proof π, constructed via P_{NIWI}, shows knowledge of a signature σ, a verification key v and a part b of a (re-randomized) certificate (a, b) which satisfy $e(a, hv)e(f, b) = T \wedge e(\sigma, vg^{\mathcal{H}(vk_{\text{sots}})}) = e(g, g)$. The first part a of the certificate can be safely revealed as part of the group signature since it does not leak any information about the identity of the member due to the re-randomization. The second proof ψ, constructed via P_{NIZK}, demonstrates that the plaintext of y is same as the witness σ used in π. More specifically, the tag-based encryption y has the form $(y_1, y_2, y_3, y_4, y_5) = (F^{r_y}, H^{s_y}, g^{r_y+s_y}\sigma, (g^{\mathcal{H}(vk_{\text{sots}})}K)^{r_y}, (g^{\mathcal{H}(vk_{\text{sots}})}L)^{s_y})$, while the Groth-Sahai proof π contains a commitment $c = (c_1, c_2, c_3) = (F^{r_c}U^t, H^{s_c}V^t, g^{r_c+s_c}W^t\sigma)$. The proof demonstrates that there exists (r, s, t) such that $(c_1y_1^{-1}, c_2y_2^{-1}, c_3y_3^{-1}) = (F^rU^t, H^sV^t, g^{r+s}W^t)$. When y and c encrypt the same message, there exists (r, s, t) that satisfies above equation, but if y and c encrypt different messages, no such tuple (r, s, t) exists.

The verification algorithm GVf will, in addition to the verification of the two non-interactive proofs and the one-time signature, verify that the ciphertext y is a valid ciphertext, using the algorithm ValidCiphertext. This algorithm is easily implemented for the tag-based encryption scheme by Kiltz (see the last paragraph of Sect. 2 for details).

We will now show how a malicious group member can forge a opening proof which shows that he is the signer of any signature Σ produced by user i. As described above, an opening proof consists of a certified signature σ on vk_{sots} which is part of Σ. To verify the opening proof, it is only verified that σ is a valid signature on vk_{sots} under the verification key v_i of the user in question.

Hence, a malicious user i' who wants to impersonate the signer of the group signature Σ on m, simply uses his own private signing key $x_{i'}$ to construct a new signature σ' on vk_{sots}, and publicizes this as an opening proof together with his own identity i'. This proof will be accepted by the Judge algorithm since σ' is a valid signature in vk_{sots}.

We formally state this as a theorem:

Theorem 1. *The Groth group signature scheme does not provide weak opening soundness.*

Proof. We describe an algorithm for producing a forged proof: When the adversary receives the security parameter 1^ℓ and a group public key gpk, it firstly issues two queries AddU(1) and AddU(2) in order to add two members 1 and 2 the group. The adversary then requests the challenge by out-

putting $(i, i^*, m) = (1, 2, 0^\ell)$, and receives a tuple (Σ, gsk_2), where $\Sigma = (vk_{\text{sots}}, a, \pi, y, \psi, \sigma_{\text{sots}})$ and $gsk_2 = (x_2, a_2, b_2)$. The adversary forges a proof of ownership by computing $\sigma^* = g^{1/(x_2+\mathcal{H}(vk_{\text{sots}}))}$ and outputs σ^* (Notice that vk_{sots} is taken from the group signature Σ).

One can easily verify that $\mathsf{Judge}(gpk, reg, 2, m, \Sigma, \sigma^*)$ actually outputs 1, which means that the algorithm successfully breaks the opening soundness. □

The Bellare-Shi-Zhang Scheme. Below, we will give an intuitive description of the generic construction of a dynamic group signature scheme by Bellare, Shi, and Zhang [3]. For a full description, see Appendix A.

In the Bellare-Shi-Zhang construction, each group member i has a key pair (vk_i, sk_i) of an EUF-CMA secure signature scheme. The issuer also possesses his own key pair (ak, ck) of the signature scheme. The issuer signs the message $\langle i, vk_i \rangle$ to obtain the signature $cert_i$, and sends $cert_i$ to the user i. A group signature on a message m by the user i is a pair (C, π): here C is an encryption of $\langle i, vk_i, cert_i, s \rangle$, s is a signature on m under the key pair (vk_i, sk_i), and the NIZK proof π proves that the plaintext encrypted in C is of the form $\langle i, vk, cert, s \rangle$ and that $cert$ and s are a valid certificate on vk_i and a valid signature on the message in question, respectively. The opener attributes a group signature $\Sigma = (C, \pi)$ to the user i by providing an NIZK proof τ for another statement (i.e., different from that of π), which shows the existence of a decryption key that corresponds to the opener's public key and that under that key C is decrypted to $\langle i, vk_i, cert_i, s \rangle$.

This simple scheme provides opening soundness. Intuitively, this is due to the correctness of the public key encryption used to encrypt the signature and the certificate, and the soundness of the NIZK proof system for τ. The correctness condition of public key encryption ensures that given a public key pk and a ciphertext C, the decryption of C is determined uniquely. Now, let us assume that an adversary of the opening soundness game outputs a tuple $(m, \Sigma, i_1, \tau_1, i_2, \tau_2)$ where $\Sigma = (C, \pi)$ and wins the game. The proof τ_1 proves that C decrypts to $\langle i_1, vk, cert, s \rangle$ for some vk, $cert$, and s, whereas τ_2 proves that C decrypts to a different plaintext $\langle i_2, vk', cert', s' \rangle$ for some vk', $cert'$, and s'. However, this should not be possible since the decryption of C under a fixed public key is unique. Hence, the adversary breaks the soundness of the NIZK proof system.

This is captured by the following theorem.

Theorem 2. *The Bellare-Shi-Zhang construction (Fig. 3) provides opening soundness, assuming that the non-interactive proof systems (P_1, V_1) and (P_2, V_2) provide soundness with negligible soundness error.*

For a formal proof, see Appendix A.

The Furukawa-Imai Scheme. The Furukawa-Imai group signature scheme [16] does not have opening soundness, which we will show in the following.

The scheme makes use of a group $\mathbb{G}$ (with generator g) in which the decisional Diffie-Hellman assumption holds, in addition to bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ with an asymmetric bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. In the scheme, each group member i has a public key $Q_i = g^{x_i}$ and the corresponding secret key x_i. The public key Q_i is encrypted in a group signature with (a kind of) ElGamal encryption. Let $(R, V) = (Q_i g^r, S^r)$ be the ciphertext that appears in a group signature, where $S = g^s$ is the public key of the ElGamal encryption. The opener possesses the decryption key s, and identifies the signer by decrypting the ciphertext. An opening contains a proof of knowledge

of w such that $Q_i = R/V^{1/w}$, where Q_i is the public key of the specified member (The opener uses s as the witness for the above equation).

If the adversary corrupts the opener and two different members i and j, the adversary can construct two different openings of a single signature, each of which attributes the signature to user i and user j, respectively. The adversary proceeds as follows: At first the signature is honestly generated by the user i. Let $(R, V) = (g^{x_i+r}, S^r)$ be the ciphertext contained in this signature. The first opening is also honestly generated by the opener to attribute the signature to i. The second proof is generated by computing a proof of knowledge w that satisfies $Q_j = R/V^{1/w}$ with the witness $w = sr/(x_i + r - x_j)$. This proof attributes the signature to the user j. Note that the randomness r for the encryption is reused to forge the second proof. This is the reason why the adversary needs to corrupt the user i, not only the user j and the opener.

The Bichsel et al. Scheme. In the Bichsel et al. scheme [4], a group member receives a Camenisch-Lysyanskaya signature on a random message ξ from the issuer. To generate a group signature, the member rerandomizes this certificate and computes a "signature of knowledge" of ξ on the message m in question. This rerandomized certificate on ξ and the signature of knowledge of ξ on m constitute the group signature.

The issuer should not know the random message ξ, because otherwise non-frameability is compromised. For this reason, in the group-joining protocol, ξ is jointly generated by the user and the issuer as follows: The user i chooses a random exponent τ_i and sends $\tilde{r} = \tilde{x}^{\tau_i}$ to the issuer, while the issuer also chooses a random κ_i and computes $\tilde{w} = \tilde{r} \cdot \tilde{x}^{\kappa_i} = \tilde{x}^{\tau_i+\kappa_i}$. This $\tau_i + \kappa_i$ will be used as the random message ξ mentioned above. To establish a publicly verifiable connection between this ξ and the user i, the user i generates an (ordinary) signature on $k_i = e(g, \tilde{r})$ with a key pair which is previously registered in a public key infrastructure.

To open a signature, the opener uses $\tilde{w}$ to identify the user which corresponds to the rerandomized certificate in the group signature, which is a Camenisch-Lysyanskaya signature on the user's ξ. However, since $\tilde{w}$ makes the Camenisch-Lysyanskaya signature publicly verifiable, it cannot be used as an opening. Instead, the opener produces a non-interactive zero-knowledge proof of $\tilde{w}$ and κ_i such that $k_i = e(g, \tilde{w})/e(g, \tilde{x})^{\kappa_i}$ and provides the signature on k_i. To verify this opening, a third party simply verifies the non-interactive zero-knowledge proof and the signature.

Unfortunately this scheme does not satisfy opening soundness. Assume a malicious signer obtains a group signature by an honest user, and further obtains an honestly generated opening of the signature. The proof of ownership contains k_i and a signature on this by the honest user. The malicious signer replaces the signature on k_i with his own signature on k_i. This forged opening passes the verification.

5 Achieving Opening Soundness

In this section we present a variant of the Groth scheme, which provides opening soundness (besides anonymity, non-frameability, and traceability).

5.1 The Modified Groth Scheme

The High-Level Idea. Let us first consider a general approach for achieving opening soundness.

The opener, who has the secret opening key, will always be able to determine the correct opening of a group signature. To provide opening soundness, the opener needs to convince others that a given opening is correct. The easiest way to do that is to make the opening key public, but this will compromise the anonymity of the scheme. Instead, the opener can provide an NIZK proof of the correctness of an opening, to convince any third party. This is, in fact, the approach used in the Bellare-Shi-Zhang construction.

If the opening algorithm essentially corresponds to a "decryption" of a ciphertext contained in the group signature (this is the case for many existing schemes), we might be able to take a different and more efficient approach. In particular, if the encryption scheme provides randomness recovering, the opener can simply release the randomness used for the ciphertext in question instead of an expensive zero-knowledge proof. Any third party will then be able to verify the correctness of an opening by re-encrypting the relevant information with the randomness provided by the opener, and then confirm that the resulting ciphertext is the same as the one contained in the signature.

In the Groth scheme, an opening essentially corresponds to the decryption of a linear encryption scheme. While linear encryption is not randomness-recovering, the opener is able to release related values which, together with the use of a bilinear map, will allow a third party to confirm that the decryption was done correctly. This property will allow us to add opening soundness to the original scheme. More specifically, in our variant of the Groth scheme, the opener, given a ciphertext $(c_1, c_2, c_3) = (F^r, H^s, vg^{r+s})$, reveals g^r and g^s as a part of an opening. Using the properties of the bilinear map, these values can replace the exact randomness r and s when checking the correspondence between a ciphertext and a decryption: If a third party, given g^r and g^s, wants to check the correspondence between a ciphertext (c_1, c_2, c_3) and a decryption v, he simply checks whether the equations $e(F, g^r) = e(c_1, g)$, $(H, g^s) = e(c_2, g)$, and $v = c_3/(g^r g^s)$ hold. If this is the case, he accepts the decryption as valid.

The above described modification to the Groth scheme will ensure that a verifier running the Judge algorithm is able to verify that the public user key v_i, given as part of an opening, is the same as the public user key used in the proof π which is contained in the group signature. This will ensure that two different openings containing different public user keys cannot both be accepted as valid for a single group signature. While this property is very close to opening soundness, it will not address the possibility that two different user have the same public key. To rule this out, we make the following additional change to the Groth scheme: we let both the verification algorithm GVf and the judge algorithm Judge take the registration table *reg* as input i.e. we assume that *reg* is made public (note that this is allowed in the original scheme [21]). With this change, the Judge algorithm can simply check whether the public key, given as part of an opening, corresponds to the public key of more than one user, and reject the opening if this is the case. However, to ensure that the scheme remains traceable, the verification algorithm will have to implement a similar check. Hence, we will simply reject any signature or opening in the case the registration table *reg* contains repeated public keys. Note that to preserve correctness, this change also requires us to ensure that no honest execution of the Join protocol generates repeated public keys.

We note that the used approach to the verification of a decryption result is essentially the same as that used by Galindo et al. [17] in the context of public key encryption with non-interactive opening (PKENO). Furthermore, we note that in [17], the application of PKENO schemes to group signature is briefly discussed as a mechanism for simplifying the construction of an opening. Here, we will show that this approach is able to ensure the opening soundness of group signature schemes.

Description of our variant. The Groth scheme can achieve opening soundness with the small modification shown in Fig. 2.

$\mathsf{Join}/\mathsf{Issue}$(User i: gpk; Issuer: gpk, ik):
Run the coin-flipping protocol in [21]
  The user obtains $v_i = g^{x_i}$ and x_i
  and the issuer obtains v_i
  (Repeat until $v_i \neq reg[j]$ for all j)
Issuer: $r \leftarrow \mathbb{Z}_p$
  $(a_i, b_i) \leftarrow (f^{-r}, (v_i h)^r z)$
  set $reg[i] \leftarrow v_i$
  send (a_i, b_i) to the user
User: If $e(a_i, hv_i)e(f, b_i) = T$
  set $gsk_i \leftarrow (x_i, a_i, b_i)$

$\mathsf{GVf}(gpk, reg, m, \Sigma)$:
Return 1 if the following holds:
  $1 = \mathrm{Ver}_{vk_{\mathrm{sots}}}((vk_{\mathrm{sots}}, m, a, \pi, y, \psi), \sigma_{\mathrm{sots}})$,
  $1 = V_{\mathrm{NIWI}}(crs, (gpk, a, \mathcal{H}(vk_{\mathrm{sots}})), \pi)$,
  $1 = V_{\mathrm{NIZK}}(crs, (gpk, \pi, y), \psi)$,
  $1 = \mathrm{ValidCiphertext}(pk, \mathcal{H}(vk_{\mathrm{sots}}), y)$,
  and $reg[i] \neq reg[j]$ for all $i \neq j$
else return 0

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

$\mathsf{Judge}(gpk, i, reg, m, \Sigma, (\sigma, \tau_F, \tau_H))$:
$v_i \leftarrow reg[i]$
Return 1 if the following holds:
  $\mathsf{GVf}(gpk, reg, m, \Sigma) = 1$,
  $i \neq 0$, $e(\sigma, v_i g^{\mathcal{H}(vk_{\mathrm{sots}})}) = e(g, g)$,
  $e(F, \tau_F) = e(y_1, g)$, $e(H, \tau_H) = e(y_2, g)$,
  and $\sigma\tau_F\tau_H = y_3$,
else return 0

Figure 2: The proposed modification of the Groth group signature scheme. The algorithms that do not appear in the figure are exactly the same as in Fig. 1.

Theorem 3. *The modified Groth scheme shown in Fig. 2 provides opening soundness.*

Proof. Let us consider the game in Definition 6, and let gpk be the group public key in the game, where the key is parsed as $(F, H, \cdots)$, and let $(m, \Sigma, i, \tau, i', \tau')$ be the output of the adversary. Furthermore, let Σ, τ, and τ' be parsed as follows: $\Sigma = (vk_{\mathrm{sots}}, a, \pi, y, \psi, \sigma_{\mathrm{sots}})$ in which $y = (y_1, y_2, y_3, y_4, y_5)$, $\tau = (\sigma, \tau_F, \tau_H)$ and $\tau' = (\sigma', \tau'_F, \tau'_H)$.

We hereafter show that given a fixed Σ, it must hold that $i = i'$: Given a fixed Σ (in particular y_1, y_2, and y_3), the verification equations

$$e(F, \tau_F) \stackrel{?}{=} e(y_1, g) \wedge e(H, \tau_H) \stackrel{?}{=} e(y_2, g) \wedge \sigma\tau_F\tau_H \stackrel{?}{=} y_3$$

uniquely determine τ_F, τ_H, and σ. Since both $\tau = (\sigma, \tau_F, \tau_H)$ and $\tau' = (\sigma', \tau'_F, \tau'_H)$ are accepted by Judge and hence must satisfy the verification equations, we must have that $(\sigma, \tau_F, \tau_H) = (\sigma', \tau'_F, \tau'_H)$. Now, since $\sigma = \sigma'$ and the equation $e(\sigma, vg^{\mathcal{H}(vk_{\mathrm{sots}})}) = e(g, g)$ uniquely determines v given fixed σ and $\mathcal{H}(vk_{\mathrm{sots}})$, that v_i and $v_{i'}$ satisfy $e(\sigma, v_i g^{\mathcal{H}(vk_{\mathrm{sots}})}) = e(g, g)$ and $e(\sigma, v_{i'} g^{\mathcal{H}(vk_{\mathrm{sots}})}) = e(g, g)$ respectively, must imply that $v_i = v_{i'}$. Hence, since $v_i = reg[i] \neq reg[j] = v_j$ for all $i \neq j$, we conclude that $i = i'$. □

The changes shown in Fig. 2 yields a scheme which is secure in the Bellare-Shi-Zhang model i.e. the anonymity, the non-frameability, and the traceability of the original Groth scheme are maintained. This will be shown in the following.

Theorem 4. *The modified Groth scheme provides anonymity if the decisional linear assumption holds in $\mathbb{G}$, the one-time signature scheme is strongly unforgeable, and the hash function is target collision-resistant.*

Proof. Let $\mathcal{A}$ be an adversary that have the advantage ε in the anonymity game. To bound the probability ε we gradually modify the game played by $\mathcal{A}$. In the following S_i denotes the event that the adversary $\mathcal{A}$ 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 that $\Pr[S_0] = 1/2 + \varepsilon$.

Game 1. We modify the behavior of the Open oracle as follows: If the Open oracle receives a valid signature which reuses the verification key vk^*_{sots} from the challenge Σ^*, then the game aborts. Due to the strong unforgeability of the one-time signature scheme ($\text{KeyGen}_{\text{sots}}$, $\text{Sign}_{\text{sots}}, \text{Ver}_{\text{sots}}$), this modification does not change the success probability of $\mathcal{A}$ with more than a negligible amount, that is, we have that $|\Pr[S_0] - \Pr[S_1]|$ is negligible.

Game 2. We further modify the Open oracle to abort when a queried signature contains a one-time signature verification key vk_{sots} that, when applying the hash function $\mathcal{H}$, collides with the challenge verification key vk^*_{sots} i.e. $\mathcal{H}(vk_{\text{sots}}) = \mathcal{H}(vk^*_{\text{sots}})$. This causes at most a negligible change in the probability in which $\mathcal{A}$ successfully guess the challenge bit due to the collision resistant property of $\mathcal{H}$.

Game 3. We then modify how the Open oracle obtains a signer identity i: When the Open oracle is required to open a group signature, it first extracts a witness (b, v, σ) from the proof π using the extraction key xk. However, in Game 3, instead of then searching for i such that $reg[i] = v$ (which is done until Game 2), the Open oracle searches for i such that

$$e(\sigma, v_i g^{\mathcal{H}(vk_{\text{sots}})}) = e(g, g)$$

that is, σ is a valid signature on vk_{sots} under v_i. Note that the above verification equation uniquely defines v_i given a signature σ and a message $\mathcal{H}(vk_{\text{sots}})$. Furthermore, since the perfect soundness of π guarantees that σ is a valid signature on $\mathcal{H}(vk_{\text{sots}})$ under the extracted v, the v_i identified in the above procedure must be identical to v, and hence, the user identity i returned by the oracle does not vary between Game 2 and Game 3.

Game 4. We now modify how the Open oracle obtains the signature σ: Specifically, in Game 4, the Open oracle obtains σ by decrypting y with xk, instead of extracting σ from the proof of knowledge π. Due to the perfect soundness of ψ, this modification produces the same σ as in Game 3.

Game 5. Now we change how (σ, τ_F, τ_H) is computed. Instead of decrypting y with xk (recall that xk consists of $\log_g F$ and $\log_g H$), we proceed as follows: In the generation of the public key of

the tag-based encryption, K and L are constructed as $K := F^{\kappa}$ and $L := H^{\lambda}$. The Open oracle then uses κ and λ to compute (σ, τ_F, τ_H) as $\tau_F := (y_4/{y_1}^{\kappa})^{1/\mathcal{H}(vk_{\text{sots}})}$, $\tau_H := (y_5/{y_2}^{\lambda})^{1/\mathcal{H}(vk_{\text{sots}})}$, and $\sigma := y_3/\tau_F\tau_H$. As shown in Lemma 1, this will not change the behavior of the oracle.

Game 6. In this game we switch the common reference string from a string providing perfect soundness to a string providing perfect witness-indistinguishability and perfect zero-knowledge, respectively, for the two types of proof systems used in the scheme. Since to two types of reference strings are computationally indistinguishable under the decisional linear assumption, the success probability of the adversary will not change by more than a negligible amount. Note that this change is possible because the Open oracle no longer needs the extraction key xk. Furthermore, in this game, all proofs ψ are simulated with the zero-knowledge trapdoor.

Game 7. Finally we change the component y_3 in the challenge to a random element in $\mathcal{G}$. As shown in Lemma 2, this will not introduce more than a negligible change in the success probability of the adversary assuming the decisional linear assumption holds.

In Game 7 we can conclude that $\Pr[S_7] = 1/2$, because the view of the adversary is independent from the challenge bit b. Specifically, the challenge $(vk^*_{\text{sots}}, a, \pi, y, \psi, \sigma^*_{\text{sots}})$ contains no information on b. Indeed, vk^*_{sots} is independently generated in the setup, a is distributed uniformly due to rerandomization, the perfectly witness-indistinguishable proof π distributes independently from the witness and hence the bit b, y is merely a random encryption, ψ does not contain the information on b since it is computed from y and the zero-knowledge trapdoor, and finally σ^*_{sots} is a signature on $\langle vk^*_{\text{sots}}, m, a, \pi, y, \psi\rangle$, which are all independent of b as seen above. The oracles (Open, SndToU, WReg, USK and CrptU) also behave independently of b.

Finally we prove that the changes in Game 5 and Game 7 will only introduce a negligibly change in the success probability of the adversary.

Lemma 1. $\Pr[S_4] = \Pr[S_5]$.

Proof (of Lemma 1). We will show that the response of the Open oracle does not change between Game 4 and Game 5.

Consider a group signature $\Sigma = (vk_{\text{sots}}, a, \pi, y, \psi, \sigma_{\text{sots}})$ submitted to the Open oracle. If the ciphertext y, which is a part of Σ, does not pass the validity check ValidCiphertext, the oracles in both games simply outputs $\perp$.

Hence, we consider the case in which the ciphertext y passes the validity check. In this case we can assume that there exist r and s in $\mathbb{Z}_p$ such that $y_1 = F^r$, $y_2 = H^s$, $y_4 = (g^{\mathcal{H}(vk_{\text{sots}})}K)^r$, and $y_5 = (g^{\mathcal{H}(vk_{\text{sots}})}L)^s$. We now show that the three equations $\tau_F = g^r$, $\tau_H = g^r$ and $\sigma = y_3/g^{r+s}$ hold in both games, and hence, the openings (τ_F, τ_H, σ) returned by Open in Game 4 and Game 5 are identical.

Consider the first two equations. In Game 4, τ_F and τ_H are computed as $\tau_F := y_1^{1/d_F}$ and $\tau_H := y_2^{1/d_H}$. Since $F = g^{d_F}$ and $H = g^{d_H}$, $\tau_F = y_1^{1/d_F} = (F^r)^{1/d_F} = g^r$ and $\tau_H = y_2^{1/d_H} = (H^s)^{1/d_H} = g^s$ hold. In Game 5, τ_F and τ_H are computed as $\tau_F := (y_4/{y_1}^{\kappa})^{1/\mathcal{H}(vk_{\text{sots}})}$ and $\tau_H := (y_5/{y_2}^{\lambda})^{1/\mathcal{H}(vk_{\text{sots}})}$, where $K = F^{\kappa}$ and $L = H^{\lambda}$. Thus

$$\tau_F = \left(\frac{y_4}{{y_1}^{\kappa}}\right)^{1/\mathcal{H}(vk_{\text{sots}})} = \left(\frac{(g^{\mathcal{H}(vk_{\text{sots}})}K)^r}{(F^r)^{\kappa}}\right)^{1/\mathcal{H}(vk_{\text{sots}})} = g^r$$

and

$$\tau_H = \left(\frac{y_5}{{y_2}^{\lambda}}\right)^{1/\mathcal{H}(vk_{\mathrm{sots}})} = \left(\frac{(g^{\mathcal{H}(vk_{\mathrm{sots}})}L)^s}{(H^s)^{\lambda}}\right)^{1/\mathcal{H}(vk_{\mathrm{sots}})} = g^s.$$

Lastly, consider the third equation $\sigma = y_3/g^{r+s}$. Note that σ is computed as $\sigma := y_3/{y_1}^{1/d_F}{y_3}^{d_H}$ in Game 4, whereas it is computed as $\sigma := y_3/(y_4/{y_1}^{\kappa})^{1/\mathcal{H}(vk_{\mathrm{sots}})}(y_5/{y_2}^{\lambda})^{1/\mathcal{H}(vk_{\mathrm{sots}})}$ in Game 5. Since we have already established that ${y_1}^{1/d_F} = (y_4/{y_1}^{\kappa})^{1/\mathcal{H}(vk_{\mathrm{sots}})} = g^r$ and ${y_2}^{1/d_H} = (y_5/{y_2}^{\lambda})^{1/\mathcal{H}(vk_{\mathrm{sots}})} = g^s$, we can conclude that the two computations yield the same value y_3/g^{r+s}. □

Lemma 2. $|\Pr[S_6] - \Pr[S_7]|$ *is negligible if the decisional linear assumption holds.*

Proof (of Lemma 2). To see this we construct a simulator that distinguishes a linear tuple from a random tuple, given that $|\Pr[S_6] - \Pr[S_7]|$ is non-negligible for some $\mathcal{A}$. The simulator receives the description of bilinear groups gk and a tuple (F, H, g, F^r, H^s, R) where R is g^{r+s} or a random group element, and simulates either Game 6 or Game 7, respectively.

Given gk and (F, H, g, F^r, H^s, R), the simulator constructs a witness-indistinguishable common reference string on the top of g, F, H together with a zero-knowledge trapdoor, which can be done because the trapdoor consists of only the discrete logarithms of U', V', W' with respect to F, H, and g. Then the simulator sets up K, L as $K := F^{c_1}g^{-\mathcal{H}(vk^*_{\mathrm{sots}})}$, $L := H^{c_2}g^{-\mathcal{H}(vk^*_{\mathrm{sots}})}$ where c_1, c_2 are randomly chosen from $\mathbb{Z}_p$. The rest of the public verification key gpk is honestly generated, and the adversary $\mathcal{A}$ is run with input gpk and ik.

When the adversary $\mathcal{A}$ issues an oracle query, the simulator responds as follows: User joining queries, both corrupted and uncorrupted, is dealt with by simply following the real protocol. The challenge request (i_0, i_1, m) is handled by picking a random bit b, computing a and π correctly from the signing key x_{i_b} of user i_b, computing a ciphertext y as $(y_1, y_2, y_3, y_4, y_5) := (F^r, H^s, R\sigma, (F^r)^{c_1}, (H^s)^{c_2})$, generating a simulated proof ψ from the zero-knowledge trapdoor, and generating a one-time signature σ_{sots} on $(vk^*_{\mathrm{sots}}, m, a, \pi, y, \psi)$. When the simulator receives an open query $(vk_{\mathrm{sots}}, a, \pi, y, \psi, \sigma_{\mathrm{sots}})$, the simulator first verifies the signature, and if the signature does not pass the verification, it returns $\perp$. In the case the signature is valid, the simulator computes

$$\tau_F := (y_1^{c_1}/y_4)^{1/(\mathcal{H}(vk^*_{\mathrm{sots}})-\mathcal{H}(vk_{\mathrm{sots}}))}, \quad \tau_H := (y_2^{c_2}/y_5)^{1/(\mathcal{H}(vk^*_{\mathrm{sots}})-\mathcal{H}(vk_{\mathrm{sots}}))}, \quad \sigma := y_3/\tau_F\tau_H,$$

finds i for which σ is a valid signature on the message vk_{sots} under $v_i = reg[i]$, and outputs $(i, (\sigma, \tau_F, \tau_H))$. If no such i is found, output $(0, \perp)$.

Finally the adversary outputs a bit b' and halts. The simulator outputs 1 if $b = b'$, and outputs 0 if $b \neq b'$.

In the above simulation, if R in the tuple given to the simulator is equal to g^{r+s}, the simulated oracle response is identical to that of Game 6. On the other hand, if R is randomly chosen, the simulation is identical Game 7. Hence if $|\Pr[S_6]-\Pr[S_7]|$ is non-negligible, the simulator's advantage in distinguishing linear tuples is also non-negligible. □

These two lemmas complete the proof of Theorem 4. □

Non-frameability and traceability can be proven more easily since these security notions do not require simulation of the Open oracle. For non-frameability, once an opening of the modified scheme that compromises the non-frameability notion is produced, one can obtain an opening for the original scheme (by simply dropping the extra components of τ_F and τ_H) which will compromise

the non-frameability of the original scheme. The proof of the following theorems are essentially identical to the original proofs given in [21], and are therefore not given here.

Theorem 5. *The modified Groth scheme provides non-frameability assuming the q-SDH assumption [6] holds, the one-time signature scheme is existentially unforgeable under a weak chosen message attack, and that the hash function is collision resistant.*

Theorem 6. *The modified Groth scheme provides traceability assuming the q-U assumption [21] holds.*

6 Conclusion

We have identified an overlooked security concern for dynamic group signatures, namely, the possibility that a false opening proof can be produced by a corrupt user. To address this concern, we defined (two variants of) a new security notion denoted opening soundness, and furthermore discussed the opening soundness of several existing schemes. As a result, we have shown that the Bellare-Shi-Zhang construction [3] provides opening soundness as it is, and that small modifications to the Groth scheme (of the full version) [21] allow this scheme to provide opening soundness as well. We have also briefly discussed the opening soundness of some of the random oracle schemes [16, 4], but leave further investigation of these schemes as future work.

Acknowledgment

The authors would like to thank the anonymous reviewers of PKC 2012 for their helpful comments. Furthermore, the authors would like to thank Benoît Libert for pointing out the similarity between the idea for obtaining opening soundness in the Groth scheme, and techniques of [17].

References

[1] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In M. Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 255–270. Springer, Heidelberg, 2000.

[2] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 644–644. Springer, Heidelberg, 2003.

[3] M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In A. Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, Heidelberg, 2005.

[4] P. Bichsel, J. Camenisch, G. Neven, N. P. Smart, and B. Warinschi. Get shorty via group signatures without encryption. In J. A. Garay and R. De Prisco, editors, *SCN 2007*, volume 6280 of *LNCS*, pages 381–398. Springer, Heidelberg, 2010.

[5] M. Blum, A. De Santis, S. Micali, and G. Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, 1991.

[6] D. Boneh and X. Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *J. Cryptol.*, 21:149–177, 2008.

[7] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 227–242. Springer, Heidelberg, 2004.

[8] X. Boyen and B. Waters. Compact group signatures without random oracles. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 427–444. Springer, Heidelberg, 2006.

[9] X. Boyen and B. Waters. Full-domain subgroup hiding and constant-size group signatures. In T. Okamoto and X. Wang, editors, *PKC 2007*, volume 4450 of *LNCS*, pages 1–15. Springer, Heidelberg, 2006.

[10] J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72. Springer, Heidelberg, 2004.

[11] J. Camenisch and M. Michels. A group signature scheme with improved efficiency (extended abstract). In K. Ohta and D. Pei, editors, *ASIACRYPT '98*, volume 1514 of *LNCS*, pages 160–174. Springer, Heidelberg, 1998.

[12] D. Chaum and E. van Heyst. Group signatures. In D. W. Davies, editor, *EUROCRYPT '91*, volume 547 of *LNCS*, pages 257–265. Springer, Heidelberg, 1991.

[13] C. Delerablée and D. Pointcheval. Dynamic fully anonymous short group signatures. In P. Nguyen, editor, *VIETCRYPT 2006*, volume 4341 of *LNCS*, pages 193–210. Springer, Heidelberg, 2006.

[14] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string. In *31st Annual Symposium on Foundations of Computer Science*, pages 308–317. IEEE Computer Society, 1990.

[15] U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *22nd Annual ACM Symposium on Theory of Computing*, pages 416–426. ACM, 1990.

[16] J. Furukawa and H. Imai. An efficient group signature scheme from bilinear maps. In C. Boyd and J. M. González Nieto, editors, *ACISP 2005*, volume 3574 of *LNCS*, pages 92–128. Springer, Heidelberg, 2005.

[17] D. Galindo, B. Libert, M. Fischlin, G. Fuchsbauer, A. Lehmann, M. Manulis, and D. Schröder. Public-key encryption with non-interactive opening: New constructions and stronger definitions. In D. J. Bernstein and T. Lange, editors, *AFRICACRYPT 2010*, volume 6055 of *LNCS*, pages 333–350. Springer, Heidelberg, 2010.

[18] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[19] J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Heidelberg, 2006.

[20] J. Groth. Fully anonymous group signatures without random oracles. In K. Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 164–180. Springer, Heidelberg, 2007.

[21] J. Groth. Fully anonymous group signatures without random oracles. Manuscript, May 17, 2010. `http://www.cs.ucl.ac.uk/staff/J.Groth/CertiSignFull.pdf`.

[22] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In N. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Heidelberg, 2008.

[23] A. Kiayias, Y. Tsiounis, and M. Yung. Traceable signatures. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 571–589. Springer, Heidelberg, 2004.

[24] A. Kiayias and M. Yung. Group signatures with efficient concurrent join. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 198–214. Springer, Heidelberg, 2005.

[25] J. Kilian and E. Petrank. Identity escrow. In H. Krawczyk, editor, *CRYPTO '98*, volume 1462 of *LNCS*, pages 169–185. Springer, Heidelberg, 1998.

[26] E. Kiltz. Chosen-ciphertext security from tag-based encryption. In S. Halevi and T. Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 581–600. Springer, Heidelberg, 2006.

[27] C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *CRYPTO '91*, volume 576 of *LNCS*, pages 433–444. Springer, Heidelberg, 1992.

[28] A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th Annual Symposium on Foundations of Computer Science*, pages 543–553. IEEE Computer Society, 1999.

[29] S. Zhou and D. Lin. Shorter verifier-local revocation group signatures from bilinear maps. In D. Pointcheval, Y. Mu, and K. Chen, editors, *CANS 2006*, volume 4301 of *LNCS*, pages 126–143. Springer, Heidelberg, 2006.

A Omitted Definitions and Proofs

For completeness, we provide a complete description of the Bellare-Shi-Zhang construction and a formal proof of Theorem 2.

The description of the scheme is presented in Fig. 3. The construction is a generic construction from a EUF-CMA secure signature scheme $(\mathsf{SKg}, \mathsf{Sign}, \mathsf{Ver})$, a IND-CCA secure public-key encryption scheme $(\mathsf{EKg}, \mathsf{Enc}, \mathsf{Dec})$, a simulation-sound zero-knowledge non-interactive proof system (K_1, P_1, V_1), and a zero-knowledge non-interactive proof system (K_2, P_2, V_2). The proof system (K_1, P_1, V_1) is for the relation

$$((pk, ak, m, C), (i, vk', cert, \sigma, r)) \in R_1 \\ \iff \mathsf{Ver}_{ak}(\langle i, vk' \rangle, cert) = 1 \wedge \mathsf{Ver}_{vk'}(m, s) = 1 \wedge \mathsf{Enc}_{pk}(\langle i, vk', cert, s \rangle; r) = C,$$

while the proof system (K_2, P_2, V_2) is for the relation

$$((pk, C, i, vk, cert, \sigma), (dk, R)) \in R_2 \iff \mathsf{EKg}(1^k; R) = (pk, dk) \wedge \mathsf{Dec}(dk, C) = \langle i, vk, cert, \sigma \rangle.$$

$\mathsf{GKg}(1^k)$:
$crs_1 \leftarrow K_1(1^\ell)$; $crs_2 \leftarrow K_2(1^\ell)$
$R \leftarrow \{0,1\}^k$; $(pk, dk) \leftarrow \mathsf{EKg}(1^k; R)$
$(ak, ck) \leftarrow \mathsf{SKg}(1^k)$
$gpk := (1^k, crs_1, crs_2, pk, ak)$
$ok := (dk, R)$; $ik := ck$
Return (gpk, ok, ik)

$\mathsf{UKg}(1^k)$:
$(upk, usk) \leftarrow \mathsf{SKg}(1^k)$
Return (upk, usk)

Join/Issue :
$\mathsf{Join}(gpk, upk_i, usk_i)$:
  $(vk_i, sk_i) \leftarrow \mathsf{SKg}(1^k)$; $s_i \leftarrow \mathsf{Sign}_{usk_i}(vk_i)$
  Send (vk_i, s_i) to the issuer
$\mathsf{Issue}(gpk, upk_i, ik)$:
  If $\mathsf{Ver}_{upk_i}(vk_i, s_i) = 1$ then
    $cert_i \leftarrow \mathsf{Sign}_{ck}(\langle i, vk_i \rangle)$
    $reg[i] := (vk_i, s_i)$,
  Else $cert_i := \varepsilon$
  Send $cert_i$ to the user
User:
  $gsk_i := (i, vk_i, sk_i, cert_i)$

$\mathsf{GSig}(gpk, gsk_i, m)$:
Parse gpk as $(1^k, crs_1, crs_2, pk, ak)$
Parse gsk_i as $(i, vk_i, sk_i, cert_i)$
$\sigma \leftarrow \mathsf{Sign}_{sk_i}(m)$
$r \leftarrow \{0,1\}^k$; $C \leftarrow \mathsf{Enc}_{pk}(\langle i, vk_i, cert_i, \sigma \rangle; r)$
$\pi_1 \leftarrow P_1(1^k, (pk, ak, m, C),$
$(i, vk_i, cert_i, \sigma, r), crs_1)$
Return $\Sigma := (C, \pi)$;

$\mathsf{GVf}(gpk, reg, m, \Sigma)$:
Parse gpk as $(1^k, crs_1, crs_2, pk, ak)$
Parse Σ as (C, π_1)
Return $V_1(1^k, (pk, ak, m, C), \pi_1, crs_1)$

$\mathsf{Open}(gpk, ok, reg, m, \Sigma)$:
Parse gpk as $(1^k, crs_1, crs_2, pk, ak)$
Parse ok as (dk, R)
Parse Σ as (C, π_1)
$M \leftarrow \mathsf{Dec}_{dk}(C)$
Parse M as $\langle i, vk, cert, \sigma \rangle$
If $reg[i] \neq \varepsilon$ then
  Parse $reg[i]$ as (vk_i, s_i)
Else $vk_i := \varepsilon$; $s_i := \varepsilon$
$\pi_2 \leftarrow P_2(1^k, (pk, C, i, vk, cert, \sigma), (dk, R), crs_2)$
If $V_1(1^k, (pk, ak, m, C), \pi_1, crs_1) = 0$ then
  Return $(0, \varepsilon)$
If $vk \neq vk_i$ or $reg[i] = \varepsilon$ then
  Return $(0, \varepsilon)$
$\tau := (vk_i, s_i, i, vk, cert, \sigma, \pi_2)$
Return (i, τ)

$\mathsf{Judge}(gpk, reg, i, upk_i, m, \Sigma, \tau)$:
Parse gpk as $(1^k, crs_1, crs_2, pk, ak)$
Parse Σ as (C, π_1)
If $(i, \tau) = \varepsilon$ then
  Return $V_1(1^k, (pk, ak, m, C), \pi_1, crs_1) = 0$
Parse τ as $(\bar{vk}, \bar{s}, i', vk', cert', \sigma', \pi_2)$
If $V_2(1^k, (C, i', vk', cert', \sigma'), \pi_2, crs_2) = 0$ then
  Return 0
If $i = i'$ and $\mathsf{Ver}_{upki}(\bar{vk}, \bar{s}) = 1$
  and $\bar{pk} = pk'$ then
  Return 1
Else Return 0

Figure 3: The Bellare-Shi-Zhang group signature scheme [3].

The proof of Theorem 2 is as follows:

Proof. Let $(\mathsf{GKg}, \mathsf{UKg}, \mathsf{Join}, \mathsf{Issue}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{Judge})$ be the Bellare-Shi-Zhang construction. Let us consider an adversary $\mathcal{A}$ that is run in the environment of the opening soundness experiment, and let succ be the event that $\mathcal{A}$ breaks the opening soundness of the scheme.

We will show that the probability $\Pr[\mathsf{succ}]$ is negligible. Toward this end we define three

events $\mathsf{invalid}$, $\mathsf{non\text{-}trace}_1$, and $\mathsf{non\text{-}trace}_2$. The event $\mathsf{invalid}$ is that $\mathcal{A}$ outputs $(M, \Sigma, i, \tau, i', \tau')$ such that the group signature $\Sigma = (c, \pi)$ contains a ciphertext c that has no corresponding plaintext m and randomness r which satisfy $c = E_{pk}(m; r)$. The event $\mathsf{non\text{-}trace}_1$ denotes that, for the ciphertext c output by $\mathcal{A}$, there exists no decryption key dk that satisfies $pk = G(1^k, dk)$ and $D_{dk}(c) = \langle i_1, vk, cert, s\rangle$ for some vk, $cert$, and s, and finally $\mathsf{non\text{-}trace}_2$ denotes the same event for i_2.

By the union bound, we obtain an upper bound for $\Pr[\mathsf{succ}]$ as

$$\begin{aligned}\Pr[\mathsf{succ}] \leq \Pr[\mathsf{succ} \wedge \neg\mathsf{invalid} \wedge \neg\mathsf{non\text{-}trace}_1 \wedge \neg\mathsf{non\text{-}trace}_2] \\ + \Pr[\mathsf{succ} \wedge \mathsf{invalid}] + \Pr[\mathsf{succ} \wedge \mathsf{non\text{-}trace}_1] + \Pr[\mathsf{succ} \wedge \mathsf{non\text{-}trace}_2].\end{aligned}$$

The last three terms $\Pr[\mathsf{succ} \wedge \mathsf{invalid}]$, $\Pr[\mathsf{succ} \wedge \mathsf{non\text{-}trace}_1]$, and $\Pr[\mathsf{succ} \wedge \mathsf{non\text{-}trace}_2]$ are all negligible due to the soundness of the underlying zero-knowledge proof systems which are assumed to have negligible soundness error i.e. if the event $\mathsf{invalid}$ occurs, it is straightforward to construct an algorithm which breaks the soundness of (K_1, P_1, V_1), and likewise, if either of the events $\mathsf{non\text{-}trace}_1$, or $\mathsf{non\text{-}trace}_2$ occur, it is straightforward to construct algorithms which break the soundness of (K_2, P_2, V_2).

The remaining part is to show that $\Pr[\mathsf{succ} \wedge \neg\mathsf{invalid} \wedge \neg\mathsf{non\text{-}trace}_1 \wedge \neg\mathsf{non\text{-}trace}_2]$ is negligible. This term is in fact exactly equal to zero, due to the correctness of the public key encryption scheme used. The condition $\neg\mathsf{invalid} \wedge \neg\mathsf{non\text{-}trace}_1 \wedge \neg\mathsf{non\text{-}trace}_2$ means that there are two different decryption keys dk_1 and dk_2 that correspond to the same public key pk (i.e., there are random tapes ρ_1 and ρ_2 such that $(pk, dk_1) = \mathsf{EKg}(1^k; \rho_1)$ and $(pk, dk_2) = \mathsf{EKg}(1^k; \rho_2)$) but which produce different decryption results for a single valid ciphertext c. The correctness condition requires that if a ciphertext c is honestly generated under a public key pk, two decryption keys which are different but correspond to the same public key pk, produce the same decryption results. Since the above situation contradicts this requirement, the probability $\Pr[\mathsf{succ} \wedge \neg\mathsf{invalid} \wedge \neg\mathsf{non\text{-}trace}_1 \wedge \neg\mathsf{non\text{-}trace}_2]$ is equal to zero. □