Traceable Group Encryption

Benoît Libert[1], Moti Yung[2], Marc Joye[1], and Thomas Peters[3] ⋆

[1] Technicolor (France)
[2] Google Inc. and Columbia University (USA)
[3] Université catholique de Louvain, Crypto Group (Belgium)

Abstract. Group encryption (GE) is the encryption analogue of group signatures. It allows a sender to verifiably encrypt a message for some certified but anonymous member of a group. The sender is further able to convince a verifier that the ciphertext is a well-formed encryption under some group member's public key. As in group signatures, an opening authority is empowered with the capability of identifying the receiver if the need arises. One application of such a scheme is secure repository at an unknown but authorized cloud server, where the archive is made accessible by a judge order in the case of misbehavior, like a server hosting illegal transaction records (this is done in order to balance individual rights and society's safety). In this work we describe Traceable GE system, a group encryption with refined tracing capabilities akin to those of the primitive of "traceable signatures" (thus, balancing better privacy vs. safety). Our primitive enjoys the properties of group encryption, and, in addition, it allows the opening authority to reveal a user-specific trapdoor which makes it possible to publicly trace all the ciphertexts encrypted for that user without harming the anonymity of other ciphertexts. In addition, group members are able to non-interactively prove that specific ciphertexts are intended for them or not. This work provides rigorous definitions, concrete constructions in the standard model, and security proofs.

Keywords. Group encryption, traceability, anonymity, provable security, standard model.

1 Introduction

Group signatures [16] are a fundamental privacy primitive allowing members of a group to sign messages on behalf of the group while hiding their identity. To deter abuses, an authority is capable of identifying the author of any valid signature using privileged information. Group encryption (GE) is a primitive suggested by Kiayias, Tsiounis and Yung [30], which is the encryption analogue of group signatures [16]. Namely, it allows the sender of a ciphertext to hide the identity of the receiver within a population of certified users —under the control of a group manager (GM)— while providing universally verifiable guarantees that this receiver belongs to the group. If necessary, an opening authority (OA) is empowered with a key allowing it to "open" a ciphertext and pin down the receiver's identity in the same way as group signatures can be opened. Moreover, the system should support a mechanism allowing the sender to convince any verifier that (1) the ciphertext is well-formed and intended for some registered group member who will be able to decrypt; (2) the opening authority can identify the receiver if the need arises; (3) the plaintext satisfies certain properties such as being a witness for some public relation.

As a natural use case, group encryption allows a firewall to block all encrypted emails attempting to enter a network unless they are generated for some certified organization member and they carry a proof of malware-freeness. The GE primitive was also motivated by privacy applications such as anonymous trusted third parties (TTP) or oblivious retriever storage. In optimistic protocols, it allows verifiably encrypting messages to *anonymous* trusted third parties which remain offline most of their lifetime and only wake up when there is a problem to sort out. Group encryption provides a convenient way to hide the identity of users' preferred trusted third party, which can be

⋆ This author was supported by the CAMUS Walloon Region Project.

a privacy-sensitive piece information by itself as it can betray, e.g., the participant's citizenship.

Group encryption also finds applications in cloud storage systems. When encrypting datasets on a remote storage server, the sender can convince this server that the data is intended for some legitimate certified user without disclosing the latter's identity.

As exemplified in [30], group encryption also allows constructing hierarchical group signatures [38], where signers can flexibly specify how a set of trustees should operate to open their signatures.

Here we suggest a primitive extending the group encryption primitive and describe a refined traceabilty mechanism analogous to the way traceable signatures [29] extend group signatures. Specifically, when a given group member is suspected of conducting illegal activities, the opening authority is able to release a trapdoor allowing anyone to publicly trace ciphertexts encrypted for this member *without affecting the anonymity of other users*. As in the case of traceable signatures, the tracing trapdoor can be distributed to several tracing agents who can proceed in parallel when it comes to search for a given group member's ciphertexts. In contrast, in ordinary GE schemes, this task requires the OA to sequentially operate on all ciphertexts.

Related work. Kiayias, Tsiounis and Yung (KTY) [30] formalized the concept of group encryption and provided a modular design using zero-knowledge proofs, digital signatures, anonymous CCA-secure public-key encryption and commitment schemes. They also gave an efficient instantiation using Paillier's cryptosystem [36] and Camenisch-Lysyanskaya signatures [13]. While efficient, their scheme uses interactive proof systems. It can be made non-interactive using the Fiat-Shamir paradigm [20] at the expense of relying on the random oracle model [8], which is understood to only provide heuristic arguments (see, e.g., [21,14]) in terms of security.

Qin *et al.* [37] considered a sort of group encryption mechanism with non-interactive proofs and short ciphertexts. However, they appeal to random oracles and interactive assumptions in their security analysis. A non-interactive realization in the standard model was put forth by Cathalo, Libert and Yung [15]. More recently, El Aimani and Joye [18] considered more efficient interactive and non-interactive constructions using various optimizations.

As a matter of fact, none of the above solutions makes it possible to trace specific users' ciphertexts and only those ones. If messages encrypted for a specific misbehaving user have to be identified within a collection of, say $n = 100000$ ciphertexts, the opening authority has to open all of these in order to find those it is looking for. This is clearly harmful to the privacy of honest users who lose their anonymity just because they belong to the same group as a rogue user. In [29], Kiayias, Tsiounis and Yung suggested a technique to address this concern in the context of group signatures. To our knowledge, no real encryption analogue of their primitive has been studied so far.

The closest work addressing the problem at hand is that of Izabachène, Pointcheval and Vergnaud [26] who focus on eliminating subliminal channels by means of randomizable encryption. However, their mediated traceable anonymous encryption primitive does not provide all the functionalities we are aiming at. First, their scheme only provides message confidentiality and anonymity against *passive* adversaries, who have no access to decryption oracles at any time. Second, while their constructions enable individual user traceability, they do not provide a mechanism allowing the authority to identify the receiver of a ciphertext in $O(1)$ time. If their scheme is set up for groups of up to n users, their opening algorithm requires $O(n)$ operations in the worst case. Finally, the schemes of [26] provide no method allowing users to claim or disclaim ciphertexts they are the recipients of or not without disclosing their private keys.

Our contribution. This paper suggests a primitive called *traceable group encryption* (TGE) as the direct encryption analogue of traceable signatures, as suggested by Kiayias, Tsiounis and Yung [29]. Beyond the usual functionalities of group encryption, a TGE system allows the opening authority to reveal trapdoors associated with specific group members. These trapdoors enable the recognition of ciphertexts intended for these group members and leak no information about the identity of other ciphertexts' recipients. For example, when an employee leaves a company, the firewall can use a tracing trapdoor to sieve out all incoming ciphertexts encrypted for that former employee without learning anything else. As in the traceable signature scenario [29], this implicit tracing process can be run in parallel by clerks equipped with a copy of the tracing trapdoor.

In addition, similarly to the claiming mechanism of traceable signatures [29], TGE schemes support a procedure whereby group members are able to claim and prove that they are the legitimate receiver of some initially anonymous ciphertexts. Moreover, we further consider the dual problem of allowing group members to disclaim ciphertexts that are *not* encrypted under their public keys (this feature was not part of the original traceable signature model but it can be added on top of it in a modular way). Of course, our security notions explicitly require that group members be unable to falsely claim or disclaim ciphertexts.

The above claiming and disclaiming capabilities can serve in certain applications like cloud storage. While storage servers may require anonymous data retrievers to hold a certificate from some authority, the disclaiming procedure allows group members to convince investigators that they are not the intended recipient of some suspicious ciphertext without revealing their private key.

The first contribution of this paper is to define the primitive and to further provide stringent security definitions for traceable group encryption systems: like its group encryption counterpart [30], our model considers powerful adversaries who have oracle access to the private key functionalities of all users and authorities. As a second contribution, we provide a concrete construction and prove its security in the standard model under non-interactive assumptions. Our system is not just a proof of concept. At the 128-bit security level, ciphertexts and proofs fit within 2.18 and 9.38 kB, respectively. The efficiency is thus competitive with that of state-of-the-art group signatures [23] or traceable signatures [33] relying on non-interactive assumptions in the standard model.

2 Background

In the paper, when S is a set, $x \stackrel{R}{\leftarrow} S$ denotes the action of choosing x at random in S. By $a \in \mathsf{poly}(\lambda)$, we mean that a is a polynomial in λ while $b \in \mathsf{negl}(\lambda)$ says that b is a negligible function of λ. When a and b are two binary strings, $a\|b$ stands for their concatenation. For equal-dimension vectors $\vec{A}$ and $\vec{B}$ containing group elements, $\vec{A} \odot \vec{B}$ stands for their component-wise product.

2.1 Complexity Assumptions

We use groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order p with an efficiently computable map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G} \times \mathbb{G}$, $a, b \in \mathbb{Z}$ and $e(g, h) \neq 1_{\mathbb{G}_T}$ whenever $g, h \neq 1_{\mathbb{G}}$. In this setting, we consider several problems.

Definition 1 ([11]). *The* Decision Linear Problem *(DLIN) in $\mathbb{G}$, is to distinguish the distribution of linear tuples $D_1 = \{(g, g^a, g^b, g^{ac}, g^{bd}, g^{c+d}) \mid a, b, c, d \stackrel{R}{\leftarrow} \mathbb{Z}_p\}$ from the distribution of random tuples $D_2 = \{(g, g^a, g^b, g^{ac}, g^{bd}, g^z) \mid a, b, c, d, z \stackrel{R}{\leftarrow} \mathbb{Z}_p\}$.*

We also rely on the q-SFP problem, the generic hardness of which was proved by Abe *et al.* [1].

Definition 2 ([1]). *In a group $\mathbb{G}$, the q-*Simultaneous Flexible Pairing Problem *(q-SFP) is, given $(g_z, h_z, g_r, h_r, a, \tilde{a}, b, \tilde{b}) \in \mathbb{G}^8$ as well as q tuples $(z_j, r_j, s_j, t_j, u_j, v_j, w_j) \in \mathbb{G}^7$ such that*

$$e(a, \tilde{a}) = e(g_z, z_j) \cdot e(g_r, r_j) \cdot e(s_j, t_j) \quad \textit{and} \quad e(b, \tilde{b}) = e(h_z, z_j) \cdot e(h_r, u_j) \cdot e(v_j, w_j)\,, \tag{1}$$

to find a new tuple $(z^\star, r^\star, s^\star, t^\star, u^\star, v^\star, w^\star) \in \mathbb{G}^7$ satisfying (1) and such that $z^\star \notin \{1_{\mathbb{G}}, z_1, \ldots, z_q\}$.

Definition 3 (**[12]**). *The* Decision 3-party Diffie-Hellman Problem *(D3DH) in* $\mathbb{G}$*, is to distinguish the distributions* $(g, g^a, g^b, g^c, g^{abc})$ *and* (g, g^a, g^b, g^c, g^z)*, where* $a, b, c, z \xleftarrow{R} \mathbb{Z}_p$.

2.2 Groth-Sahai Proof Systems

In their instantiation based on the DLIN assumption in symmetric pairing configurations, the Groth-Sahai (GS) proof systems [24] use a common reference string (CRS) consisting of three vectors $\vec{g_1}, \vec{g_2}, \vec{g_3} \in \mathbb{G}^3$, where $\vec{g_1} = (g_1, 1, g)$, $\vec{g_2} = (1, g_2, g)$ for some $g_1, g_2 \in \mathbb{G}$. To commit to a group element $X \in \mathbb{G}$, the prover computes $\vec{C} = (1, 1, X) \odot \vec{g_1}^r \odot \vec{g_2}^s \odot \vec{g_3}^t$ with $r, s, t \xleftarrow{R} \mathbb{Z}_p$. When the proof system is configured to provide perfectly sound proofs, $\vec{g_3}$ is set as $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$ with $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p$. In this case, commitments $\vec{C} = (g_1^{r+\xi_1 t}, g_2^{s+\xi_2 t}, X \cdot g^{r+s+t(\xi_1+\xi_2)})$ can be interpreted as Boneh-Boyen-Shacham (BBS) ciphertexts as X can be recovered by running the BBS decryption algorithm using the private key $(\alpha_1, \alpha_2) = (\log_g(g_1), \log_g(g_2))$. When the CRS is set up to give perfectly witness indistinguishable (WI) proofs, $\vec{g_1}, \vec{g_2}$ and $\vec{g_3}$ are linearly independent vectors, so that $\vec{C}$ is a perfectly hiding commitment to $X \in \mathbb{G}$: a typical choice is $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2} \odot (1, 1, g)^{-1}$. Under the DLIN assumption, the two distributions of CRS are computationally indistinguishable.

To commit to an exponent $x \in \mathbb{Z}_p$, the prover computes $\vec{C} = \vec{\varphi}^x \odot \vec{g_1}^r \odot \vec{g_2}^s$, with $r, s \xleftarrow{R} \mathbb{Z}_p$, using a CRS containing $\vec{\varphi}, \vec{g_1}, \vec{g_2}$. In the perfect soundness setting $\vec{\varphi}, \vec{g_1}, \vec{g_2}$ are linearly independent (typically $\vec{\varphi} = \vec{g_3} \odot (1, 1, g)$ where $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$) whereas, in the perfect WI setting, choosing $\vec{\varphi} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$ yields perfectly hiding commitments since $\vec{C}$ is statistically independent of x.

To prove that committed variables satisfy a set of relations, the GS techniques replace variables by the corresponding commitments in each relation. The entire proof consists of one commitment per variable and one proof element (made of a constant number of elements) per relation.

Such proofs are available for pairing-product relations, which are equations of the type

$$\prod_{i=1}^{n} e(\mathcal{A}_i, \mathcal{X}_i) \cdot \prod_{i=1}^{n} \cdot \prod_{j=1}^{n} e(\mathcal{X}_i, \mathcal{X}_j)^{a_{ij}} = t_T, \tag{2}$$

for variables $\mathcal{X}_1, \ldots, \mathcal{X}_n \in \mathbb{G}$ and constants $t_T \in \mathbb{G}_T$, $\mathcal{A}_1, \ldots, \mathcal{A}_n \in \mathbb{G}$, $a_{ij} \in \mathbb{Z}_p$, for $i, j \in \{1, \ldots, n\}$. Efficient proofs also exist for multi-exponentiation equations like

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

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

Multi-exponentiation equations always admit non-interactive zero-knowledge (NIZK) proofs at no additional cost. On a perfectly witness indistinguishable CRS, a trapdoor (like the hidden exponents $(\xi_1, \xi_2) \in \mathbb{Z}_p^2$ when $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2} \odot (1, 1, g)^{-1}$) makes it possible to simulate proofs without knowing witnesses and simulated proofs are perfectly indistinguishable from real proofs. As for pairing-product equations, zero-knowledge proofs are often possible – this is usually the case when the right-hand-side member t_T of (2) is a product of pairings involving known group elements – but the number of group elements per proof may not be constant anymore. Here, when using such NIZK simulators, we just introduce a constant number of extra group elements in the proofs.

In both cases, proofs for quadratic equations cost 9 group elements. Linear pairing-product equations (when $a_{ij} = 0$ for all i, j) take 3 group elements each. Linear multi-exponentiation equations of the type $\prod_{j=1}^{n} \mathcal{X}_j^{b_j} = T$ (resp. $\prod_{i=1}^{m} \mathcal{A}_i^{y_i} = T$) demand 3 (resp. 2) group elements.

2.3 Chameleon Hash Functions

A chameleon hash function [32] is a tuple of algorithms $\mathsf{CMH} = (\mathsf{CMKg}, \mathsf{CMhash}, \mathsf{CMswitch})$ which contains an algorithm CMKg that, given a security parameter λ, outputs a key pair $(hk, tk) \leftarrow \mathcal{G}(\lambda)$. The randomized hashing algorithm outputs $y = \mathsf{CMhash}(hk, m, r)$ given the public key hk, a message m and random coins $r \in \mathcal{R}_{hash}$. On input of messages m, m', random coins $r \in \mathcal{R}_{hash}$ and the trapdoor key tk, the switching algorithm $r' \leftarrow \mathsf{CMswitch}(tk, m, r, m')$ computes $r' \in \mathcal{R}_{hash}$ such that $\mathsf{CMhash}(hk, m, r) = \mathsf{CMhash}(hk, m', r')$. The collision-resistance property mandates that it be infeasible to come up with pairs $(m', r') \neq (m, r)$ such that $\mathsf{CMhash}(hk, m, r) = \mathsf{CMhash}(hk, m', r')$ without knowing the trapdoor key tk. Uniformity guarantees that the distribution of hash values is independent of the message m: in particular, for all hk, and all messages m, m', the distributions $\{r \leftarrow \mathcal{R}_{hash} : \mathsf{CMHash}(hk, m, r)\}$ and $\{r \leftarrow \mathcal{R}_{hash} : \mathsf{CMHash}(hk, m', r)\}$ are identical.

3 Traceable Group Encryption

3.1 Syntax

Traceable group encryption (TGE) schemes involve a sender, a verifier, a group manager (GM) that manages the group of receivers and an opening authority (OA) that is able to uncover the identity of ciphertext receivers. A group encryption system is formally specified by the description of a relation $\mathcal{R}$ as well as a collection $\mathsf{TGE} = \big(\mathsf{SETUP}, \mathsf{JOIN}, \langle \mathcal{G}_r, \mathcal{R}, \mathsf{sample}_{\mathcal{R}} \rangle, \mathsf{ENC}, \mathsf{DEC}, \langle \mathcal{P}, \mathcal{V} \rangle, \mathsf{OPEN}, \mathsf{REVEAL}, \mathsf{TRACE}, \mathsf{CLAIM/DISCLAIM}, \mathsf{CLAIM\text{-}VERIFY}, \mathsf{DISCLAIM\text{-}VERIFY}\big)$ of algorithms or protocols. Among these, SETUP is a set of initialization procedures that all take (explicitly or implicitly) a security parameter λ as input. They can be split into one that generates a set of public parameters param (a common reference string), one for the GM and another one for the OA. We call them $\mathsf{SETUP}_{\mathsf{init}}(\lambda)$, $\mathsf{SETUP}_{\mathsf{GM}}(\mathsf{param})$ and $\mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$, respectively. The latter two procedures are used to produce key pairs $(\mathsf{pk}_{\mathsf{GM}}, \mathsf{sk}_{\mathsf{GM}})$, $(\mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}})$ for the GM and the OA. In the following, param is incorporated in the inputs of all algorithms although we sometimes omit it.

$\mathsf{JOIN} = (\mathsf{J}_{\mathsf{user}}, \mathsf{J}_{\mathsf{GM}})$ is an interactive protocol between the GM and the prospective user. As in [15], we aim for two-message protocols: the first message is the user's public key pk sent by $\mathsf{J}_{\mathsf{user}}$ to J_{GM} and the latter's response is a certificate $\mathsf{cert}_{\mathsf{pk}}$ for pk vouching for the user's group membership. The user is not required to prove knowledge of his private key sk. Valid public keys are assumed to be publicly recognizable, so that proofs of validity are not needed. After the execution of JOIN, the GM stores the public key pk and its certificate $\mathsf{cert}_{\mathsf{pk}}$ in a public directory $\mathsf{database}$.

Algorithm sample allows sampling pairs $(x, w) \in \mathcal{R}$ (comprised of a public value x and a witness w) using keys $(\mathsf{pk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{R}})$ produced by $\mathcal{G}_r$. Depending on the relation, $\mathsf{sk}_{\mathcal{R}}$ may be the empty string (as in the scheme we describe). The testing procedure $\mathcal{R}(x, w)$ returns 1 iff $(x, w) \in \mathcal{R}$. To encrypt a witness w such that $(x, w) \in \mathcal{R}$ for some public x, the sender picks the pair $(\mathsf{pk}, \mathsf{cert}_{\mathsf{pk}})$ from $\mathsf{database}$ and runs the encryption algorithm. The latter takes as input w, a label L, the receiver's pair $(\mathsf{pk}, \mathsf{cert}_{\mathsf{pk}})$ as well as public keys $\mathsf{pk}_{\mathsf{GM}}$ and $\mathsf{pk}_{\mathsf{OA}}$. Its output is a ciphertext $\psi \leftarrow \mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, w, L)$. On input of the same elements, the certificate $\mathsf{cert}_{\mathsf{pk}}$, the ciphertext ψ and the random coins $coins_{\psi}$ that were used to produce it, the non-interactive algorithm $\mathcal{P}$ generates a proof π_{ψ} that there exists a certified receiver whose public key was registered in $\mathsf{database}$ and that is able to decrypt ψ and obtain a witness w such that $(x, w) \in \mathcal{R}$. The verification algorithm $\mathcal{V}$ takes as input ψ, $\mathsf{pk}_{\mathsf{GM}}$, $\mathsf{pk}_{\mathsf{OA}}$, π_{ψ} and the description of $\mathcal{R}$ and outputs 0 or 1. Given ψ, L and the receiver's private key sk, the output of DEC is either a witness w such that $(x, w) \in \mathcal{R}$ or a rejection symbol $\perp$.

The next three algorithms provide explicit and implicit tracing capabilities. First, OPEN takes as input a ciphertext/label pair (ψ, L) and the OA's secret key $\mathsf{sk}_{\mathsf{OA}}$ and returns a receiver's identity i. Algorithm REVEAL takes as input the joining transcript $\mathsf{transcript}_i$ of user i and allows the OA to extract a tracing trapdoor trace_i using its private key $\mathsf{sk}_{\mathsf{OA}}$. This tracing trapdoor can be subsequently used to determine whether or not a given ciphertext-label pair (ψ, L) is a valid encryption under the public key pk_i of user i: namely, algorithm TRACE takes in public keys $\mathsf{pk}_{\mathsf{GM}}$ and $\mathsf{pk}_{\mathsf{OA}}$ as well as a pair (ψ, L) and the tracing trapdoor trace_i associated with user i. It returns 1 if and only if (ψ, L) is believed to be a valid encryption intended for user i.

Finally, the last three algorithms ($\mathsf{CLAIM/DISCLAIM}, \mathsf{CLAIM\text{-}VERIFY}, \mathsf{DISCLAIM\text{-}VERIFY}$) implement a functionality that allows user to convincingly claim or disclaim being the legitimate recipient of a given anonymous ciphertext. Concretely, $\mathsf{CLAIM/DISCLAIM}$ takes as input all public keys $(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk})$, a ciphertext-label pair (ψ, L) and a private key sk. It reveals a publicly verifiable piece of evidence τ that (ψ, L) is or is not a valid encryption under the public key pk. Algorithms $\mathsf{CLAIM\text{-}VERIFY}$ and $\mathsf{DISCLAIM\text{-}VERIFY}$ are then used to verify the assertion established by τ. They take as input all public keys, a pair (ψ, L) and a claim/disclaimer τ and output 1 or 0.

3.2 Security Definitions

Beyond the standard correctness requirement, our security model involves four properties called message privacy, anonymity, soundness and claiming soundness. In the definitions hereunder, we use the notation $\langle \mathsf{output}_A | \mathsf{output}_B \rangle \leftarrow \langle A(\mathsf{input}_A), B(\mathsf{input}_B)\rangle(\mathsf{common\text{-}input})$ to denote the execution of a protocol between A and B obtaining their own outputs from their respective inputs.

Correctness. This property requires the following experiment to return 1 w.h.p.

```
Experiment $\mathbf{Expt}^{\mathrm{correctness}}(\lambda)$
    $\mathsf{param} \leftarrow \mathsf{SETUP}_{\mathsf{init}}(\lambda)$; $(\mathsf{pk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{R}}) \leftarrow \mathcal{G}_r(\lambda)$; $(x, w) \leftarrow \mathsf{sample}_{\mathcal{R}}(\mathsf{pk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{R}})$;
    $(\mathsf{pk}_{\mathsf{GM}}, \mathsf{sk}_{\mathsf{GM}}) \leftarrow \mathsf{SETUP}_{\mathsf{GM}}(\mathsf{param})$; $(\mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}}) \leftarrow \mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$;
    $\langle \mathsf{pk}_i, \mathsf{sk}_i, \mathsf{cert}_{\mathsf{pk}_i} | \mathsf{pk}_i, \mathsf{cert}_{\mathsf{pk}_i} \rangle \leftarrow \langle \mathsf{J}_{\mathsf{user}}, \mathsf{J}_{\mathsf{GM}}(\mathsf{sk}_{\mathsf{GM}}) \rangle(\mathsf{pk}_{\mathsf{GM}})$;
    $\psi \leftarrow \mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_i, \mathsf{cert}_{\mathsf{pk}_i}, w, L)$;
    $\pi_\psi \leftarrow \mathcal{P}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_i, \mathsf{cert}_{\mathsf{pk}_i}, x, w, L, \psi, coins_\psi)$;
    If $\big((w \neq \mathsf{DEC}(\mathsf{sk}_i, \psi, L)) \vee (i \neq \mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}}, \psi, L))$
        $\vee (\mathcal{V}(\psi, L, \pi_\psi, \mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}) = 0)\big)$ return 0 else return 1.
```

Message privacy. The message privacy property is defined by an experiment where the adversary has access to oracles that may be stateful (and maintain a state across queries) or stateless:

- $\mathsf{DEC}(\mathsf{sk})$: is a stateless oracle for the user decryption function DEC. When this oracle is restricted not to decrypt a ciphertext-label pair (ψ, L), we denote it by $\mathsf{DEC}^{\neg\langle\psi,L\rangle}$.
- $\mathsf{CH}^b_{\mathsf{ror}}(\lambda, \mathsf{pk}, w, L)$: is a real-or-random challenge oracle that is only queried once. It returns $(\psi, coins_\psi)$ such that $\psi \leftarrow \mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, w, L)$ if $b = 1$ whereas, if $b = 0$, $\psi \leftarrow \mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, w', L)$ encrypts a random plaintext uniformly chosen in the space of plaintexts of length $O(\lambda)$. In either case, $coins_\psi$ are the random coins used to generate ψ.
- $\mathsf{PROVE}^b_{\mathcal{P},\mathcal{P}'}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, \mathsf{pk}_{\mathcal{R}}, x, w, \psi, L, coins_\psi)$: is a stateful oracle that the adversary can query on multiple occasions. If $b = 1$, it runs the real prover $\mathcal{P}$ on the inputs to produce an actual proof π_ψ. If $b = 0$, the oracle runs a simulator $\mathcal{P}'$ that uses the same inputs as $\mathcal{P}$ except witness $w, coins_\psi$ and generates a simulated proof.
- $\mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, L, \mathsf{sk})$: is a stateful oracle that allows the adversary to obtain either claims or disclaimer proofs for arbitrary ciphertexts. Specifically, the oracle first uses the private key sk to determine whether (ψ, L) is a valid ciphertext-label pair w.r.t. the public key pk. If so, the oracle uses sk to compute and return a non-interactive claim τ for ψ. Otherwise, the oracle generate a disclaimer proof τ showing that (ψ, L) is not a valid encryption under pk. In either case, (ψ, L) is stored in a list claims, which is initially empty.

These oracles are used in an experiment where the adversary controls the GM, the OA and all members but the honest receiver. The adversary $\mathcal{A}$ is the dishonest GM that certifies the honest receiver in an execution of JOIN. It has oracle access to the decryption function DEC of that receiver. At the challenge phase, it probes the challenge oracle for a label and a pair $(x, w) \in \mathcal{R}$ of her choice. After the challenge phase, $\mathcal{A}$ can also invoke the PROVE oracle on multiple occasions and eventually aims to guess the bit b chosen by the challenger.

As pointed out in [30], designing an efficient simulator $\mathcal{P}'$ (for executing $\mathsf{PROVE}^b_{\mathcal{P},\mathcal{P}'}(.)$ when $b = 0$) is part of the security proof and might require a simulated common reference string.

Definition 4. *A TGE scheme satisfies message security if, for any PPT adversary $\mathcal{A}$, the experiment below returns 1 with probability at most $1/2 + \mathsf{negl}(\lambda)$.*

```
Experiment $\mathbf{Expt}^{\mathrm{sec}}_{\mathcal{A}}(\lambda)$
    $\mathsf{param} \leftarrow \mathsf{SETUP}_{\mathsf{init}}(\lambda)$; $(\mathsf{aux}, \mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}) \leftarrow \mathcal{A}(\mathsf{param})$;
    $\langle \mathsf{pk}, \mathsf{sk}, \mathsf{cert}_{\mathsf{pk}} | \mathsf{aux} \rangle \leftarrow \langle \mathsf{J}_{\mathsf{user}}, \mathcal{A}(\mathsf{aux}) \rangle (\mathsf{pk}_{\mathsf{GM}})$;
    $(\mathsf{aux}, x, w, L, \mathsf{pk}_{\mathcal{R}}) \leftarrow \mathcal{A}^{\mathsf{DEC}(\mathsf{sk},.),\ \mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}},.,.,\mathsf{sk})}(\mathsf{aux})$; If $(x, w) \notin \mathcal{R}$ return 0;
    $b \overset{R}{\leftarrow} \{0, 1\}$; $(\psi, coins_{\psi}) \leftarrow \mathsf{CH}^b_{\mathsf{ror}}(\lambda, \mathsf{pk}, w, L)$;
    $b' \leftarrow \mathcal{A}^{\mathsf{PROVE}^b_{\mathcal{P},\mathcal{P}'}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, \mathsf{pk}_{\mathcal{R}}, x, w, \psi, L, coins_{\psi}), \mathsf{DEC}^{\neg\langle \psi, L \rangle}(\mathsf{sk},.), \mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}},.,.,\mathsf{sk})}(\mathsf{aux}, \psi)$;
    If $b = b'$ return 1 else return 0.
```

Anonymity. In anonymity attacks, the adversary controls the entire system except the opening authority. One way to jeopardize the anonymity property is to mount a chosen-ciphertext attack on the encryption scheme used by the OA. A difference with the usual group encryption scenario is that we must pay attention to the information revealed by the traceability components of ciphertexts. Throughout the game, the adversary can act as a dishonest group manager and register honest users in the system. In the challenge phase, the adversary A will choose a pair $(x, w) \in \mathcal{R}$ as well as the public keys $\mathsf{pk}_0, \mathsf{pk}_1$ of two honest users. In return, it will receive an encryption of w under the public key pk_b for some $b \in \{0, 1\}$ chosen by the challenger. It has access to the following oracles:

- $\mathsf{USER}(\mathsf{pk}_{\mathsf{GM}})$: is a stateful oracle simulating executions of $\mathsf{J}_{\mathsf{user}}$ on behalf of honest users who are requested to join the group. It uses an initially empty list keys. At its i-th invocation, the output $(i, \mathsf{pk}_i, \mathsf{sk}_i, \mathsf{cert}_{\mathsf{pk}_i})$ of $\mathsf{J}_{\mathsf{user}}$ is stored in keys if the J_{GM}-executing $\mathcal{A}$ provides a valid certificate $\mathsf{cert}_{\mathsf{pk}_i}$. If the JOIN protocol does not successfully terminate, the oracle stores $(i, \perp)$ in keys.
- $\mathsf{CORR}(.)$: is a stateful oracle that allows the adversary to corrupt honest group members. When invoked on input of an index i, the oracle first checks if the list keys contains an entry of the form $(i, \mathsf{pk}_i, \mathsf{sk}_i, \mathsf{cert}_{\mathsf{pk}_i})$. If so, it returns sk_i and adds i to the set Corr, which is initially empty.
- $\mathsf{DEC}(.,.)$: is a stateless decryption oracle that extends the one of the message security property in that it provides a decryption capability for each secret key. It takes as input an index i and a ciphertext-label pair (ψ, L). It first checks if the list keys contains an entry of the form $(i, \mathsf{pk}_i, \mathsf{sk}_i, \mathsf{cert}_{\mathsf{pk}_i})$. If no such entry exists, it returns $\perp$. Otherwise, it uses sk_i to run DEC on the input (ψ, L) and returns the result. When this oracle is restricted not to decrypt a ciphertext-label pair (ψ, L) for some user index $i \in \{i_0, i_1\}$, we denote it by $\mathsf{DEC}^{\neg\{i_0, i_1\} \times \langle \psi, L \rangle}$.
- $\mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}}, .)$: is a stateless oracle that simulates the opening algorithm on behalf of the OA and, on input of a TGE ciphertext, returns the receiver's identity i.
- $\mathsf{REVEAL}(\mathsf{sk}_{\mathsf{OA}}, .)$: is an oracle that takes as input a user index i and simulates the REVEAL algorithm on behalf of the OA. If no user was assigned the index i in keys, it returns $\perp$. Otherwise, it recovers the transcript $\mathsf{transcript}_i$ of user i in $\mathsf{database}$ and uses $\mathsf{sk}_{\mathsf{OA}}$ to extract and return the i-th group member's tracing trapdoor trace_i. It also adds i to the set Revs.

– $\mathsf{CH}^b_{\mathsf{anon}}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_0, \mathsf{pk}_1, w, L)$: is a challenge oracle that can only be queried once. It returns a pair $(\psi, coins_\psi)$ consisting of a ciphertext $\psi \leftarrow \mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_b, \mathsf{cert}_{\mathsf{pk}_b}, w, L)$ and the coin tosses $coins_\psi$ that were used to generate ψ.
– $\mathcal{P}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_b, \mathsf{cert}_{\mathsf{pk}_b}, \mathsf{pk}_{\mathcal{R}}, x, w, \psi, L, coins_\psi)$: is a stateful oracle which the adversary can query several times after the challenge phase. It runs the real prover $\mathcal{P}$ on the inputs to produce an actual proof π_ψ using the random coins $coins_\psi$ involved in the generation of the challenge ciphertext. It returns the resulting proof π_ψ.
– $\mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, L, i)$: is a stateful oracle that extends the one in the game modeling the message security property. It takes as input an index i and a ciphertext/label pair. It first checks whether keys contains a tuple $\mathsf{transcript}_i = (i, \mathsf{pk}_i, \mathsf{sk}_i, \mathsf{cert}_{\mathsf{pk}_i})$. If not, it returns $\perp$. Otherwise, it uses the private key sk_i to determine whether (ψ, L) is a valid ciphertext-label pair w.r.t. the public key pk_i. If yes, the oracle uses sk_i to generate a non-interactive claim τ for (ψ, L). Otherwise, the oracle generates a disclaimer τ guaranteeing that (ψ, L) is not a valid encryption under pk_i. In either case, (i, ψ, L) is stored in a list claims, which is initially empty.

Definition 5. *A TGE scheme satisfies anonymity if, for any PPT adversary $\mathcal{A}$, the experiment below returns 1 with a probability not exceeding $1/2 + \mathsf{negl}(\lambda)$.*

```
Experiment $\mathbf{Expt}^{\mathrm{anon}}_{\mathcal{A}}(\lambda)$
    $\mathsf{param} \leftarrow \mathsf{SETUP}_{\mathsf{init}}(\lambda)$; $(\mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}}) \leftarrow \mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$;
    $(\mathsf{aux}, \mathsf{pk}_{\mathsf{GM}}) \leftarrow \mathcal{A}(\mathsf{param}, \mathsf{pk}_{\mathsf{OA}})$;
    $(i_0, i_1, \mathsf{aux}, x, w, L, \mathsf{pk}_{\mathcal{R}})$
        $\leftarrow \mathcal{A}^{\mathsf{USER}(\mathsf{pk}_{\mathsf{GM}}),\ \mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}},.),\ \mathsf{REVEAL}(\mathsf{sk}_{\mathsf{OA}},.),\ \mathsf{DEC}(.,.),\ \mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}},\mathsf{pk}_{\mathsf{OA}},.,.,.),\ \mathsf{CORR}(.)}(\mathsf{aux})$;
    If $(i_0, \mathsf{pk}_0, \mathsf{sk}_0, \mathsf{cert}_{\mathsf{pk}_0}) \not\in \mathsf{keys} \ \vee\ (i_1, \mathsf{pk}_1, \mathsf{sk}_1, \mathsf{cert}_{\mathsf{pk}_1}) \not\in \mathsf{keys}$ return 0;
    If $(x, w) \not\in \mathcal{R}$ return 0;
    $b \overset{R}{\leftarrow} \{0,1\}$; $(\psi, coins_\psi) \leftarrow \mathsf{CH}^b_{\mathsf{anon}}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_0, \mathsf{pk}_1, w, L)$;
    $b' \leftarrow \mathcal{A}^{\mathsf{USER}(\mathsf{pk}_{\mathsf{GM}}),\ \mathcal{P}(\mathsf{pk}_{\mathsf{GM}},\mathsf{pk}_{\mathsf{OA}},\mathsf{pk}_b,\mathsf{cert}_{\mathsf{pk}_b},x,w,\psi,L,coins_\psi),\ \mathsf{OPEN}^{\neg\langle\psi,L\rangle}(\mathsf{sk}_{\mathsf{OA}},.),}$
        ${}^{\mathsf{REVEAL}^{\neg\{i_0,i_1\}}(\mathsf{sk}_{\mathsf{OA}},.),\ \mathsf{DEC}^{\neg\{i_0,i_1\}\times\langle\psi,L\rangle}(.,.),\ \mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}},\mathsf{pk}_{\mathsf{OA}},.,.,.),\ \mathsf{CORR}(.)}(\mathsf{aux}, \psi)$;
    If $\big((i_0, \psi, L) \in \mathsf{claims}\big) \ \vee\ \big((i_1, \psi, L) \in \mathsf{claims}\big)$ return 0;
    If $(i_0 \in \mathsf{Revs} \cup \mathsf{Corr}) \ \vee\ (i_1 \in \mathsf{Revs} \cup \mathsf{Corr})$ return 0;
    If $b = b'$ return 1 else return 0.
```

As shown in [30], TGE schemes satisfying the above notion necessarily subsume a key-private (a.k.a. receiver anonymous) [6,25] cryptosystem.

Soundness. In a soundness attack, the adversary creates the group of receivers by interacting with the honest GM. Its goal is to create a ciphertext ψ and a convincing proof that ψ is valid w.r.t. a relation $\mathcal{R}$ of its choice but either (1) the opening fails to identify a certified group member as the legitimate recipient of ψ; (2) the implicit tracing mechanism TRACE does not point to the group member pinned down by OPEN; (3) the ciphertext C is not in the language

$$\mathcal{L}^{x,L,\mathsf{pk}_{\mathcal{R}},\mathsf{pk}_{\mathsf{GM}},\mathsf{pk}_{\mathsf{OA}},\mathsf{pk}_i} = \{\mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_i, \mathsf{cert}_{\mathsf{pk}_i}, w, L) \mid (x, w) \in \mathcal{R};\ (\mathsf{pk}_i, \mathsf{cert}_{\mathsf{pk}_i}) \in \mathsf{valid}\},$$

where valid is the set of properly certified keys and i is the user identified by the opening algorithm. This notion is formalized by a game where the adversary is given access to a user registration oracle $\mathsf{REG}(\mathsf{sk}_{\mathsf{GM}}, .)$ that emulates J_{GM}. This oracle maintains a repository $\mathsf{database}$ where registered public keys and their certificates are stored.

Definition 6. *A TGE scheme is sound if, for any PPT adversary $\mathcal{A}$, the experiment below returns 1 with negligible probability.*

```
Experiment $\mathbf{Expt}_{\mathcal{A}}^{\text{soundness}}(\lambda)$
    $\mathsf{param} \leftarrow \mathsf{SETUP}_{\mathsf{init}}(\lambda)$; $(\mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}}) \leftarrow \mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$;
    $(\mathsf{pk}_{\mathsf{GM}}, \mathsf{sk}_{\mathsf{GM}}) \leftarrow \mathsf{SETUP}_{\mathsf{GM}}(\mathsf{param})$;
    $(\mathsf{pk}_{\mathcal{R}}, x, \psi, \pi_\psi, L, \mathsf{aux}) \leftarrow \mathcal{A}^{\mathsf{REG}(\mathsf{sk}_{\mathsf{GM}},.)}(\mathsf{param}, \mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}})$;
    If $\mathcal{V}(\psi, L, \pi_\psi, \mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}) = 0$ return $0$;
    $i \leftarrow \mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}}, \psi, L)$;
    If $\big((i = \perp) \vee (\psi \notin \mathcal{L}^{x,L,\mathsf{pk}_{\mathcal{R}},\mathsf{pk}_{\mathsf{GM}},\mathsf{pk}_{\mathsf{OA}},\mathsf{pk}_i})\big)$ then return $1$;
    $\mathsf{trace}_i \leftarrow \mathsf{REVEAL}(\mathsf{transcript}_i, \mathsf{sk}_{\mathsf{OA}})$;
    If $\big(i \neq \mathsf{TRACE}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, \mathsf{trace}_i)\big)$ then return $1$;
    Return $0$.
```

The above security properties are broadly similar to that for group encryption. We need to introduce the new notion of *claiming soundness* that formalizes the soundness of the claiming process.

Claiming soundness. The last security notion considers an adversary attacking the soundness of the claiming algorithm by either claiming other users' ciphertexts as its own or disclaiming ciphertexts that are actually encrypted under its public key. Moreover, the verifier of a claim/disclaimer should be convinced of the group member's intentionality to claim or repudiate ciphertexts. We require that only users be able to claim/disclaim ciphertexts encrypted under their key or not: even the sender (who knows the encryption coins) should be unable to do this.

In the attack model, the adversary controls both the GM and the OA. It is given access to oracles $\mathsf{USER}(\mathsf{pk}_{\mathsf{GM}})$, $\mathsf{CORR}(.)$, $\mathsf{DEC}(.,.)$ and $\mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, L, i)$, which are identical to those of the anonymity property. The adversary's goal is to create a public repository $\mathsf{database}$ satisfying the integrity check, a ciphertext ψ and a statement $\mathsf{statement}$ consisting of a claim/disclaimer τ and a public key pk but either: (1) the implicit tracing mechanism TRACE does not point to the group member i pinned down by OPEN; (2) $\mathsf{statement} = (\tau, \mathsf{pk})$ is a valid claim although $\mathsf{pk} \neq \mathsf{pk}_i$, where pk_i is associated with user i in $\mathsf{database}$; (3) $\mathsf{statement} = (\tau, \mathsf{pk})$ is a valid disclaimer whereas $\mathsf{pk} = \mathsf{pk}_i$ coincides with the public key associated with user i in $\mathsf{database}$; (4) $\mathsf{statement} = (\tau, \mathsf{pk}_j)$ is a valid claim/disclaimer for the public key pk_j of some uncorrupted user $j \in \mathsf{database} \backslash \mathsf{Corr}$ in the database and the pair (τ, pk_j) was not produced by the $\mathsf{CLAIM/DISCLAIM}$ oracle.

Definition 7. *A TGE scheme provides claiming-soundness if, for any PPT adversary $\mathcal{A}$, the experiment below returns 1 with negligible probability.*

```
Experiment $\mathbf{Expt}_{\mathcal{A}}^{\text{claiming-soundness}}(\lambda)$
    $\mathsf{param} \leftarrow \mathsf{SETUP}_{\mathsf{init}}(\lambda)$; $(\mathsf{pk}_{\mathsf{GM}}, \mathsf{aux}_0) \leftarrow \mathcal{A}(\mathsf{param})$; $(\mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}}) \leftarrow \mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$;
      $(\mathsf{pk}_{\mathcal{R}}^\star, x^\star, \psi^\star, L^\star, \pi_\psi^\star, \mathsf{statement}^\star, \mathsf{database}^\star, \mathsf{aux})$
            $\leftarrow \mathcal{A}^{\mathsf{USER}(\mathsf{pk}_{\mathsf{GM}}),\ \mathsf{CORR}(.),\ \mathsf{DEC}(.,.),\ \mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}},\mathsf{pk}_{\mathsf{OA}},.,.,.)}(\mathsf{param}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}}, \mathsf{aux}_0)$;
    If $\mathsf{DATABASE\text{-}CHECK}(\mathsf{param}, \mathsf{database}) = 0$ return $0$;
    If $\mathcal{V}(\psi^\star, L^\star, \pi_\psi^\star, \mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}) = 0$ return $0$;
    $i \leftarrow \mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}}, \psi^\star, L^\star)$;   If $i = \perp$ return $0$;   $\mathsf{trace}_i \leftarrow \mathsf{REVEAL}(\mathsf{transcript}_i, \mathsf{sk}_{\mathsf{OA}})$;
    If $\big(i \neq \mathsf{TRACE}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi^\star, \mathsf{trace}_i)\big)$ then return $1$;
    If $\big(\mathsf{statement}^\star = (\tau^\star, \mathsf{pk}^\star)$ s.t. $\mathsf{CLAIM\text{-}VERIFY}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi^\star, L^\star, \mathsf{pk}^\star, \tau^\star) = 1$
        $\wedge\ (\mathsf{pk}^\star \neq \mathsf{pk}_i)\big)$ then return $1$;
    If $\big(\mathsf{statement}^\star = (\tau^\star, \mathsf{pk}^\star)$ s.t. $\mathsf{DISCLAIM\text{-}VERIFY}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi^\star, L^\star, \mathsf{pk}^\star, \tau^\star) = 1$
        $\wedge\ (\mathsf{pk}^\star = \mathsf{pk}_i)\big)$ then return $1$;
    If $\Big(\mathsf{statement}^\star = (\tau^\star, \mathsf{pk}_j)$ s.t. $(j, \mathsf{pk}_j, \mathsf{cert}_j, .) \in \mathsf{database} \wedge (j \notin \mathsf{Corr}) \wedge (\psi^\star, L^\star, \mathsf{pk}_j) \notin Q_c$
          $\wedge\ \big(\mathsf{CLAIM\text{-}VERIFY}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi^\star, L^\star, \mathsf{pk}_j, \tau^\star) = 1$
          $\vee\ \mathsf{DISCLAIM\text{-}VERIFY}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi^\star, L^\star, \mathsf{pk}_j, \tau^\star) = 1\big)\Big)$ then return $1$;
    Return $0$.
```

In the above notations, Q_c denotes the set of queries to the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle made by $\mathcal{A}$.

We note that there is no need to provide the adversary with a REVEAL oracle in the definition. Indeed, since it knows $\mathsf{sk}_{\mathsf{OA}}$, it can obtain tracing trapdoors by itself, by decrypting the verifiable encryptions sent by honest users when the USER oracle is invoked.

4 A Non-Interactive Traceable Group Encryption Scheme

4.1 Intuition

We use the Libert-Yung (LY) scheme [34], a publicly verifiable variant of Cramer-Shoup recalled in Appendix A.2. We take advantage of the observation that, if certain public key components are shared by all users as common public parameters, the scheme can simultaneously provide receiver anonymity *and* publicly verifiable ciphertexts. In other words, anyone can publicly verify that a ciphertext is valid without knowing who the receiver is. When proofs are generated for the ciphertext, this saves the prover from having to provide evidence that the ciphertext is valid and thus yields shorter proofs.

The message is encrypted under the receiver's public key using the LY scheme. At the same time, the two last components of the receiver's public key is encrypted under the public key of the opening authority using Kiltz's encryption scheme [31] (see Appendix A.2). We use this scheme because it is the most efficient DLIN-based CCA2-secure cryptosytem where the validity of ciphertexts is publicly verifiable and we do not need it to hide the public key under which it is generated. We note that slightly shorter ciphertexts can be obtained using more efficient publicly verifiable variants of the Cramer-Shoup cryptosystem. For example, the techniques of [19,27,28] can be used for this purpose. We used the scheme of [34] to keep the description as simple as possible.

When new users join the group, the GM provides them with a membership certificate consisting of a structure-preserving signature on their public key which comprises group elements (X_1, X_2). We chose to work with the scheme of Abe, Haralambiev and Ohkubo (AHO) [1] (see Appendix A.1) because it allows working exclusively with linear pairing-product equations (and thus obtain a better efficiency) when non-interactive proofs are generated.

The implicit tracing mechanism must allow the OA to disclose user-specific tracing trapdoors. To this end, we include in each membership certificate a pair $(\varGamma_1, \varGamma_2) = (g^{\gamma_1}, g^{\gamma_2}) \in \mathbb{G}^2$, where $(\gamma_1, \gamma_2) \in \mathbb{Z}_p^2$ are part of the user's private key. When users join the group, they are thus requested to produce a pair $(\varGamma_1, \varGamma_2) = (g^{\gamma_1}, g^{\gamma_2})$ for which $g^{\gamma_1\gamma_2}$ will serve as a tracing trapdoor for them. Since $g^{\gamma_1\gamma_2}$ cannot be publicly revealed, we appeal to a verifiable encryption mechanism as was suggested in [9] in a related context: namely, the prospective user provides the GM with an encryption $\varPhi_{venc}$ of $g^{\gamma_1\gamma_2}$ under the OA's public key and generates a non-interactive proof that the encrypted value is indeed an element $g^{\gamma_1\gamma_2}$ such that $(g, g^{\gamma_1}, g^{\gamma_2}, g^{\gamma_1\gamma_2})$ is a Diffie-Hellman tuple. The REVEAL algorithm thus uses the OA's private key to decrypt $\varPhi_{venc}$ so as to expose $g^{\gamma_1\gamma_2}$. Armed with the information $\mathsf{trace}_i = g^{\gamma_1\gamma_2}$, a tracing agent can test whether a ciphertext ψ is prepared for user i as follows. We require each ciphertext ψ to contain elements of the form $(T_1, T_2, T_3) = (g^{\delta}, \varGamma_1^{\delta/\varrho}, \varGamma_2^{\varrho})$, where $\delta, \varrho \in_R \mathbb{Z}_p$ are chosen by the sender. Since $(\varGamma_1, \varGamma_2) = (g^{\gamma_1}, g^{\gamma_2})$, the TRACE algorithm concludes that user i is indeed the receiver if $e(T_1, g^{\gamma_1\gamma_2}) = e(T_2, T_3)$. At the same time, we can show that recognizing ciphertexts encrypted for user i without trace_i is as hard as solving the D3DH problem.

For technical reasons, we need to introduce an extra traceability component $T_4 = (\varLambda_0^{\mathsf{VK}} \cdot \varLambda_1)^{\delta}$, where $\varLambda_0, \varLambda_1 \in \mathbb{G}$ are part of common public parameters and VK is the verification key of a one-time signature. The reason is that, in order to prove anonymity in our model, we need to bind (T_1, T_2, T_3) to the one-time verification key VK in a non-malleable way. Otherwise, an anonymity adversary would be able to break the anonymity of the scheme by having access to a $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle.

To prove or disprove that he is the intended recipient of a given ciphertext-label pair (ψ, L), a user i can use the traceability components $(T_1, T_2, T_3) = (g^{\delta}, \Gamma_1^{\delta/\varrho}, \Gamma_2^{\varrho})$ of ψ and his private key $\gamma_1 = \log_g(\Gamma_1)$ to compute $\Gamma_1^{\delta} = T_1^{\gamma_1}$ (although he does not know δ), which allows anyone to realize that $(g, T_1, \Gamma_1, \Gamma_1^{\delta})$ forms a Diffie-Hellman tuple and that $e(\Gamma_1^{\delta}, \Gamma_2) = e(T_2, T_3)$. This is sufficient for proving that (ψ, L) was created for the public key $\mathsf{pk} = (X_1, X_2, \Gamma_1, \Gamma_2)$. In order to make sure that only the user will be able to compute non-interactive claims, we also require him to provide a non-interactive proof of knowledge of $\Gamma_{-1} = g^{1/\gamma_1}$ satisfying $e(\Gamma_1^{\delta}, \Gamma_{-1}) = e(T_1, g)$. Moreover, the claim is non-malleably bound to (ψ, L, pk) – where pk is the claimer's public key —by generating the non-interactive Groth-Sahai proof for a CRS $(\vec{g_1}, \vec{g_2}, \vec{h}_{\mathsf{v}})$ that depends on the ciphertext which is being claimed and the receiver's public key (the idea of data-dependent CRS is borrowed from [35]): this prevents malicious users from convincingly claiming other users' ciphertexts by intercepting honestly generated claims and presenting them as claims of their own. To eliminate an annoying case in the proof of anonymity, we chose to derive the vector $\vec{h}_{\mathsf{v}}$ from a bit string obtained by applying a chameleon hash function [32] (rather than a an ordinary hash function) to (ψ, L, pk).

4.2 Description

We build a non-interactive group encryption scheme for the Diffie-Hellman relation $\mathcal{R} = \{(X, Y), W\}$ where $e(g, W) = e(X, Y)$, for which the keys are $\mathsf{pk}_{\mathcal{R}} = \{\mathbb{G}, \mathbb{G}_T, g\}$ and $\mathsf{sk}_{\mathcal{R}} = \varepsilon$.

$\mathsf{SETUP}_{\mathsf{init}}(\lambda)$: Let $\ell \in \mathsf{poly}(\lambda)$ be a polynomial, where $\lambda \in \mathbb{N}$ is the security parameter. Generate public parameters according to the following steps.

1. Choose bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^{\lambda}$ with $g, g_1, g_2 \overset{R}{\leftarrow} \mathbb{G}$. Define vectors $\vec{g_1} = (g_1, 1, g)$, $\vec{g_2} = (1, g_2, g)$ and $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$ with $\xi_1, \xi_2 \overset{R}{\leftarrow} \mathbb{Z}_p$, which form a perfectly sound Groth-Sahai common reference string $\mathbf{g} = (\vec{g_1}, \vec{g_2}, \vec{g_3})$.
2. For $i = 0$ to ℓ choose $\zeta_{i,1}, \zeta_{i,2} \overset{R}{\leftarrow} \mathbb{Z}_p$ and set $\vec{h_i} = \vec{g_1}^{\zeta_{i,1}} \odot \vec{g_2}^{\zeta_{i,2}}$ so as to obtain vectors $\{\vec{h_i}\}_{i=0}^{\ell}$.
3. Choose $\eta_1, \eta_2 \overset{R}{\leftarrow} \mathbb{Z}_p$ and compute $\vec{f} = \vec{g_1}^{\eta_1} \odot \vec{g_2}^{\eta_2} = (f_{3,1}, f_{3,2}, f_{3,3})$ so as to form yet another CRS $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$, which will be used to prove statements about the ciphertexts.
4. Choose $\Lambda_0, \Lambda_1 \overset{R}{\leftarrow} \mathbb{G}$ at random.
5. Select a strongly unforgeable one time signature scheme $\Sigma = (\mathcal{G}, \mathcal{S}, \mathcal{V})$ and a chameleon hash function $\mathcal{CMH} = (\mathsf{CMKg}, \mathsf{CMhash}, \mathsf{CMswitch})$ with a key pair $(hk, tk) \leftarrow \mathcal{G}(\lambda)$.

Public parameters consists of $\mathsf{param} = \{\lambda, \mathbb{G}, \mathbb{G}_T, g, \vec{g_1}, \vec{g_2}, \vec{g_3}, \vec{f}, \{\vec{h_i}\}_{i=0}^{\ell}, \Lambda_0, \Lambda_1, \Sigma, \mathcal{CMH}, hk\}$.

$\mathsf{SETUP}_{\mathsf{GM}}(\mathsf{param})$: runs the setup algorithm of the structure-preserving signature of Abe *et al.* with $n = 4$. The private key is $\mathsf{sk}_{\mathsf{GM}} = \big(\alpha_a, \alpha_b, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^{4}\big)$ while the public key consists of

$$\mathsf{pk}_{\mathsf{GM}} = \big(G_r, H_u, G_z, H_z, \{G_i, H_i\}_{i=1}^{4},\ \Omega_a,\ \Omega_b\big) \in \mathbb{G}^8 \times \mathbb{G}_T^2 \ .$$

$\mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$: generates $\mathsf{pk}_{\mathsf{OA}} = (Y_1, Y_2, Y_3, Y_4) = (g^{y_1}, g^{y_2}, g^{y_3}, g^{y_4})$, as a public key for Kiltz's encryption scheme [31], and the corresponding private key as $\mathsf{sk}_{\mathsf{OA}} = (y_1, y_2, y_3, y_4)$.

JOIN : The prospective user $\mathcal{U}_i$ and the GM run the following protocol.

1. $\mathcal{U}_i$ chooses $x_1, x_2, z, \gamma_1, \gamma_2 \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes a public key $\mathsf{pk} = (X_1, X_2, \Gamma_1, \Gamma_2) \in \mathbb{G}^4$ where

$$X_1 = g_1^{x_1} \cdot g^z , \qquad X_2 = g_2^{x_2} \cdot g^z , \qquad \Gamma_1 = g^{\gamma_1} , \qquad \Gamma_2 = g^{\gamma_2} \ .$$

The private key is defined to be $\mathsf{sk} = (x_1, x_2, z, \gamma_1, \gamma_2)$. Here, (X_1, X_2) form a public key for the LY encryption scheme recalled in Appendix A.2 whereas (Γ_1, Γ_2) will provide user traceability.

2. $\mathcal{U}_i$ defines $\Gamma_0 = g^{\gamma_1\gamma_2}$ and generates a verifiable encryption of Γ_0 under $\mathsf{pk}_{\mathsf{OA}}$. To this end, he chooses $w_1, w_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and computes $\Phi_{venc} = (\Phi_0, \Phi_1, \Phi_2) = \big(\Gamma_0 \cdot g^{w_1+w_2},\ Y_1^{w_1},\ Y_2^{w_2}\big)$. Then, $\mathcal{U}_i$ generates a NIZK proof π_{venc} that Φ_{venc} encrypts Γ_0 such that $e(\Gamma_0, g) = e(\Gamma_1, \Gamma_2)$. Namely, $\mathcal{U}_i$ uses the CRS $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$ to generate GS commitments $\vec{C}_{W_1}, \vec{C}_{W_2}$ to the group elements $W_1 = g^{w_1}$ and $W_2 = g^{w_2}$, respectively, and non-interactively prove that

$$e(\Phi_0, g) = e(\Gamma_1, \Gamma_2) \cdot e(g, W_1) \cdot e(g, W_2) \quad (3)$$
$$e(\Phi_1, g) = e(Y_1, W_1) \quad (4)$$
$$e(\Phi_2, g) = e(Y_2, W_2) \quad (5)$$

Equations (3)–(5) are linear pairing product equations. However, since their proofs must be NIZK proofs, they cost 21 group elements to prove altogether[4]. We denote by π_{venc} the resulting NIZK proof. The prospective user $\mathcal{U}_i$ then sends the certification request consisting of $\big(\mathsf{pk} = (X_1, X_2, \Gamma_1, \Gamma_2), \Phi_{venc}, \vec{C}_{W_1}, \vec{C}_{W_2}, \pi_{venc}\big)$ to the GM.

3. If $\mathsf{database}$ already contains a record $\mathsf{transcript}_j$ for which the certified public key $\mathsf{pk}_j = (X_{j,1}, X_{j,2}, \Gamma_{j,1}, \Gamma_{j,2})$ is such that $(X_1, X_2) = (X_{j,1}, X_{j,2})$ or $e(\Gamma_{j,1}, \Gamma_{j,2}) = e(\Gamma_1, \Gamma_2)$, the GM returns $\perp$. Otherwise, the GM generates a certificate $\mathsf{cert}_{\mathsf{pk}} = (Z, R, S, T, U, V, W) \in \mathbb{G}^7$ for pk, which consists of an AHO signature on the tuple $(X_1, X_2, \Gamma_1, \Gamma_2)$. Then, it stores the entire interaction transcript

$$\mathsf{transcript}_i = \Big(\mathsf{pk} = (X_1, X_2, \Gamma_1, \Gamma_2), (\Phi_{venc}, \vec{C}_{W_1}, \vec{C}_{W_2}, \pi_{venc}), \mathsf{cert}_{\mathsf{pk}}\Big)$$

in $\mathsf{database}$. We also define the $\mathsf{DATABASE\text{-}CHECK}$ algorithm in such a way that it returns 0 (meaning that $\mathsf{database}$ is not well-formed) if $\mathsf{database}$ contains two distinct records $\mathsf{transcript}_i$ and $\mathsf{transcript}_j$ for which the corresponding public keys $\mathsf{pk}_i = (X_{i,1}, X_{i,2}, \Gamma_{i,1}, \Gamma_{i,2})$ and $\mathsf{pk}_j = (X_{j,1}, X_{j,2}, \Gamma_{j,1}, \Gamma_{j,2})$ are such that $(X_{i,1}, X_{i,2}) = (X_{j,1}, X_{j,2})$ or $e(\Gamma_{i,1}, \Gamma_{i,2}) = e(\Gamma_{j,1}, \Gamma_{j,2})$. Otherwise, it returns 1.

$\mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, M, L)$: To encrypt a witness $M \in \mathbb{G}$ such that $((A, B), M) \in \mathcal{R}_{dh}$ (for public $A, B \in \mathbb{G}$), parse $\mathsf{pk}_{\mathsf{GM}}$, $\mathsf{pk}_{\mathsf{OA}}$ and pk as $(X_1, X_2, \Gamma_1, \Gamma_2) \in \mathbb{G}^4$. Then, do the following.

1. Generate a one-time signature key pair $(\mathsf{SK}, \mathsf{VK}) \leftarrow \mathcal{G}(\lambda)$.
2. Generate traceability components $(T_1, T_2, T_3, T_4) \in \mathbb{G}^4$ by choosing $\delta, \varrho \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and computing

$$T_1 = g^{\delta}, \qquad T_2 = \Gamma_1^{\delta/\varrho}, \qquad T_3 = \Gamma_2^{\varrho}, \qquad T_4 = (\Lambda_0^{\mathsf{VK}} \cdot \Lambda_1)^{\delta} \ . \quad (6)$$

3. Compute a LY encryption of M under the label L. To this end, conduct the following steps.

   (a) Choose $\theta_1, \theta_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and compute

   $$C_0 = M \cdot X_1^{\theta_1} \cdot X_2^{\theta_2}, \qquad C_1 = g_1^{\theta_1}, \qquad C_2 = g_2^{\theta_2}, \qquad C_3 = g^{\theta_1+\theta_2} \ .$$

   (b) Construct a vector $\vec{g}_{\mathsf{VK}} = \vec{g_3} \cdot (1, 1, g)^{\mathsf{VK}}$ and use $\mathbf{g}_{\mathsf{VK}} = (\vec{g_1}, \vec{g_2}, \vec{g}_{\mathsf{VK}})$ as a Groth-Sahai CRS to generate a NIZK proof that $(g, g_1, g_2, C_1, C_2, C_3)$ form a linear tuple. More precisely, generate commitments $\vec{C}_{\theta_1}$, $\vec{C}_{\theta_2}$ to $\theta_1, \theta_2 \in \mathbb{Z}_p$ (namely, compute $\vec{C}_{\theta_i} = \vec{g}_{\mathsf{VK}}^{\,\theta_i} \cdot \vec{g_1}^{\,r_i} \cdot \vec{g_2}^{\,s_i}$ with $r_i, s_i \stackrel{R}{\leftarrow} \mathbb{Z}_p$ for each $i \in \{1, 2\}$) and a proof π_{LIN} that they satisfy

   $$C_1 = g_1^{\theta_1}, \qquad C_2 = g_2^{\theta_2}, \qquad C_3 = g^{\theta_1+\theta_2} \ . \quad (7)$$

[4] Namely, the prover actually generates proofs for the five relations $e(\Phi_0, \mathcal{X}_g) = e(\mathcal{X}_{\Gamma_1}, \Gamma_2) \cdot e(g, W_1) \cdot e(g, W_2)$, $e(\Phi_1, \mathcal{X}_g) = e(Y_1, W_1)$, $e(\Phi_2, \mathcal{X}_g) = e(Y_2, W_2)$ and $\mathcal{X}_g = g$, $\mathcal{X}_{\Gamma_1} = \Gamma_1$, by introducing auxiliary variables $\mathcal{X}_g, \mathcal{X}_{\Gamma_1}$. On a fake CRS, simulated proofs can be obtained using the assignment $\mathcal{X}_g = \mathcal{X}_{\Gamma_1} = W_1 = W_2 = 1_{\mathbb{G}}$ to prove the first three relations. The trapdoor of the CRS then allows computing fake proofs that $\mathcal{X}_g = g$, $\mathcal{X}_{\Gamma_1} = \Gamma_1$.

The whole proof for (7) consists of $\vec{C}_{\theta_1}$, $\vec{C}_{\theta_2}$ and π_{LIN} is obtained as

$$\pi_{\mathrm{LIN}} = (\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6) = \left(g_1^{r_1}, g_1^{s_1}, g_2^{r_2}, g_2^{s_2}, g^{r_1+r_2}, g^{s_1+s_2}\right) .$$

(c) Define the partial LY ciphertext $\psi_{\mathsf{LY}} = (C_0, C_1, C_2, C_3, \vec{C}_{\theta_1}, \vec{C}_{\theta_2}, \pi_{\mathrm{LIN}})$.

4. For $i = 1, 2$, choose $z_{i,1}, z_{i,2} \xleftarrow{R} \mathbb{Z}_p$ and encrypt Γ_i under $\mathsf{pk}_{\mathsf{OA}}$ using Kiltz's cryptosystem using the same one-time verification key VK as in step 1. Let $\{\psi_{\mathsf{K}_i}\}_{i=1,2}$ be the ciphertexts.
5. Set the TGE ciphertext ψ as $\psi = \mathsf{VK}\|(T_1, T_2, T_3, T_4)\|\psi_{\mathsf{LY}}\|\psi_{\mathsf{K}_1}\|\psi_{\mathsf{K}_2}\|\sigma$ where σ is a one-time signature obtained as $\sigma = \mathcal{S}(\mathsf{SK}, ((T_1, T_2, T_3, T_4)\|\psi_{\mathsf{LY}}\|\psi_{\mathsf{K}_1}\|\psi_{\mathsf{K}_2}\|L))$.

Return (ψ, L) and $coins_\psi$ consist of δ, ϱ, $\{(z_{i,1}, z_{i,2})\}_{i=1}^2$ and (θ_1, θ_2). If the one-time signature of [22] is used, the pair (VK, σ) takes 5 group elements, so that ψ comprises 35 elements of $\mathbb{G}$.

$\mathcal{P}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, (X, Y), M, \psi, L, coins_\psi)$: Parse $\mathsf{pk}_{\mathsf{GM}}$, $\mathsf{pk}_{\mathsf{OA}}$, pk and ψ as above. Using the vectors $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$ as a Groth-Sahai CRS, generate a non-interactive proof for ψ as follows.

1. Parse the certificate $\mathsf{cert}_{\mathsf{pk}}$ as $(Z, R, S, T, U, V, W) \in \mathbb{G}^7$ and re-randomize it (as explained in Appendix A.1) to obtain $(Z', R', S', T', U', V') \leftarrow \mathsf{ReRand}(\mathsf{pk}_{\mathsf{GM}}, (Z, R, S, T, U, V, W))$. Generate GS commitments $\vec{C}_{Z'}, \vec{C}_{R'}, \vec{C}_{U'}$ to Z', R' and U'. The overall commitment to $\mathsf{cert}_{\mathsf{pk}}$ is $com_{\mathsf{cert}_{\mathsf{pk}}} = (\vec{C}_{Z'}, \vec{C}_{R'}, \vec{C}_{U'}, S', T', V', W') \in \mathbb{G}^{13}$.
2. Generate GS commitments to the components of the public key $\mathsf{pk} = (X_1, X_2, \Gamma_1, \Gamma_2)$ and obtain the set $com_{\mathsf{pk}} = \{\vec{C}_{X_i}, \vec{C}_{\Gamma_i}\}_{i=1,2}$, which consists of 12 group elements.
3. Generate a proof $\pi_{\mathsf{cert}_{\mathsf{pk}}}$ that $com_{\mathsf{cert}_{\mathsf{pk}}}$ is a commitment to a valid certificate for the public key contained in com_{pk}. The proof $\pi_{\mathsf{cert}_{\mathsf{pk}}}$ is a non-interactive proof that committed group elements (Z', R', U') satisfy the relations

$$\Omega_a \cdot e(S', T')^{-1} \cdot \prod_{i=1}^{2} e(G_i, X_i)^{-1} \cdot \prod_{i=1}^{2} e(G_{i+2}, \Gamma_i)^{-1} = e(G_z, Z') \cdot e(G_r, R') ,$$
$$\Omega_b \cdot e(V', W')^{-1} \cdot \prod_{i=1}^{2} e(H_i, X_i)^{-1} \cdot \prod_{i=1}^{2} e(H_{i+2}, \Gamma_i)^{-1} = e(H_z, Z') \cdot e(H_u, U') ,$$

which cost 3 elements each. The whole proof $\pi_{\mathsf{cert}_{\mathsf{pk}}}$ thus takes 6 group elements.

4. Generate a NIZK proof π_T that (T_1, T_2, T_3) satisfies $(T_1, T_2, T_3) = (g^\delta, \Gamma_1^{\delta/\varrho}, \Gamma_2^{\varrho})$ for some $\delta, \varrho \in \mathbb{Z}_p$. To this end, generate a commitment $\vec{C}_\Upsilon$ to the group element $\Upsilon = g^{\delta/\varrho}$ and generate a NIZK proof that

$$e(\Upsilon, T_3) = e(T_1, \Gamma_2) , \qquad (8)$$
$$e(T_2, g) = e(\Gamma_1, \Upsilon) . \qquad (9)$$

Since π_T must include $\vec{C}_\Upsilon$ and must be a NIZK proof, it requires 33 group elements. Specifically, Equation (8) requires to prove $e(\Upsilon, T_3) = e(\mathcal{X}_{T_1}, \Gamma_2)$ and $e(\mathcal{X}_{T_1}, g) = e(T_1, g)$ whereas (9) requires to prove $e(T_2, \mathcal{X}_g) = e(\Gamma_1, \Upsilon)$ and $e(\mathcal{X}_g, g) = e(g, g)$ using an auxiliary variables $\mathcal{X}_g = g$ and $\mathcal{X}_{T_1} = T_1$.

5. For $i = 1, 2$, generate NIZK proofs $\pi_{\textit{eq-key},i}$ that $\vec{C}_{\Gamma_i}$ (which are part of com_{pk}) and ψ_{K_i} are encryptions of the same Γ_i. If $\psi_{\mathsf{K}_i} = (V_{i,0}, V_{i,1}, V_{i,2}, V_{i,3}, V_{i,4})$ comprises

$$(V_{i,0}, V_{i,1}, V_{i,2}) = \left(\Gamma_i \cdot g^{z_{i,1}+z_{i,2}}, Y_1^{z_{i,1}}, Y_2^{z_{i,2}}\right)$$

and $\vec{C}_{\Gamma_i}$ is parsed as $(c_{\Gamma_{i1}}, c_{\Gamma_{i2}}, c_{\Gamma_{i3}}) = \left(g_1^{\rho_{i1}} \cdot f_{3,1}^{\rho_{i3}},\ g_2^{\rho_{i2}} \cdot f_{3,2}^{\rho_{i3}},\ \Gamma_i \cdot g^{\rho_{i1}+\rho_{i2}} \cdot f_{3,3}^{\rho_{i3}}\right)$, where $z_{i,1}, z_{i,2} \in coins_\psi$, $\rho_{i1}, \rho_{i2}, \rho_{i3} \in \mathbb{Z}_p$ and $\vec{f} = (f_{3,1}, f_{3,2}, f_{3,3})$, this amounts to prove knowledge of values $z_{i,1}, z_{i,2}, \rho_{i1}, \rho_{i2}, \rho_{i3} \in \mathbb{Z}_p$ such that

$$\Big(\frac{V_{i,1}}{c_{\Gamma_{i1}}}, \frac{V_{i,2}}{c_{\Gamma_{i2}}}, \frac{V_{i,0}}{c_{\Gamma_{i3}}}\Big) = \big(Y_1^{z_{i,1}} \cdot g_1^{-\rho_{i1}} \cdot f_{3,1}^{-\rho_{i3}}, Y_2^{z_{i,2}} \cdot g_2^{-\rho_{i2}} \cdot f_{3,2}^{-\rho_{i3}}, g^{z_{i,1}+z_{i,2}-\rho_{i1}-\rho_{i2}} \cdot f_{3,3}^{-\rho_{i3}}\big) \ .$$

Committing to exponents $z_{i,1}, z_{i,2}, \rho_{i1}, \rho_{i2}, \rho_{i3}$ introduces 30 group elements whereas the above relations only require two elements each. Together with their corresponding commitments to $\{z_{i,1}, z_{i,2}, \rho_{i1}, \rho_{i2}, \rho_{i3}\}_{i=1,2}$, the proof element $\pi_{eq\text{-}key,i}$ incurs 42 elements.

6. Generate a NIZK proof $\pi_{\mathcal{R}}$ that the ciphertext ψ_{LY} encrypts a group element $M \in \mathbb{G}$ such that $((A,B),M) \in \mathcal{R}$. To this end, generate a commitment

$$com_M = (c_{M,1}, c_{M,2}, c_{M,3}) = \big(g_1^{\rho_1} \cdot f_{3,1}^{\rho_3}, g_2^{\rho_2} \cdot f_{3,2}^{\rho_3}, M \cdot g^{\rho_1+\rho_2} \cdot f_{3,3}^{\rho_3}\big)$$

and prove that the underlying M is the same as the one for which $C_0 = M \cdot X_1^{\theta_1} \cdot X_2^{\theta_2}$ in ψ_{LY}. In other words, prove knowledge of exponents $\theta_1, \theta_2, \rho_1, \rho_2, \rho_3$ such that

$$\Big(C_1, C_2, \frac{C_1}{c_{M,1}}, \frac{C_2}{c_{M,2}}, \frac{C_0}{c_{M,3}}\Big) = \Big(g_1^{\theta},\ g_2^{\theta},\ g_1^{\theta_1-\rho_1} \cdot f_{3,1}^{-\rho_3},\ g_2^{\theta_2-\rho_2} \cdot f_{3,2}^{-\rho_3},\ g^{-\rho_1-\rho_2} \cdot f_{3,3}^{-\rho_3} \cdot X_1^{\theta_1} \cdot X_2^{\theta_2}\Big) \ . \quad (10)$$

Committing to $\theta_1, \theta_2, \rho_1, \rho_2, \rho_3$ takes 15 elements. Proving the first four relations of (10) requires 8 elements whereas the last one is quadratic and its proof is 9 elements. Proving the linear pairing-product relation $e(g,M) = e(A,B)$ in NIZK[5] demands 9 elements. Since $\pi_{\mathcal{R}}$ includes com_M, it entails a total of 44 elements.

The entire proof $\pi_\psi = com_{\mathsf{cert}_{\mathsf{pk}}} \| com_{\mathsf{pk}} \| \pi_{\mathsf{cert}_{\mathsf{pk}}} \| \pi_T \| \pi_{eq\text{-}key,1} \| \pi_{eq\text{-}key,2} \| \pi_{\mathcal{R}}$ takes 150 elements.

$\mathcal{V}(\mathsf{param}, \psi, L, \pi_\psi, \mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}})$: Parse $\mathsf{pk}_{\mathsf{GM}}$, $\mathsf{pk}_{\mathsf{OA}}$, pk, ψ and π_ψ as above. Return 1 if and only if the conditions below are all satisfied.

1. $\mathcal{V}(\mathsf{VK}, \sigma, ((T_1,T_2,T_3,T_4) \| \psi_{\mathsf{LY}} \| \psi_{\mathsf{K}_1} \| \psi_{\mathsf{K}_2} \| L)) = 1$.
2. The equality $e(T_1, \Lambda_0^{\mathsf{VK}} \cdot \Lambda_1) = e(g, T_4)$ is satisfied and ψ_{LY} is a valid LY ciphertext.
3. All proofs verify and if $\{\psi_{\mathsf{K}_i}\}_{i=1}^2$ are valid Kiltz encryptions w.r.t. VK.

$\mathsf{DEC}(\mathsf{sk}, \psi, L)$: Parse ψ as $\mathsf{VK} \| (T_1,T_2,T_3,T_4) \| \psi_{\mathsf{LY}} \| \psi_{\mathsf{K}_1} \| \psi_{\mathsf{K}_2} \| \sigma$. Return $\perp$ in the event that either: (i) $\mathcal{V}(\mathsf{VK}, \sigma, ((T_1,T_2,T_3,T_4) \| \psi_{\mathsf{LY}} \| \psi_{\mathsf{K}_1} \| \psi_{\mathsf{K}_2} \| L)) = 0$; (ii) $e(T_1, \Lambda_0^{\mathsf{VK}} \cdot \Lambda_1) \neq e(g, T_4)$ or ψ_{LY} and $\{\psi_{\mathsf{K}_i}\}_{i=1,2}$ are not all valid ciphertexts. Otherwise, use sk to decrypt (ψ_{LY}, L).

$\mathsf{REVEAL}(\mathsf{transcript}_i, \mathsf{sk}_{\mathsf{OA}})$: Parse $\mathsf{transcript}_i$ as

$$\big((X_{i,1}, X_{i,2}, \Gamma_{i,1}, \Gamma_{i,2}), (\Phi_{venc,i}, \vec{C}_{W_{i,1}}, \vec{C}_{W_{i,2}}, \pi_{venc,i}), \mathsf{cert}_{\mathsf{pk},i}\big).$$

Parse $\Phi_{venc,i}$ as a BBS ciphertext $(\Phi_{i,0}, \Phi_{i,1}, \Phi_{i,2}) \in \mathbb{G}^3$ and verify that $(\vec{C}_{W_{i,1}}, \vec{C}_{W_{i,2}}, \pi_{venc,i})$ form a valid proof for the statements (3)-(5). If not, return $\perp$. Otherwise, use $\mathsf{sk}_{\mathsf{OA}} = (y_1, y_2, y_3, y_4)$ to compute $\Gamma_{i,0} = \Phi_{i,0} \cdot \Phi_{i,1}^{-1/y_1} \cdot \Phi_{i,2}^{-1/y_2}$. Return the resulting plaintext $\mathsf{trace}_i = \Gamma_{i,0} \in \mathbb{G}$ which can serve as a tracing trapdoor for user i as it is necessarily of the form $\Gamma_{i,0} = \Gamma_{i,2}^{\log_g(\Gamma_{i,1})}$.

$\mathsf{TRACE}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, \mathsf{trace}_i)$: Parse the ciphertext ψ as $\mathsf{VK} \| (T_1,T_2,T_3,T_4) \| \psi_{\mathsf{LY}} \| \psi_{\mathsf{K}_1} \| \psi_{\mathsf{K}_2} \| \sigma$ and the tracing trapdoor trace_i as a group element $\Gamma_{i,0} \in \mathbb{G}$. If the equality $e(T_1, \Gamma_{i,0}) = e(T_2, T_3)$ holds, it returns 1 (meaning that ψ is indeed intended for user i). Otherwise, it outputs 0.

[5] It requires to introduce an auxiliary variable $\mathcal{A}$ and prove that $e(g,M) = e(\mathcal{A}, B)$ and $\mathcal{A} = A$, for variables $M, \mathcal{A}$ and constants g, A, B. The two proofs take 3 elements each and 3 elements are needed to commit to $\mathcal{A}$.

$\mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}}, \psi, L):$ Parse ψ as $\mathsf{VK}\|(T_1,T_2,T_3,T_4)\|\psi_{\mathsf{LY}}\|\psi_{\mathsf{K}_1}\|\psi_{\mathsf{K}_2}\|\sigma$. Return $\perp$ if $\{\psi_{\mathsf{K}_i}\}_{i=1}^{2}$ are not both valid ciphertexts w.r.t. VK or if $\mathcal{V}(\mathsf{VK}, \sigma, ((T_1,T_2,T_3,T_4)\|\psi_{\mathsf{LY}}\|\psi_{\mathsf{K}_1}\|\psi_{\mathsf{K}_2}\|L)) = 0$. Otherwise, decrypt $\{\psi_{\mathsf{K}_i}\}_{i=1,2}$ to obtain group elements $\Gamma_1, \Gamma_2 \in \mathbb{G}$ and look up $\mathsf{database}$ to find a record $\mathsf{transcript}_i$ containing a key $\mathsf{pk}_i = (X_{i,1}, X_{i,2}, \Gamma_{i,1}, \Gamma_{i,2})$ such that $(\Gamma_{i,1}, \Gamma_{i,2}) = (\Gamma_1, \Gamma_2)$ (note that, unless $\mathsf{database}$ is ill-formed, such a record is unique if it exists). If such a record is found, output the matching i. Otherwise, output $\perp$.

$\mathsf{CLAIM/DISCLAIM}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, L, \mathsf{sk}):$ Parse ψ as $\mathsf{VK}\|(T_1,T_2,T_3,T_4)\|\psi_{\mathsf{LY}}\|\psi_{\mathsf{K}_1}\|\psi_{\mathsf{K}_2}\|\sigma$ and the private key as $\mathsf{sk} = (x_1, x_2, z, \gamma_1, \gamma_2)$. To generate a claim/disclaimer τ for the ciphertext ψ, first verify that $e(T_1, \Lambda_0^{\mathsf{VK}} \cdot \Lambda_1) = e(g, T_4)$ and that σ is a valid one-time signature. If these conditions, do not hold, return $\perp$. Otherwise, compute $T_{\delta,1} = T_1^{\gamma_1} = \Gamma_1^{\delta}$, where $\delta = \log_g(T_1)$. Then, compute a collision-resistant hash $\mathsf{v} = \mathsf{CMhash}(hk, (\psi, L, \mathsf{pk}), s_{hash}) \in \{0,1\}^{\ell}$, where $s_{hash} \stackrel{R}{\leftarrow} \mathcal{R}_{hash}$. Then, parse v as $\mathsf{v}[1] \ldots \mathsf{v}[\ell] \in \{0,1\}^{\ell}$ and assemble the vector $\vec{h}_{\mathsf{v}} = \vec{h}_0 \odot \bigodot_{i=1}^{\ell} \vec{h}_i^{\mathsf{v}[i]}$. Using $(\vec{g_1}, \vec{g_2}, \vec{h}_{\mathsf{v}})$ as a Groth-Sahai CRS, generate a commitment $\vec{C}_{\Gamma_{-1}}$ to $\Gamma_{-1} = g^{1/\gamma_1}$ and a NIZK proof that Γ_{-1} satisfies $e(T_{\delta,1}, \Gamma_{-1}) = e(T_1, g)$. To this end, generate a commitment $\vec{C}_{\mathcal{X}_\tau}$ to the auxiliary variable $\mathcal{X}_\tau = g$ and non-interactive proofs $\pi_{\tau,1}$, $\pi_{\tau,2}$ for the equations

$$e(T_{\delta,1}, \Gamma_{-1}) = e(T_1, \mathcal{X}_\tau), \qquad e(g, \mathcal{X}_\tau) = e(g, g) \ . \tag{11}$$

The claim/disclaimer τ consists of

$$\tau = \big(T_{\delta,1}, \vec{C}_{\Gamma_{-1}}, \vec{C}_{\mathcal{X}_\tau}, \pi_{\tau,1}, \pi_{\tau,2}, s_{hash}\big) \in \mathbb{G}^{14} \ . \tag{12}$$

$\mathsf{CLAIM\text{-}VERIFY}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, L, \mathsf{pk}, \tau):$ Parse ψ as $\mathsf{VK}\|(T_1,T_2,T_3,T_4)\|\psi_{\mathsf{LY}}\|\psi_{\mathsf{K}_1}\|\psi_{\mathsf{K}_2}\|\sigma$ and the public key pk as $(X_1, X_2, \Gamma_1, \Gamma_2)$. Parse τ as per (12). Return 1 if and only if the relations

$$e(T_{\delta,1}, \Gamma_2) = e(T_2, T_3), \qquad e(T_1, \Gamma_1) = e(g, T_{\delta,1}) \tag{13}$$

hold and $\pi_{\tau,1}, \pi_{\tau,2}$ are valid proofs for the relations (11) with respect to the CRS $(\vec{g_1}, \vec{g_2}, \vec{h}_{\mathsf{v}})$, where $\vec{h}_{\mathsf{v}} = \vec{h}_0 \odot \bigodot_{i=1}^{\ell} \vec{h}_i^{\mathsf{v}[i]}$ and $\mathsf{v} = \mathsf{CMhash}(hk, (\psi, L, \mathsf{pk}), s_{hash}) \in \{0,1\}^{\ell}$.

$\mathsf{DISCLAIM\text{-}VERIFY}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \psi, L, \mathsf{pk}, \tau):$ Parse ψ as $\mathsf{VK}\|(T_1,T_2,T_3,T_4)\|\psi_{\mathsf{LY}}\|\psi_{\mathsf{K}_1}\|\psi_{\mathsf{K}_2}\|\sigma$ and the public key pk as $(X_1, X_2, \Gamma_1, \Gamma_2)$. Parse τ as per (12). Return 1 if and only if it holds that

$$e(T_{\delta,1}, \Gamma_2) \neq e(T_2, T_3), \qquad e(T_1, \Gamma_1) = e(g, T_{\delta,1}) \tag{14}$$

and $\pi_{\tau,1}, \pi_{\tau,2}$ are valid proofs for the relations (11) and the Groth-Sahai CRS $(\vec{g_1}, \vec{g_2}, \vec{h}_{\mathsf{v}})$, where $\vec{h}_{\mathsf{v}} = \vec{h}_0 \odot \bigodot_{i=1}^{\ell} \vec{h}_i^{\mathsf{v}[i]}$ and $\mathsf{v} = \mathsf{CMhash}(hk, (\psi, L, \mathsf{pk}), s_{hash}) \in \{0,1\}^{\ell}$.

4.3 Analysis

From an efficiency point of view, the length of ciphertexts is about 2.18 kB in an implementation using symmetric pairings with a 512-bit representation for each group element (at the 128-bit security level), which is more compact than in the Paillier-based system of Kiayias *et al.* [30] where ciphertexts already take 2.5 kB using 1024-bit moduli (and thus at the 80-bit security level). Moreover, our proofs only require 9.38 kB (against roughly 32 kB for the same security in [15]), which is significantly cheaper than in the original group encryption scheme [30], where interactive proofs reach a communication cost of 70 kB to achieve a 2^{-50} knowledge error.

The correctness of the scheme stems from that of Groth-Sahai proofs. From a security point of view, we prove the security properties under the q-SFP, D3DH and DLIN assumptions and also require the one-time signatures to be strongly unforgeable [4]. All proofs are given in Appendix B.

References

1. M. Abe, K. Haralambiev, M. Ohkubo. Signing on Elements in Bilinear Groups for Modular Protocol Design. Cryptology ePrint Archive: Report 2010/133, 2010.
2. M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. Ohkubo. Structure-Preserving Signatures and Commitments to Group Elements. In *Crypto'10*, *LNCS* 6223, pp. 209–236, 2010.
3. M. Abe, J. Groth, K. Haralambiev, M. Ohkubo. Optimal Structure-Preserving Signatures in Asymmetric Bilinear Groups. In *Crypto'11*, *LNCS* 6841, pp. 649–666, 2011.
4. J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In *Eurocrypt'02*, LNCS 2332, pages 83–107, 2002.
5. F. Bao, R. Deng, H. Zhu. Variations of Diffie-Hellman Problem. In *ICICS'03*, *LNCS* 2836, pp. 301–312, Springer, 2003.
6. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Asiacrypt'01*, LNCS 2248, pages 566–582, 2001.
7. M. Bellare, T. Ristenpart. Simulation without the Artificial Abort: Simplified Proof and Improved Concrete Security for Waters' IBE Scheme. In *Eurocrypt'09*, *LNCS* 5479, pp. 407–424, 2009.
8. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS'93*, pages 62–73, 1993.
9. V. Benjumea, S. G. Choi, J. Lopez, M. Yung. Fair traceable multi-group signatures. In *Financial Cryptography 2008*, volume 5143 of *Lecture Notes in Computer Science*, pages 231–246. Springer, 2008.
10. D. Boneh and X. Boyen. Short signatures without random oracles. In *Eurocrypt'04*, LNCS 3027, pages 56–73, 2004.
11. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Crypto'04*, LNCS 3152, pages 41–55, 2004.
12. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. *SIAM J. of Computing*, 32(3):586–615, 2003. Extended abstract in Crypto'01, LNCS 2139, pages 213–229, 2001.
13. J. Camenisch and A. Lysyanskaya. A Signature Scheme with Efficient Protocols. In *SCN'02*, LNCS 2576, pages 268–289, 2003.
14. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *Journal of the ACM*, 51(4):557–594, 2004.
15. J. Cathalo, B. Libert, M. Yung. Group Encryption: Non-Interactive Realization in the Standard Model. In *Asiacrypt'09*, *LNCS* 5912, pp. 179–196, 2009.
16. D. Chaum and E. van Heyst. Group signatures. In *Eurocrypt'91*, LNCS 547, pages 257–265, 1991.
17. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Crypto'98*, LNCS 1462, pages 13–25, 1998.
18. L. El Aimani, M. Joye. Toward Practical Group Encryption. In *ACNS 2013*, LNCS series, to appear, 2013. Available as Cryptology ePrint Archive: Report 2012/155, 2012.
19. A. Escala, G. Herold, E. Kiltz, C. Ràfols, J. Villar. An Algebraic Framework for Diffie-Hellman Assumptions. In *Crypto 2013*, *LNCS* 8043, pp. 129–147, 2013.
20. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Crypto'86*, LNCS 263, pages 186–194, 1986.
21. S. Goldwasser and Y. Tauman-Kalai. On the (In)security of the Fiat-Shamir Paradigm In *FOCS'03*, pages 102–115, 2003.
22. J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *Asiacrypt'06*, LNCS 4284, pages 444–459, 2006.
23. J. Groth. Fully anonymous group signatures without random oracles. In *Asiacrypt'07*, LNCS 4833, pages 164–180, 2007.
24. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Eurocrypt'08*, LNCS 4965, pages 415–432, 2008.
25. S. Halevi. A Sufficient Condition for Key-Privacy. Cryptology ePrint Archive: Report 2005/005, 2005.
26. M. Izabachène, D. Pointcheval, D. Vergnaud. Mediated Traceable Anonymous Encryption. In *Latincrypt'08*, LNCS 6212, pages 40–60, 2010.
27. C. Jutla, A. Roy. Relatively-Sound NIZKs and Password-Based Key-Exchange. In *PKC'12*, *LNCS* series, pp. 485-503, 2012.
28. C. Jutla, A. Roy. Shorter Quasi-Adaptive NIZK Proofs for Linear Subspaces. In *Asiacrypt'13*, *LNCS* series, 2013. Cryptology ePrint Archive: Report 2013/109, 2013.
29. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable signatures. In *Eurocrypt 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 571–589. Springer, 2004.
30. A. Kiayias, Y. Tsiounis, and M. Yung. Group encryption. In *Asiacrypt'07*, LNCS 4833, pages 181–199, 2007.
31. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC'06*, LNCS 3876, pages 581–600, 2006.
32. H. Krawczyk and T. Rabin. Chameleon signatures. In *NDSS'00*, 2000.

33. B. Libert and M. Yung. Efficient Traceable Signatures in the Standard Model. In *Pairing-Based Cryptography 2009 (Pairing'09)*, volume 5671 of *Lecture Notes in Computer Science*, pages 187–205. Springer, 2009.
34. B. Libert, M. Yung. Non-Interactive CCA2-Secure Threshold Cryptosystems with Adaptive Security: New Framework and Constructions. In *TCC 2012*, LNCS 7194, pp. 75–93, Springer, 2012.
35. T. Malkin, I. Teranishi, Y. Vahlis, M. Yung. Signatures resilient to continual leakage on memory and computation. In *TCC'11*, *LNCS* 6597, pp. 89–106, 2011.
36. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt'99*, *LNCS* 1592, pages 223–238, 1999.
37. B. Qin, Q. Wu, W. Susilo, Y. Mu, Y. Wang. Publicly Verifiable Privacy-Preserving Group Decryption. In *Inscrypt'08*, *LNCS* 5487, pages 72–83, 2008.
38. M. Trolin, D. Wiström. Hierarchical Group Signatures. In *ICALP'05*, *LNCS* 3580, pp. 446–458, 2005.
39. B. Waters. Efficient Identity-Based Encryption Without Random Oracles. In *Eurocrypt'05*, *LNCS* 3494, 2005.

A Building Blocks

A.1 Structure-Preserving Signatures

In cryptographic protocols involving Groth-Sahai proofs, it is often useful to sign elements of an abelian group $\mathbb{G}$ over which a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is efficiently computable. Importantly, it should be possible to sign group elements *without* knowing their discrete logarithms and *without* hashing them to the message space of an ordinary signature scheme: indeed, one should preserve the feasibility of efficiently proving statements about a committed message-signature pair.

The first signature scheme with this property was suggested by Groth [22]. A much more efficient solution was given by Abe, Haralambiev and Ohkubo [1,2] (AHO) who introduced the *structure-preserving* terminology. The Abe *et al.* construction is recalled hereunder. The description assumes public parameters $\mathsf{pp} = \big((\mathbb{G}, \mathbb{G}_T), g\big)$ consisting of bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$, where $\lambda \in \mathbb{N}$ and a generator $g \in \mathbb{G}$.

$\mathsf{Keygen}(\mathsf{pp}, n)$: Given an upper bound $n \in \mathbb{N}$ on the number of group elements per signed message, choose generators $G_r, H_u \overset{R}{\leftarrow} \mathbb{G}$. Pick $\gamma_z, \delta_z \overset{R}{\leftarrow} \mathbb{Z}_p$ and $\gamma_i, \delta_i \overset{R}{\leftarrow} \mathbb{Z}_p$, for $i = 1$ to n. Then, compute $G_z = G_r^{\gamma_z}$, $H_z = H_u^{\delta_z}$ and $G_i = G_r^{\gamma_i}$, $H_i = H_u^{\delta_i}$ for each $i \in \{1, \ldots, n\}$. Finally, choose $\alpha_a, \alpha_b \overset{R}{\leftarrow} \mathbb{Z}_p$ and define $\Omega_a = e(G_r, g^{\alpha_a})$ and $\Omega_b = e(H_u, g^{\alpha_b})$. The public key is defined to be

$$\mathsf{pk} = \big(G_r, H_u, G_z, H_z, \{G_i, H_i\}_{i=1}^n, \Omega_a, \Omega_b\big) \in \mathbb{G}^{2n+4} \times \mathbb{G}_T^2$$

while the private key is $\mathsf{sk} = \big(\alpha_a, \alpha_b, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^n\big)$.

$\mathsf{Sign}(\mathsf{sk}, (M_1, \ldots, M_n))$: To sign a vector $(M_1, \ldots, M_n) \in \mathbb{G}^n$ using sk, choose $\zeta, \rho_a, \rho_b, \omega_a, \omega_b \overset{R}{\leftarrow} \mathbb{Z}_p$ and compute $Z = g^\zeta$ as well as

$$R = g^{\rho_a - \gamma_z \zeta} \cdot \prod_{i=1}^n M_i^{-\gamma_i}, \qquad S = G_r^{\omega_a}, \qquad T = g^{(\alpha_a - \rho_a)/\omega_a},$$
$$U = g^{\rho_b - \delta_z \zeta} \cdot \prod_{i=1}^n M_i^{-\delta_i}, \qquad V = H_u^{\omega_b}, \qquad W = g^{(\alpha_b - \rho_b)/\omega_b}.$$

The signature consists of $\sigma = (Z, R, S, T, U, V, W) \in \mathbb{G}^7$.

$\mathsf{Verify}(\mathsf{pk}, \sigma, (M_1, \ldots, M_n))$: Given $\sigma = (Z, R, S, T, U, V, W)$, return 1 iff these equalities hold:

$$\Omega_a = e(G_z, Z) \cdot e(G_r, R) \cdot e(S, T) \cdot \prod_{i=1}^n e(G_i, M_i),$$
$$\Omega_b = e(H_z, Z) \cdot e(H_u, U) \cdot e(V, W) \cdot \prod_{i=1}^n e(H_i, M_i).$$

The scheme was proved [1,2] existentially unforgeable under chosen-message attacks under the q-SFP assumption, where q is the number of signing queries.

As proved by Abe *et al.* [1,2], signature components $\{\theta_i\}_{i=2}^{7}$ can be publicly randomized to obtain a different signature $(Z', R', S', T', U', V', W') \leftarrow \mathsf{ReRand}(\mathsf{pk}, \sigma)$ on $(M_1, \ldots, M_n)$. After randomization, we have $Z' = Z$ while (R', S', T', U', V', W') are uniformly distributed among the values such that $e(G_r, R') \cdot e(S', T') = e(G_r, R) \cdot e(S, T)$ and $e(H_u, U') \cdot e(V', W') = e(H_u, U) \cdot e(V, W)$. This re-randomization is performed by choosing $\varrho_2, \varrho_5, \mu, \nu \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and computing

$$\begin{aligned} R' &= R \cdot T^{\varrho_2}, & S' &= (S \cdot G_r^{-\varrho_2})^{1/\mu}, & T' &= T^{\mu}, \\ U' &= U \cdot W^{\varrho_5}, & V' &= (V \cdot H_u^{-\varrho_5})^{1/\nu}, & W' &= W^{\nu} \end{aligned} .$$

As a result, the group elements (S, T, V, W) are statistically independent of the message $(M_1, \ldots, M_n)$ and the rest of the signature. This implies that, in anonymity-related protocols, re-randomized group elements (S', T', V', W') can be safely given out as long as $(M_1, \ldots, M_n)$ and (Z', R', U') are both given in committed form.

A.2 Public-Key Encryption Schemes Based on the Linear Problem

We need cryptosystems based on the DLIN assumption. The first one is a variant of the Cramer-Shoup cryptosystem [17] suggested in [34]. We also use Kiltz's tag-based encryption (TBE) scheme [31] since it is the most efficient DLIN-based system with ciphertexts of publicly verifiable validity.

A PUBLICLY VERIFIABLE VARIANT OF CRAMER-SHOUP. Libert and Yung [34] proposed a variant of the Cramer-Shoup cryptosystem where ciphertexts contain a publicly verifiable proof of their validity. We will use it because it can combine the properties of anonymity and public ciphertext verifiability. Namely, the validity of ciphertexts is publicly verifiable and, at the same time, ciphertexts do not betray the public key under which they were encrypted. Although these two properties appear antagonistic, we show that they can co-exist here if certain public components are shared by all users.

$\mathsf{Keygen}(\lambda)$:

1. Choose a group $\mathbb{G}$ of prime order $p > 2^{\lambda}$, $g, g_1, g_2 \stackrel{R}{\leftarrow} \mathbb{G}$, $x_1, x_2, z \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and sets $X_1 = g_1^{x_1} g^z$, $X_2 = g_2^{x_2} g^z$. Define the vectors $\vec{g}_1 = (g_1, 1, g)$ and $\vec{g}_2 = (1, g_2, g)$.
2. Pick $\xi_1, \xi_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and define $\vec{g}_3 = \vec{g}_1^{\,\xi_1} \cdot \vec{g}_2^{\,\xi_2}$.
3. Choose a strongly unforgeable one-time signature $\Sigma = (\mathcal{G}, \mathcal{S}, \mathcal{V})$.
4. Define private key as $\mathsf{sk} = (x_1, x_2, z) \in \mathbb{Z}_p^3$ and the public key is $\mathsf{pk} = (\vec{g}_1, \vec{g}_2, \vec{g}_3, X_1, X_2, \Sigma)$.

$\mathsf{Encrypt}(\mathsf{pk}, M)$: To encrypt $M \in \mathbb{G}$, generate a one-time signature key pair $(\mathsf{SK}, \mathsf{VK}) \leftarrow \mathcal{G}(\lambda)$ and do the following.

1. Choose $\theta_1, \theta_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and compute $(C_0, C_1, C_2, C_3) = (M \cdot X_1^{\theta_1} \cdot X_2^{\theta_2}, g_1^{\theta_1}, g_2^{\theta_2}, g^{\theta_1 + \theta_2})$.
2. Construct a vector $\vec{g}_{\mathsf{VK}} = \vec{g}_3 \cdot (1, 1, g)^{\mathsf{VK}}$ and use $\mathbf{g}_{\mathsf{VK}} = (\vec{g}_1, \vec{g}_2, \vec{g}_{\mathsf{VK}})$ as a Groth-Sahai CRS to generate a NIZK proof that $(g, g_1, g_2, C_1, C_2, C_3)$ form a linear tuple. More precisely, generate commitments $\vec{C}_{\theta_1}, \vec{C}_{\theta_2}$ to encryption exponents $\theta_1, \theta_2 \in \mathbb{Z}_p$ (in other words, compute $\vec{C}_{\theta_i} = \vec{g}_{\mathsf{VK}}^{\,\theta_i} \cdot \vec{g}_1^{\,r_i} \cdot \vec{g}_2^{\,s_i}$ with $r_i, s_i \stackrel{R}{\leftarrow} \mathbb{Z}_p$ for each $i \in \{1, 2\}$) and a proof π_{LIN} that they satisfy

$$C_1 = g_1^{\theta_1}, \qquad C_2 = g_2^{\theta_2}, \qquad C_3 = g^{\theta_1 + \theta_2} . \qquad (15)$$

The whole proof for (7) consists of $\vec{C}_{\theta_1}$, $\vec{C}_{\theta_2}$ and π_{LIN} is obtained as

$$\pi_{\mathrm{LIN}} = (\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6) = (g_1^{r_1}, g_1^{s_1}, g_2^{r_2}, g_2^{s_2}, g^{r_1 + r_2}, g^{s_1 + s_2}) .$$

3. Output the ciphertext $C = (\mathsf{VK}, C_0, C_1, C_2, C_3, \vec{C}_{\theta_1}, \vec{C}_{\theta_2}, \pi_{\mathrm{LIN}}, \sigma)$ where σ is computed as $\sigma = \mathcal{S}(\mathsf{SK}, (C_0, C_1, C_2, C_3, \vec{C}_{\theta_1}, \vec{C}_{\theta_2}, \pi_{\mathrm{LIN}}))$.

$\mathsf{Decrypt}(\mathsf{sk}, C)$: Parse C as $(\mathsf{VK}, C_0, C_1, C_2, C_3, \vec{C}_{\theta_1}, \vec{C}_{\theta_2}, \pi_{\mathrm{LIN}}, \sigma)$. Return 0 if σ is an invalid one-time signature on $(C_0, C_1, C_2, C_3, \vec{C}_{\theta_1}, \vec{C}_{\theta_2}, \pi_{\mathrm{LIN}})$ w.r.t. VK or if π_{LIN} is not a valid proof for Eqs (15) and the CRS $\mathbf{g}_{\mathsf{VK}} = (\vec{g_1}, \vec{g_2}, \vec{g_3} \cdot (1, 1, g)^{\mathsf{VK}})$. This entails testing the equalities

$$E(g_1, \vec{C}_{\theta_1}) = E(C_1, \vec{g}_{\mathsf{VK}}) \cdot E(\pi_1, \vec{g_1}) \cdot E(\pi_2, \vec{g_2}) \,,$$
$$E(g_2, \vec{C}_{\theta_2}) = E(C_2, \vec{g}_{\mathsf{VK}}) \cdot E(\pi_3, \vec{g_1}) \cdot E(\pi_4, \vec{g_2}) \,,$$
$$E(g, \vec{C}_{\theta_1} \cdot \vec{C}_{\theta_2}) = E(C_3, \vec{g}_{\mathsf{VK}}) \cdot E(\pi_5, \vec{g_1}) \cdot E(\pi_6, \vec{g_2}) \ .$$

Otherwise, parse sk as $(x_1, x_2, z) \in \mathbb{Z}_p^3$. Compute and return $M = C_0 \cdot C_1^{-x_1} \cdot C_2^{-x_2} \cdot C_3^{-z}$.

In our construction of traceable group encryption, we use the observation that the public key components $(\vec{g_1}, \vec{g_2}, \vec{g_3})$ can be shared by many users: each individual public key thus only consists of (X_1, X_2). In this case, the scheme can combine the features of public verifiability and receiver anonymity.

Kiltz's tag-based encryption scheme. In [31], Kiltz described a CCA2-secure public-key encryption scheme based on the DLIN assumption.

$\mathsf{Keygen}(\lambda)$: Choose a group $\mathbb{G}$ of prime order $p > 2^\lambda$ with a generator $g \stackrel{R}{\leftarrow} \mathbb{G}$. Pick random exponents $x_1, x_2, x_3, x_4 \stackrel{R}{\leftarrow} \mathbb{Z}_p$. The private key is $\mathsf{sk} = (y_1, y_2, y_3, y_4) \in \mathbb{Z}_p^4$ and the public key is $\mathsf{pk} = (g, Y_1, Y_2, Y_3, Y_4) = (g, g^{y_1}, g^{y_2}, g^{y_3}, g^{y_4})$.

$\mathsf{Encrypt}(\mathsf{pk}, M)$: To encrypt $M \in \mathbb{G}$, generate a one-time signature key pair $(\mathsf{SK}, \mathsf{VK}) \leftarrow \mathcal{G}(\lambda)$, pick $z_1, z_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and compute

$$C = (\mathsf{VK}, V_0, V_1, V_2, V_3, V_4, \sigma) = \big(\mathsf{VK}, M \cdot g^{z_1+z_2}, Y_1^{z_1}, Y_2^{z_2}, (g^{\mathsf{VK}} \cdot Y_3)^{z_1}, (g^{\mathsf{VK}} \cdot Y_4)^{z_2}, \sigma\big) \,,$$

where σ is a signature of $(V_0, V_1, V_2, V_3, V_4)$.

$\mathsf{Decrypt}(\mathsf{sk}, C)$: The receiver first checks that σ is a valid one-time signature on $(V_0, V_1, V_2, V_3, V_4)$ and returns $\perp$ if it is not. Otherwise, it checks that $V_3 = V_1^{(\mathsf{VK}+y_3)/y_1}$ and $V_4 = V_2^{(\mathsf{VK}+y_4)/y_1}$. If so, it outputs the plaintext $M = V_0/(V_1^{1/y_1} V_2^{1/y_2})$.

B Proofs of Security

B.1 Message Privacy

Theorem 1. *The scheme satisfies message security assuming that Σ is a strongly unforgeable one-time signature and that the DLIN assumption holds in $\mathbb{G}$.*

Proof. We use a sequence of games. The first one mirrors the experiment of definition 4 where the challenger's bit b is 1 and the adversary obtains a encryption of the witness $M^\star$ and real proofs when invoking the $\mathsf{PROVE}(.)$ oracle. In the last game, the adversary $\mathcal{A}$ obtains an encryption of a random plaintext and proofs are simulated using the trapdoor associated with the fake CRS. In Game_i, W_i denotes the event that $\mathcal{A}$ outputs $b' = 1$.

Game_1: the challenger $\mathcal{B}$ provides $\mathcal{A}$ with common public parameters param that include a real CRS $\mathbf{g}$ containing $(\vec{g_1}, \vec{g_2}, \vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2})$ and $\vec{f} = \vec{g_1}^{\eta_1} \odot \vec{g_2}^{\eta_2}$, with $\xi_1, \xi_2, \eta_1, \eta_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p$. The adversary generates public keys $\mathsf{pk}_{\mathsf{OA}}$ and $\mathsf{pk}_{\mathsf{GM}}$ on its own. The challenger and $\mathcal{A}$ run an execution of JOIN where $\mathcal{A}$ certifies the public key $\mathsf{pk} = (X_1, X_2, \Gamma_1, \Gamma_2)$ of a honest receiver chosen by $\mathcal{B}$. Then, $\mathcal{A}$

makes a number of decryption queries that $\mathcal{B}$ handles using the private key $\mathsf{sk} = (x_1, x_2, z, \gamma_1, \gamma_2)$ associated with pk. At some point, $\mathcal{A}$ outputs $((A, B), M^\star, L, \mathsf{pk}_{\mathcal{R}})$ such that $((A, B), M^\star) \in \mathcal{R}$ and obtains in return a TGE encryption $\psi^\star = \mathsf{VK}^\star||(T_1^\star, T_2^\star, T_3^\star, T_4^\star)||\psi^\star_{\mathsf{LY}}||\psi^\star_{\mathsf{K}_1}||\psi^\star_{\mathsf{K}_2}||\sigma^\star$ of M under the public key pk for the label L. Then, $\mathcal{A}$ obtains polynomially many proofs $\pi^\star_{\psi^\star}$ for $\psi^\star$ and makes new decryption queries under the obvious restrictions. Finally, $\mathcal{A}$ outputs b' and we call W_1 the event that $b' = 1$. W.l.o.g. we assume that $\mathsf{VK}^\star$ and its corresponding $\mathsf{SK}^\star$ are chosen by the challenger at the beginning of the game. In Game_1 and subsequent games, we denote by $\theta_1^\star = \log_{g_1}(C_1^\star)$ and $\theta_2^\star = \log_{g_2}(C_2^\star)$ the random encryption exponents used in the computation of $\psi^\star_{\mathsf{LY}}$.

Game_2: we modify the generation of public parameters param and choose the vector $\vec{g_3} \in \mathbb{G}^3$ as $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2} \odot (1, 1, g)^{-\mathsf{VK}^\star}$ (instead of $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$ as previously). In other words, $\vec{g_3}$ becomes a BBS encryption of $g^{-\mathsf{VK}^\star}$ instead of an encryption of $1_{\mathbb{G}}$. Consequently, $\mathsf{VK}^\star$ is no longer independent of $\mathcal{A}$'s view between the setup phase and the challenge phase. However, as far as the DLIN assumption holds in $\mathbb{G}$, this modification will have no noticeable impact on $\mathcal{A}$'s behavior and we have $|\Pr[W_2] - \Pr[W_1]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

Game_3: we modify again the generation of param. This time, the vector $\vec{f} \in \mathbb{G}^3$ is chosen as $\vec{f} = \vec{g_1}^{\eta_1} \odot \vec{g_2}^{\eta_2} \odot (1, 1, g)^{-1}$ (instead of $\vec{f} = \vec{g_1}^{\eta_1} \odot \vec{g_2}^{\eta_2}$). Under the DLIN assumption, this change can be made without $\mathcal{A}$ noticing and we can write $|\Pr[W_3] - \Pr[W_2]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

Game_4: we now modify the $\mathsf{DEC}(.)$ oracle. Namely, the challenger $\mathcal{B}$ rejects all ciphertexts of the form $\psi = \mathsf{VK}||(T_1, T_2, T_3, T_4)||\psi_{\mathsf{LY}}||\psi_{\mathsf{K}_1}||\psi_{\mathsf{K}_2}||\sigma$ such that $\mathsf{VK} = \mathsf{VK}^\star$. Let F_4 be the event that this modification leads $\mathcal{B}$ to reject a ciphertext that would not have been rejected in Game_3. A standard argument shows that $|\Pr[W_4] - \Pr[W_3]| \leq \Pr[F_3] \leq \mathbf{Adv}^{\text{suf-ots}}(\lambda)$.

Game_5: we change the generation of proofs $\pi^\star_{\psi^\star}$ and use the trapdoor (η_1, η_2) of the CRS $(\vec{g_1}, \vec{g_2}, \vec{f})$ instead of witnesses $M^\star$ and $coins_{\psi^\star} = \{\delta^\star, \varrho^\star, \{z^\star_{i,1}, z^\star_{i,2}\}_{i=1,2}, \theta_1^\star, \theta_2^\star\}$. More precisely, $\{\pi^\star_{\mathit{eq\text{-}key},i}\}_{i=1,2}$ (which demonstrate that $\{com_{\Gamma_i}\}_{i=1,2}$ and $\{\psi_{\mathsf{K}_i}\}_{i=1,2}$ hide the same elements $\{\Gamma_i\}_{i=1,2}$), as well as $\pi^\star_{\mathcal{R}}$ (*i.e.*, the proof that $\psi^\star_{\mathsf{LY}}$ and com_M conceal the same $M^\star$) are simulated without using encryption exponents $\{z^\star_{i,1}, z^\star_{i,2}\}_{i=1,2}$ and $\theta_1^\star, \theta_2^\star$ and commitments to the latter values are replaced by commitments to 0. The same goes for the proof π_T that (T_1, T_2, T_3) are well-formed at step 4 of the proving algorithm[6]. Also, the part of $\pi^\star_{\mathcal{R}}$ that proves relation $e(g, M^\star) = e(A, B)$ (and thus $((A, B), M^\star) \in \mathcal{R}$) is simulated in NIZK[7] by setting com_M as a commitment to $1_{\mathbb{G}}$. The trapdoor η_1, η_2 allows generating simulated proofs that are perfectly indistinguishable from real proofs, so that $\Pr[W_5] = \Pr[W_4]$.

Game_6: In the calculation of $\psi^\star$, we modify the generation of $\psi^\star_{\mathsf{LY}} = (C_0^\star, C_1^\star, C_2^\star, C_3^\star, \vec{C}^\star_{\theta_1}, \vec{C}^\star_{\theta_2}, \pi^\star_{\mathrm{LIN}})$. This time, instead of generating $\pi^\star_{\mathrm{LIN}}$ using the witnesses $(\theta_1^\star, \theta_2^\star)$, we compute $\pi^\star_{\mathrm{LIN}}$ as a simulated NIZK proof that $(g, g_1, g_2, C_1^\star, C_2^\star, C_3^\star)$ is a linear tuple. Recall that $\pi^\star_{\mathrm{LIN}}$ has to be generated for the Groth-Sahai CRS $(\vec{g_1}, \vec{g_2}, \vec{g}_{\mathsf{VK}^\star})$, where $\vec{g}_{\mathsf{VK}^\star} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$. This means that (ξ_1, ξ_2) can serve as a simulation trapdoor for multi-exponentiation equations in the same way as in [34]. Here, $\pi^\star_{\mathrm{LIN}}$ is a simulated proof for a true statement and simulated proofs have the same distribution as real proofs. This implies $\Pr[W_6] = \Pr[W_5]$.

Game_7: We bring a new modification to the generation of $\psi^\star_{\mathsf{LY}} = (C_0^\star, C_1^\star, C_2^\star, C_3^\star, \vec{C}^\star_{\theta_1}, \vec{C}^\star_{\theta_2}, \pi^\star_{\mathrm{LIN}})$. Namely, instead of computing $C_0^\star$ using the encryption exponents $(\theta_1^\star, \theta_2^\star)$ as in Game_5, the challenger

[6] Here, the proof π_T is simulated by using the assignment $\mathcal{X}_{T_1} = \mathcal{X}_g = \Upsilon = 1_{\mathbb{G}}$ to prove the equalities $e(\Upsilon, T_3) = e(\mathcal{X}_{T_1}, \Gamma_2)$ and $e(T_2, \mathcal{X}_g) = e(\Gamma_1, \Upsilon)$. The trapdoor (η_1, η_2) allows faking proofs that $e(\mathcal{X}_{T_1}, g) = e(T_1, g)$ and $e(\mathcal{X}_g, g) = e(g, g)$.

[7] In addition to the variable $M^\star$, the latter proof introduces an auxiliary variable $\mathcal{A}$ and provides evidence that $e(g, M^\star) = e(\mathcal{A}, B)$ and $\mathcal{A} = A$, for constants g, A, B. The NIZK simulator can use witnesses $\mathcal{A} = M^\star = 1_{\mathbb{G}}$ to prove the relation $e(g, M^\star) = e(\mathcal{A}, B)$ and simulate a proof that $e(g, \mathcal{A}) = e(g, A)$ thanks to the trapdoor of the fake CRS.

$\mathcal{B}$ computes $C_0^\star = M^\star \cdot C_1^{\star x_1} \cdot C_2^{\star x_2} \cdot C_3^{\star z}$. Note that this change is only conceptual since $C_0^\star$ has the same value as if it were computed as $C_0^\star = M^\star \cdot X_1^{\theta_1^\star} \cdot X_2^{\theta_2^\star}$. It comes that $\Pr[W_7] = \Pr[W_6]$.

Game_8: We bring yet another change in the computation of $\psi_{\mathsf{LY}}^\star = (C_0^\star, C_1^\star, C_2^\star, C_3^\star, \vec{C}_{\theta_1}^\star, \vec{C}_{\theta_2}^\star, \pi_{\mathrm{LIN}}^\star)$. Here, $C_3^\star$ is replaced by a random group element instead of being computed as $C_3^\star = g^{\theta_1^\star + \theta_2^\star}$. Under the DLIN assumption, this cannot be noticed by $\mathcal{A}$ and we have $|\Pr[W_8] - \Pr[W_7]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

Game_9: We bring one more conceptual change to the generation of $\psi_{\mathsf{LY}}^\star$ in the challenge ciphertext. Namely, we now compute $C_0^\star = M_{rand} \cdot C_1^{\star x_1} \cdot C_2^{\star x_2} \cdot C_3^{\star z}$, where $M_{rand} \stackrel{R}{\leftarrow} \mathbb{G}$ is chosen independently of the actual plaintext $M^\star$. The same arguments as in [17,34] show that this change does not affect $\mathcal{A}$'s view whatsoever since the distribution of $C_0^\star$ remains unchanged. We have $\Pr[W_9] = \Pr[W_8]$.

Game_{10}: We change again the calculation of $\psi_{\mathsf{LY}}^\star$ and now restore $(C_1^\star, C_2^\star, C_3^\star)$ to its original form $(g_1^{\theta_1^\star}, g_2^{\theta_2^\star}, g^{\theta_1^\star + \theta_2^\star})$ instead of choosing $C_3^\star$ at random. The computation of $C_0^\star = M_{rand} \cdot C_1^{\star x_1} \cdot C_2^{\star x_2} \cdot C_3^{\star z}$ remains the same as in Game_9. We clearly have $|\Pr[W_{10}] - \Pr[W_9]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

Game_{11}: Here, we bring one more conceptual change in the way to compute $\psi_{\mathsf{LY}}^\star$ in the challenge ciphertext. That is, we set $C_0^\star = M_{rand} \cdot X_1^{\theta_1^\star} \cdot X_2^{\theta_2^\star}$ instead of computing $C_0^\star$ using the private key $\mathsf{sk} = (x_1, x_2, z, \gamma_1, \gamma_2)$. Since $C_0^\star$ has the same distribution either way, we have $\Pr[W_{11}] = \Pr[W_{10}]$.

Game_{12}: As a final change in the generation of $\psi_{\mathsf{LY}}^\star$, we compute $\pi_{\mathrm{LIN}}^\star$ as a real proof (using the witnesses $\theta_1^\star, \theta_2^\star$) instead of a simulated proof. Since $(g, g_1, g_2, C_1^\star, C_2^\star, C_3^\star)$ is a linear tuple in both games, the adversary's view remains the same as in Game_{11}, so that $\Pr[W_{12}] = \Pr[W_{11}]$.

Game_{13}: We finally restore the original distribution of the vector $\vec{g_3}$ and now generate public parameters param by setting $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$ as in the real system. The usual argument allows writing $|\Pr[W_{13}] - \Pr[W_{12}]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

Game_{14}: we change again the $\mathsf{DEC}(.)$ oracle and do not apply the rejection rule of Game_4 anymore. If Σ is strongly unforgeable, we must have $|\Pr[W_{14}] - \Pr[W_{13}]| \in \mathbf{Adv}^{\text{suf-ots}}(\lambda)$.

We see that, from Game_5 onwards, the oracle $\mathsf{PROVE}(.)$ does not use the witnesses $M^\star, coins_{\psi^\star}$ at any time. Game_{14} is thus the experiment of definition 4 where the challenger's bit b is 0. When combining the above, we obtain $|\Pr[W_{14}] - \Pr[W_1]| \in \mathsf{negl}(\lambda)$, which establishes the result. □

Soundness directly follows from the security of the certification system. From a soundness adversary, the simulator interacts with a challenger for the certification security game and generates the CRS $\mathbf{g}$ for the perfect soundness setting (which precludes the generation of valid proofs for ill-formed ciphertexts). Then, soundness can only be broken by attacking the certification scheme.

B.2 Anonymity

Theorem 2. *The TGE scheme satisfies anonymity assuming that Σ is strongly unforgeable and that the DLIN and D3DH assumptions both hold in $\mathbb{G}$.*

Proof. We consider a sequence of games where the first game is the actual anonymity experiment of definition 5 while the final game is a game where even a computationally unbounded adversary has absolutely no advantage. In Game_i, we call W_i the event that the challenger $\mathcal{B}$ returns 1.

Game_1: the challenger $\mathcal{B}$ generates public parameters param which include common reference strings $\mathbf{g} = (\vec{g_1}, \vec{g_2}, \vec{g_3})$ and $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$ where $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$ and $\vec{f} = \vec{g_1}^{\eta_1} \odot \vec{g_2}^{\eta_2}$, with $\xi_1, \xi_2, \eta_1, \eta_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p$. The public key $\mathsf{pk}_{\mathsf{OA}} = (Y_1, Y_2, Y_3, Y_4)$ and param are given to $\mathcal{A}$ who generates $\mathsf{pk}_{\mathsf{GM}}$ on its own. By invoking the USER oracle, $\mathcal{A}$ is allowed to repeatedly introduce honest users in the system and certify their public keys $\mathsf{pk}_i = (X_{i,1}, X_{i,2}, \Gamma_{i,1}, \Gamma_{i,2})$. For each honest user i (for which an entry of the form $(i, \mathsf{pk}_i, \mathsf{sk}_i, \mathsf{cert}_{\mathsf{pk}_i})$ exists in keys), $\mathcal{A}$ is granted access to a decryption oracle $\mathsf{DEC}(i, .)$. The $\mathsf{CORR}(.)$ oracle also allows $\mathcal{A}$ to corrupt honest users and thereby obtain their private sk_i. In addition, $\mathcal{A}$ is allowed to obtain claims/disclaimers on behalf of honest users using the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle and expose their tracing trapdoor by invoking the $\mathsf{REVEAL}(\mathsf{sk}_{\mathsf{OA}}, .)$ oracle. Finally, $\mathcal{A}$ makes a number of opening queries for ciphertexts of its choice, for which $\mathcal{B}$ identifies the receivers of using $\mathsf{sk}_{\mathsf{OA}}$. At some point, $\mathcal{A}$ outputs a pair of distinct identifiers (i_0, i_1) along with a tuple $((A, B), M^*, L, \mathsf{pk}_{\mathcal{R}})$ such that $((A, B), M^*) \in \mathcal{R}$ and obtains, as a challenge, a group encryption $\psi^\star = \mathsf{VK}^\star || (T_1^\star, T_2^\star, T_3^\star, T_4^\star) || \psi_{\mathsf{LY}}^\star || \psi_{\mathsf{K}_1}^\star || \psi_{\mathsf{K}_2}^\star || \sigma^\star$ of $M^\star$ under pk_b, for some bit $b \in \{0, 1\}$ of $\mathcal{B}$'s choice. We assume w.l.o.g. that the one-time signature key pair $(\mathsf{SK}^\star, \mathsf{VK}^\star)$ is generated at the very beginning of the game. After the challenge phase, $\mathcal{A}$ obtains proofs $\pi_{\psi^\star}^\star$ for $\psi^\star$ and makes new queries under the natural restrictions. It finally outputs b' and we let the challenger $\mathcal{B}$ output 1 in the event W_1 that $b' = b$.

Game_2: is as Game_1 but $\mathcal{B}$ aborts and outputs a random bit in the event F_2 that $\mathcal{A}$ queries the opening of a ciphertext $\psi = \mathsf{VK} || (T_1, T_2, T_3, T_4) || \psi_{\mathsf{LY}} || \psi_{\mathsf{K}_1} || \psi_{\mathsf{K}_2} || \sigma$ such that $\mathsf{VK} = \mathsf{VK}^\star$ and σ is valid (we may assume that $\mathsf{VK}^\star$ is generated at the outset of the game). If F_2 occurs, $\mathcal{A}$ is necessarily able to break the strong security of Σ (even if the query occurs before the challenge phase, $\mathcal{A}$ has forged a signature without seeing any signature) and $|\Pr[W_2] - \Pr[W_1]| \leq \Pr[F_2] \leq \mathbf{Adv}^{\text{suf-ots}}(\lambda)$, so that Game_2 proceed identically to Game_1 if Σ is strongly unforgeable.

Game_3: is like Game_2 with the following modification. At the beginning of the game, the challenger $\mathcal{B}$ chooses two indexes $i_0^\star, i_1^\star \xleftarrow{R} \{1, \ldots, q_u\}$, where q_u denotes the maximal number of queries to the USERS oracle, as a guess that $\mathcal{B}$ will choose to be challenged on a pair of users that corresponds to those introduced at the $i_0^\star$-th and $i_1^\star$-th invocations of USERS. During the challenge phase, $\mathcal{B}$ halts and outputs a random bit if its initial guess for $(i_0^\star, i_1^\star)$ was incorrect (*i.e.*, if $(i_0^\star, i_1^\star) \neq (i_0, i_1)$). Since the choice of $(i_0^\star, i_1^\star)$ is independent of $\mathcal{A}$'s view, we have $\Pr[W_3] = \Pr[W_2]/q_u^2$.

Game_4: is identical to Game_3 but we modify param by changing the distribution of $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$ and that of the vectors $\{\vec{h_i}\}_{i=0}^{\ell}$ and $\vec{g_3}$. Instead of choosing $\mathbf{f}$ as a perfectly binding CRS, we generate it as a perfectly NIWI Groth-Sahai CRS, where $\vec{f} = \vec{g_1}^{\eta_1} \odot \vec{g_2}^{\eta_2} \odot (1, 1, g)^{-1}$ and $\eta_1, \eta_2 \xleftarrow{R} \mathbb{Z}_p$. Moreover, instead of setting $\vec{h_i} = \vec{g_1}^{\zeta_{i,1}} \odot \vec{g_2}^{\zeta_{i,2}}$ for each $i \in \{0, \ldots, \ell\}$, we set each vector $\vec{h_i}$ as $\vec{h_i} = \vec{g_1}^{\zeta_{i,1}} \odot \vec{g_2}^{\zeta_{i,2}} \odot (1, 1, g)^{\zeta_{i,3}}$ for randomly chosen $\zeta_{i,1}, \zeta_{i,2}, \zeta_{i,3} \xleftarrow{R} \mathbb{Z}_p$ for $i = 0$ to ℓ. Finally, instead of setting $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$, with $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p$, we define $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2} \odot (1, 1, g)^{-\mathsf{VK}^\star}$ (said otherwise, $\vec{g_3}$ is now a BBS encryption of $g^{-\mathsf{VK}^\star}$ instead of an encryption of $1_{\mathbb{G}}$). Under the DLIN assumption, these changes should remain unnoticed for any PPT adversary $\mathcal{A}$ and we have $|\Pr[W_4] - \Pr[W_3]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

Game_5: is as Game_4 but we change the treatment of USER queries. Namely, instead of generating π_{venc} using the witnesses at step 2 of the JOIN protocol, the challenger uses the trapdoor (η_1, η_2) of the Groth-Sahai CRS $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$ to generate a simulated NIZK proof π_{venc} without using the witnesses. Clearly, since simulated NIZK proofs have the same distribution as real proofs on a simulated CRS $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$, $\mathcal{A}$'s view remains the same as in Game_4. We have $\Pr[W_5] = \Pr[W_4]$.

Game_6: we bring a new modification to the USER oracle at the $i_0^\star$-th and $i_1^\star$-th queries. Instead of computing Φ_{venc} by verifiably encrypting the actual witness $\Gamma_0 = g^{\gamma_0 \gamma_1}$, $\mathcal{B}$ generates Φ_{venc} by encrypting a random group element. The semantic security of the BBS cryptosystem guarantees that $\mathcal{A}$'s view will not be significantly affected by this change. We have $|\Pr[W_6] - \Pr[W_5]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

Game_7: in this game, we modify the oracle $\mathcal{P}$ which generates proofs about the challenge ciphertext $\psi^\star$ after the challenge phase. Namely, instead of generating proofs $\pi_{\psi^\star}$ using the real witnesses $coins_{\psi^\star} = \{\delta^\star, \varrho^\star, \{(z_{i,1}^\star, z_{i,2}^\star)\}_{i=1,2}, (\theta_1^\star, \theta_2^\star)\}$, $\mathcal{B}$ simulates these proofs using the trapdoor (η_1, η_2) of the CRS $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$ in the same way as the simulator of the proof of Theorem 1 (in Game_5).

Since $\mathbf{f}$ is a perfectly hiding CRS in both games, this modification leaves the distribution of $\pi_{\psi^\star}$ unchanged. We have $\Pr[W_7] = \Pr[W_6]$.

Game_8: here, we modify the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle and simulate the non-interactive proofs $\vec{\pi}_{\tau,1}, \vec{\pi}_{\tau,2}$ without using the witness $\varGamma_{-1} = g^{1/\gamma_1}$. To this end, the challenger $\mathcal{B}$ uses the trapdoor information $\{(\zeta_{i,1}, \zeta_{i,2}, \zeta_{i,3})\}_{i=0}^{\ell}$ associated with the vectors $\vec{h_i} = \vec{g_1}^{\zeta_{i,1}} \odot \vec{g_2}^{\zeta_{i,3}}$. It first chooses a random ℓ-bit string $\mathsf{v} = \mathsf{v}[1] \ldots \mathsf{v}[\ell] \in \{0,1\}^\ell$ in the range of the chameleon hashing algorithm CMhash subject to the condition $J_3(\mathsf{v}) = \sum_{i=0}^{\ell} \mathsf{v}[i]\zeta_{i,3} \neq 0$. If we similarly define

$$J_1(\mathsf{v}) = \sum_{i=0}^{\ell} \mathsf{v}[i]\zeta_{i,1} \qquad \text{and} \qquad J_2(\mathsf{v}) = \sum_{i=0}^{\ell} \mathsf{v}[i]\zeta_{i,2},$$

we know that $\vec{h_\mathsf{v}} = \vec{g_1}^{J_1(\mathsf{v})} \odot \vec{g_2}^{J_2(\mathsf{v})} \odot (1,1,g)^{J_3(\mathsf{v})}$ with $J_3(\mathsf{v}) \neq 0$, so that $(\vec{g_1}, \vec{g_2}, \vec{h_\mathsf{v}})$ forms a perfectly hiding Groth-Sahai CRS. Consequently, the proofs $\vec{\pi}_{\tau,1}$ and $\vec{\pi}_{\tau,2}$ can be simulated as follows. In order to satisfy the first relation of (11), $\mathcal{B}$ generates $\vec{C}_{\varGamma_{-1}} = \vec{g_1}^{r_{-1}} \odot \vec{g_2}^{s_{-1}} \odot \vec{h}_\mathsf{v}^{t_{-1}}$ and $\vec{C}_{\mathcal{X}} = \vec{g_1}^{r_\mathcal{X}} \odot \vec{g_2}^{s_\mathcal{X}} \odot \vec{h}_\mathsf{v}^{t_\mathcal{X}}$ as commitments to $\varGamma_{-1} = \mathcal{X} = 1_\mathbb{G}$, which immediately yields a valid assignment for the relation $e(T_{\delta,1}, \varGamma_{-1}) = e(T_1, \mathcal{X}_\tau)$. As for the second relation of (11), $\mathcal{B}$ uses the trapdoor $(J_1(\mathsf{v}), J_2(\mathsf{v}), J_3(\mathsf{v}))$ to simulate a fake proof that $e(g, \mathcal{X}_\tau) = e(g,g)$ by setting $\vec{\pi}_{\tau,2} = (\pi_{\tau,2,1}, \pi_{\tau,2,2}, \pi_{\tau,2,3})$ as

$$(\pi_{\tau,2,1}, \pi_{\tau,2,2}, \pi_{\tau,2,3}) = \left(g^{r_\mathcal{X}} \cdot g^{J_1(\mathsf{v})/J_3(\mathsf{v})},\ g^{s_\mathcal{X}} \cdot g^{J_2(\mathsf{v})/J_3(\mathsf{v})},\ g^{t_\mathcal{X}} \cdot g^{-1/J_3(\mathsf{v})}\right), \tag{16}$$

which satisfies $E(g, \vec{C}_\mathcal{X}) = E(g, (1,1,g)) \odot E(\pi_{\tau,2,1}, \vec{g_1}) \cdot E(\pi_{\tau,2,2}, \vec{g_2}) \cdot E(\pi_{\tau,2,3}, \vec{h}_\mathsf{v})$ and thus forms a valid proof for the second relation of (11). Finally, $\mathcal{B}$ uses the trapdoor[8] tk of the chameleon hash function to find random hashing coins $s_{hash} \in \mathcal{R}_{hash}$ that explain v as a a hash value $\mathsf{v} = \mathsf{CMhash}(hk, (\psi, L, \mathsf{pk}), s_{hash})$. This completes the description of the modified generation of claims and disclaimers (12) in Game_8. Clearly, since simulated NIZK proofs are perfectly indistinguishable from actual proofs on a simulated CRS, $\mathcal{A}$'s view is the same as previously and we have $\Pr[W_8] = \Pr[W_7]$.

Game_9: we bring a first modification in the generation the challenge ciphertext $\psi^\star$. Instead of generating the traceability components $(T_1^\star, T_2^\star, T_3^\star, T_4^\star)$ as

$$(T_1^\star, T_2^\star, T_3^\star, T_4^\star) = \left(g^{\delta^\star}, \varGamma_{b,1}^{\delta^\star/\varrho^\star}, \varGamma_{b,2}^{\varrho^\star}, (\varLambda_0^{\mathsf{VK}^\star} \cdot \varLambda_1)^{\delta^\star}\right)$$

using $\mathsf{pk}_b = (X_{b,1}, X_{b,2}, \varGamma_{b,1}, \varGamma_{b,2})$, we set $(T_1^\star, T_2^\star, T_3^\star, T_4^\star) = \left(g^{\delta^\star}, T_2^\star, T_3^\star, (\varLambda_0^{\mathsf{VK}^\star} \cdot \varLambda_1)^{\delta^\star}\right)$ for randomly chosen $T_2^\star, T_3^\star \xleftarrow{R} \mathbb{G}$. Lemma 1 shows that, if the D3DH assumption holds, this modification is not noticeable to $\mathcal{A}$, so that we have $|\Pr[W_9] - \Pr[W_8]| \leq \mathbf{Adv}^{\mathrm{D3DH}}(\lambda)$. In Game_9, we remark that $(T_1^\star, T_2^\star, T_3^\star, T_4^\star)$ are independent of the challenger's bit $b \in_R \{0,1\}$.

Game_{10}: in this game, we modify the generation of the challenge ciphertext $\psi^\star$ and compute $(\psi^\star_{\mathsf{K}_1}, \psi^\star_{\mathsf{K}_2})$ as encryptions of random group elements instead of the actual components $(\varGamma_{b,1}, \varGamma_{b,2})$ of the public key $\mathsf{pk}_b = (X_{b,1}, X_{b,2}, \varGamma_{b,1}, \varGamma_{b,2})$. Since the encryption exponents $\{z^\star_{i,1}, z^\star_{i,2}\}_{i=1,\ldots,6}$ are not used anymore to generate the proof $\pi_{\psi^\star}$ in Game_9, we would be able to build a selective-tag weak CCA2 attacker[9] against Kiltz's tag-based encryption system (recall that opening queries do not

[8] Here, the reason to use a chameleon hash function becomes clear: if v were a uniquely determined by (ψ, L, pk), we would be unable to rule out the possibility of $\mathcal{A}$ to invoke the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle on ciphertext label pairs (ψ, L) such that $J_3(\mathsf{v}) = 0$, in which case the simulator would get stuck. Our solution to this problem is to randomize the hashing algorithm so as to keep the adversary from being in full control of the hash value $\mathsf{v} \in \{0,1\}^\ell$ which allows forming the Groth-Sahai CRS $(\vec{g_1}, \vec{g_2}, \vec{h}_\mathsf{v})$.

[9] Selective-tag weak CCA2 security is defined [31] via a game where the adversary $\mathcal{A}$ chooses a tag $t^\star$ and then obtains a public key and access to a decryption oracle which $\mathcal{A}$ can query for any ciphertext-tag pair (C, t) such that $t \neq t^\star$. At the challenge phase, $\mathcal{A}$ chooses plaintexts m_0, m_1 and receives a ciphertext $C^\star$ encrypting m_b (under the tag $t^\star$) for some bit $b \xleftarrow{R} \{0,1\}$ that $\mathcal{A}$ eventually aims to guess after further decryption queries.

involve $\mathsf{VK}^\star$ unless the rejection rule of Game_2 applies) if $\mathcal{A}$'s view would be significantly affected by this change. The result proved in [31] implies that we have $|\Pr[W_{10}] - \Pr[W_9]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

In Game_{10}, we note that the only information about the challenger's bit $b \in \{0,1\}$ carried by the challenge ciphertext $\psi^\star$ appears in the $C_0^\star$ component of the $\pi_{\mathrm{LY}}^\star$ ciphertext. We are thus left with removing this information.

Game_{11}: is like Game_{10} but we bring one final change in the challenge ciphertext $\psi^\star$. Specifically, we modify $\pi_{\mathrm{LY}}^\star = (C_0^\star, C_1^\star, C_2^\star, C_3^\star, \vec{C}_{\theta_1}^\star, \vec{C}_{\theta_2}^\star, \pi_{\mathrm{LIN}}^\star)$ and replace $C_0^\star$ by a random group element. This change can be justified as follows. Recall that, in the latter ciphertext, $\pi_{\mathrm{LIN}}^\star$ has to be generated for the CRS $(\vec{g_1}, \vec{g_2}, \vec{g}_{\mathsf{VK}^\star})$ which is a perfectly hiding Groth-Sahai CRS. Moreover, all proofs $\pi_{\psi^\star}$ involving the challenge ciphertexts are simulated and without using the encryption exponents. This means that we can use a sub-sequence of games which is identical to the sub-sequence from Game_6 to Game_{11} in the proof of Theorem 1. This sub-sequence thus ends up with a game where $\pi_{\mathrm{LY}}^\star$ is an encryption under the public key $(X_{b,1}, X_{b,2})$ of a random message M_{rand} which is completely independent of $\mathcal{A}$'s view. Consequently, this amounts to replacing $C_0^\star$ by a random element of $\mathbb{G}$. The same arguments as in the proof of Theorem 1 thus imply that $|\Pr[W_{11}] - \Pr[W_{10}]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\lambda)$.

In Game_{11}, we observe that the challenge ciphertext $\psi^\star$ carries no information about $b \in \{0,1\}$ whatsoever. $\square$

Lemma 1. *If the D3DH holds in $\mathbb{G}$, no PPT adversary $\mathcal{A}$ can distinguish* Game_9 *from* Game_8*.*

Proof. Towards a contradiction, let us assume that $\mathcal{A}$ can tell apart Game_8 and Game_9 with noticeable probability. We build a D3DH distinguisher $\mathcal{B}^{\mathrm{D3DH}}$ as follows.

Our algorithm $\mathcal{B}^{\mathrm{D3DH}}$ takes as input a D3DH instance $(g, g^a, g^b, g^c, \varsigma)$ with the task of deciding if $\varsigma = g^{abc}$ or $\varsigma \in_R \mathbb{G}$. To this end, $\mathcal{B}^{\mathrm{D3DH}}$ generates the public parameters param by choosing $g_1, g_2 \overset{R}{\leftarrow} \mathbb{G}$ and setting up vectors $\vec{g_1} = (g_1, 1, g)$, $\vec{g_2} = (1, g_2, g)$ and $\vec{f} = \vec{g_1}^{\eta_1} \odot \vec{g_2}^{\eta_2} \odot (1,1,g)^{-1}$ with $\eta_1, \eta_2 \overset{R}{\leftarrow} \mathbb{Z}_p$. For $i = 0$ to ℓ, it also defines $\vec{h_i} = \vec{g_1}^{\zeta_{i,1}} \odot \vec{g_2}^{\zeta_{i,2}} \odot (1,1,g)^{\zeta_{i,3}}$ for randomly chosen $\zeta_{i,1}, \zeta_{i,2}, \zeta_{i,3} \overset{R}{\leftarrow} \mathbb{Z}_p$. It also generates a one-time signature key pair $(\mathsf{SK}^\star, \mathsf{VK}^\star) \leftarrow \mathcal{G}(\lambda)$ and chooses a key pair $(hk, tk) \leftarrow \mathcal{G}(\lambda)$ for the chameleon hash function $\mathcal{CMH} = (\mathsf{CMKg}, \mathsf{CMhash}, \mathsf{CMswitch})$. As for the pair $(\Lambda_0, \Lambda_1) \in \mathbb{G}^2$, $\mathcal{B}^{\mathrm{D3DH}}$ sets $\Lambda_0 = g^a$ and $\Lambda_1 = (g^a)^{-\mathsf{VK}^\star} \cdot g^{\omega_1}$, with $\omega_1 \overset{R}{\leftarrow} \mathbb{Z}_p$. At the outset of the game, $\mathcal{B}^{\mathrm{D3DH}}$ also picks two indexes $i_0^\star, i_1^\star \overset{R}{\leftarrow} \{1, \ldots, q_u\}$, where q_u denotes the maximal number of queries to USERS, exactly as the actual challenger does from Game_3 onwards. During the game, $\mathcal{B}^{\mathrm{D3DH}}$ halts and outputs a random bit if $\mathcal{A}$ chooses to corrupt either user $i_0^\star$ or user $i_1^\star$ since it means that $\mathcal{B}^{\mathrm{D3DH}}$ failed to correctly guess which users the adversary would choose to be challenged upon. If $\mathcal{A}$ indeed chooses $i_0^\star, i_1^\star$ as its target indexes in the challenge phase, $\mathcal{B}^{\mathrm{D3DH}}$ will always be able to reveal the correct private keys. The way to answer queries to the USERS oracle depends on the index i of the query:

- If $i \notin \{i_0^\star, i_1^\star\}$, the reduction $\mathcal{B}^{\mathrm{D3DH}}$ generates $\mathsf{pk}_i = (X_{i,1}, X_{i,2}, \Gamma_{i,1}, \Gamma_{i,2})$ faithfully and sets $(\Gamma_{i,1}, \Gamma_{i,2}) = (g^{\gamma_{i,1}}, g^{\gamma_{i,2}})$ for randomly chosen $\gamma_{i,1}, \gamma_{i,2} \overset{R}{\leftarrow} \mathbb{Z}_p$. This implies that $\mathcal{B}^{\mathrm{D3DH}}$ knows the tracing trapdoor $\Gamma_{i,0} = g^{\gamma_{i,1}\gamma_{i,2}}$ and can verifiably encrypt it at step 2 of the JOIN protocol.
- If $i = i_0^\star$, algorithm $\mathcal{B}^{\mathrm{D3DH}}$ generates the $i_0^\star$-th pubic key as $\mathsf{pk}_{i_0^\star} = (X_{i_0^\star,1}, X_{i_0^\star,2}, \Gamma_{i_0^\star,1}, \Gamma_{i_0^\star,2})$ with $\Gamma_{i_0^\star,1} = (g^a)^{\rho_{0,\gamma_1}}$ and $\Gamma_{i_0^\star,2} = (g^b)^{\rho_{0,\gamma_2}}$, for randomly chosen $\rho_{0,\gamma_1}, \rho_{0,\gamma_2} \overset{R}{\leftarrow} \mathbb{Z}_p$. It also generates Φ_{venc} as a BBS encryption of a random group element according to the change introduced in Game_6.
- If $i = i_1^\star$, the $i_1^\star$-th pubic key is generated as $\mathsf{pk}_{i_1^\star} = (X_{i_1^\star,1}, X_{i_1^\star,2}, \Gamma_{i_1^\star,1}, \Gamma_{i_1^\star,2})$ with $\Gamma_{i_1^\star,1} = (g^a)^{\rho_{1,\gamma_1}}$ and $\Gamma_{i_1^\star,2} = (g^b)^{\rho_{1,\gamma_2}}$, for randomly drawn $\rho_{1,\gamma_1}, \rho_{1,\gamma_2} \overset{R}{\leftarrow} \mathbb{Z}_p$. As in the previous case, it computes Φ_{venc} as a BBS encryption of a random group element.

At any time, $\mathcal{A}$ is also allowed to query the REVEAL oracle and ask it for the tracing trapdoor trace_i of any honest user. When $\mathcal{A}$ queries the trapdoor of user $i \in \{1, \ldots, q_u\}$, $\mathcal{B}^{\mathrm{D3DH}}$ responds as follows.

- If $i \notin \{i_0^\star, i_1^\star\}$, $\mathcal{B}^{\mathrm{D3DH}}$ knows $\mathsf{trace}_i = \Gamma_{i,0} = g^{\gamma_{i,1}\gamma_{i,2}}$ and simply returns it to $\mathcal{A}$.
- If $i \in \{i_0^\star, i_1^\star\}$, algorithm $\mathcal{B}^{\mathrm{D3DH}}$ is unable to answer the query as it failed to correctly predict the indexes i_0, i_1 of the two users involved in the challenge phase (recall the REVEAL queries are disallowed for these users). It thus halts and outputs a random bit as the challenger $\mathcal{B}$ always does in this case due to the modification introduced in Game_3.

Queries to the CORR oracle are answered in the same way as REVEAL queries except that $\mathcal{B}^{\mathrm{D3DH}}$ returns the queried private key sk_i instead of the tracing trapdoor.

At any time, $\mathcal{A}$ may also query the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle for triples (i, ψ, L) of its choice. To answer the query, $\mathcal{B}$ parses ψ as $\psi = \mathsf{VK}||(T_1, T_2, T_3, T_4)||\psi_{\mathsf{LY}}||\psi_{\mathsf{K}_1}||\psi_{\mathsf{K}_2}||\sigma$ and returns $\perp$ if either: (i) σ is not a valid one-time signature; (ii) $e(g, T_4) \neq e(T_1, \Lambda_0^{\mathsf{VK}} \cdot \Lambda_1)$. Otherwise, since we necessarily have $\mathsf{VK} \neq \mathsf{VK}^\star$ unless the failure event introduced in Game_2 occurs. To answer the query, $\mathcal{B}^{\mathrm{D3DH}}$ considers the following situations:

- If $i \notin \{i_0^\star, i_0^\star\}$, $\mathcal{B}^{\mathrm{D3DH}}$ is able to compute $T_{\delta,1} = T_1^{\gamma_{i,1}} = \Gamma_{i,1}^{\delta}$ since it entirely knows the private key $\mathsf{sk}_i = (x_{i,1}, x_{i,2}, \gamma_{i,1}, \gamma_{i,2})$ of the i-th honest user.
- if $i = i_d^\star$ for some $d \in \{0,1\}$, we have $T_1 = g^\delta$ and $T_4 = \left((g^a)^{\mathsf{VK}-\mathsf{VK}^\star} \cdot g^{\omega_1}\right)^\delta$, so that $\mathcal{B}^{\mathrm{D3DH}}$ can compute
$$T_{\delta,1} = \Gamma_{i_d^\star,1}^{\delta} = (T_4/T_1^{\omega_1})^{\rho_{d,\gamma_1}/(\mathsf{VK}-\mathsf{VK}^\star)}.$$
Having computed $T_{\delta,1}$, $\mathcal{B}^{\mathrm{D3DH}}$ is able to compute a valid claim/disclaimer τ by simulating the proofs $\vec{\pi}_{\tau,1}$, $\vec{\pi}_{\tau,2}$ in the same way as in Game_8.

When it comes to build the challenge ciphertext $\psi^\star$, $\mathcal{B}^{\mathrm{D3DH}}$ uses its D3DH instance $(g, g^a, g^b, g^c, \varsigma)$ to construct the traceability components $(T_1^\star, T_2^\star, T_3^\star, T_4^\star)$. Specifically, it chooses $\varrho \overset{R}{\leftarrow} \mathbb{Z}_p$, flips a fair binary coin $b \overset{R}{\leftarrow} \{0,1\}$ and computes
$$T_1^\star = g^c, \qquad T_2^\star = \varsigma^{\rho_{b,\gamma_1}/\varrho}, \qquad T_3^\star = g^{\rho_{b,\gamma_2}\cdot\varrho}, \qquad T_4^\star = (g^c)^{\omega_1}.$$
We observe that, if $\varsigma = g^{abc}$, $(T_1^\star, T_2^\star, T_3^\star, T_4^\star)$ has the distribution of a well-formed tuple for the encryption exponent $\delta^\star = c$ and $\varrho^\star = \varrho/b$. In this case, the challenge ciphertext is distributed as in Game_8. Now, if $\varsigma \in_R \mathbb{G}$, $(T_1^\star, T_2^\star, T_3^\star, T_4^\star)$ has the distribution of a valid tuple where $T_2^\star$ and $T_3^\star$ have been tampered with and replaced by random group elements. In the latter case, $\mathcal{A}$'s view is the same as in Game_9.

It follows that any PPT adversary $\mathcal{A}$ that causes the challenger to output 1 with noticeably different probabilities in Game_8 and Game_9 translates into an equally efficient distinguisher $\mathcal{B}^{\mathrm{D3DH}}$ for the D3DH assumption. $\square$

B.3 Soundness and Claiming-Soundness

The soundness property directly follows from the security of AHO signatures and the perfect soundness of Groth-Sahai proofs. In particular, given that $\{\psi_{\mathsf{K}_i}\}_{i=1,2}$ encrypt $\{\Gamma_i\}_{i=1,2}$ which are also used by the TRACE algorithm, the perfect soundness of π_ψ guarantees that TRACE and OPEN cannot identify different users. The proof of the following property is straightforward and thus omitted.

Theorem 3. *The scheme provides soundness assuming that the q-SFP assumption holds, where q is the number of queries to the user registration oracle* $\mathsf{REG}(\mathsf{sk}_{\mathsf{GM}}, .)$.

As for the soundness of the claiming algorithm, it can be proved under the DLIN assumption.

Theorem 4. *The scheme provides claiming soundness assuming that the DLIN assumption holds and that the chameleon hash function is collision-resistant.*

Proof. We first remark that, since non-interactive proofs π_ψ are generated for a perfectly sound CRS $(\vec{g_1}, \vec{g_2}, \vec{f})$, even an unbounded adversary would be unable to get the experiment of Definition 7 to return 1 before reaching the last "If" instruction (*i.e.*, by generating a non-trivial claim on behalf of a user that is not under its control). In the latter case, we show that the adversary would necessarily contradict the DLIN assumption.

The proof proceeds via a sequence of games. In each game, we denote by S_i the event that the adversary $\mathcal{A}$ wins because it manages to claim or disclaim a ciphertext-label pair on behalf of some uncorrupted honest user. In the following, we denote by q_u and q_c the number of queries to the USERS and $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracles, respectively.

Game_{real} : This is the real game at the beginning of which the challenger $\mathcal{B}$ generates param and gives them to the adversary $\mathcal{A}$. The latter replies by choosing a group manager's public key $\mathsf{pk}_{\mathsf{GM}}$. Then, it receives an honestly generated key pair $(\mathsf{sk}_{\mathsf{OA}}, \mathsf{pk}_{\mathsf{OA}})$. The adversary $\mathcal{A}$ ends by outputting a pair $(\mathsf{pk}^\star_{\mathcal{R}}, x^\star)$, a ciphertext-label pair $(\psi^\star, L^\star)$ – which presumably encrypts a witness $w^\star$ such that $(x^\star, w^\star) \in \mathcal{R}$ – along with a proof $\pi^\star_{\psi^\star}$, a statement $\mathsf{statement}^\star$ and a $\mathsf{database}^\star$. The adversary is deemed successful if it fulfills the conditions specified by Definition 7, in which case the challenger $\mathcal{B}$ outputs 1. We thus have $\Pr[S_{real}] = \mathbf{Adv}^{\text{claiming-soundness}}(\mathcal{A})$.

Game_0: is like Game_{real} with one modification. At the outset of the game, the challenger $\mathcal{B}$ picks a random index $i^\star \overset{R}{\leftarrow} \{1, \ldots, q_u\}$ – where q_u is the maximal number of queries to the USERS oracle – hoping that $\mathcal{B}$ will choose to output a statement (τ, pk_j) on behalf of the user introduced at the $i^\star$-th query to USERS (in other words, $\mathcal{B}$ guesses that $j = i^\star$). When $\mathcal{A}$ terminates, $\mathcal{B}$ halts and outputs a random bit if its initial choice for $i^\star$ turns out to be incorrect. Since the choice of $i^\star$ is independent of $\mathcal{A}$'s view, we have $\Pr[S_0] = \Pr[S_{real}]/q_u$.

Game_1 : This game is the same as Game_0 with the difference that we modify the public parameters param by choosing the vectors $\{\vec{h_i}\}_{i=0}^{\ell}$ as random vectors in $\mathbb{G}^3$ instead of sampling them in $\mathrm{span}(\vec{g_1}, \vec{g_2})$. Under the DLIN assumption, this should have negligible effect on $\mathcal{A}$'s behavior. We thus have $|\Pr[S_1] - \Pr[S_0]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\mathcal{B})$.

Game_2 : This game is identical to Game_1 but we modify again the generation of param. Namely, the vectors $(\vec{g_1}, \vec{g_2}, \{\vec{h_i}\}_{i=0}^{\ell})$ are chosen by setting $\vec{g_1} = (g_1, 1_{\mathbb{G}}, g)$ and $\vec{g_2} = (1_{\mathbb{G}}, g_2, g)$, with $g_1, g_2 \overset{R}{\leftarrow} \mathbb{G}$. As for $\{\vec{h_i}\}_{i=0}^{\ell}$, they are obtained as

$$\begin{aligned} \vec{h_0} &= \vec{g_1}^{\zeta_{0,1}} \cdot \vec{g_2}^{\zeta_{0,2}} \cdot (1,1,g)^{\zeta_{0,3}} \cdot (1,1,g)^{\mu \cdot \kappa - \rho_0} \\ \vec{h_i} &= \vec{g_1}^{\zeta_{i,1}} \cdot \vec{g_2}^{\zeta_{i,2}} \cdot (1,1,g)^{\zeta_{i,3}} \cdot (1,1,g)^{-\rho_i}, \qquad i \in \{1, \ldots, \ell\} \end{aligned} \tag{17}$$

with $\mu \overset{R}{\leftarrow} \{0, \ldots, \ell\}$, $\zeta_{0,1}, \zeta_{1,1}, \ldots, \zeta_{\ell,1} \overset{R}{\leftarrow} \mathbb{Z}_p$, $\zeta_{0,2}, \zeta_{1,2}, \ldots, \zeta_{\ell,2} \overset{R}{\leftarrow} \mathbb{Z}_p$, $\zeta_{0,3}, \zeta_{1,3}, \ldots, \zeta_{\ell,3} \overset{R}{\leftarrow} \mathbb{Z}_p$ and $\rho_0, \rho_1, \ldots, \rho_\ell \overset{R}{\leftarrow} \{0, \ldots, \kappa - 1\}$, with $\kappa = 2q$ and where q is the number of distinct tags across all signing queries. Note that this change is only conceptual since $\{\vec{h_i}\}_{i=0}^{\ell}$ have the same distribution as in Game_1. We thus have $\Pr[S_2] = \Pr[S_1]$.

Game_3 : This game is like Game_2 with the difference that the challenger halts and outputs a random bit in the event that one of $\mathcal{A}$'s queries to the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle results in a chameleon hash value $\mathsf{v}_i = \mathsf{CMhash}(\psi_i, L_i, \mathsf{pk}_i)$ that collides with the one produced by $\mathcal{A}$ as part of its output $\mathsf{statement}$. If the chameleon hash function is collision-resistant, Game_3 has negligible chance of departing from Game_2 and we have the inequality $|\Pr[S_3] - \Pr[S_2]| \leq \mathbf{Adv}^{\text{CM-hash}}(\lambda)$.

Game_4 : In this game, we raise an event F_4, which causes the challenger $\mathcal{B}$ to abort and output a random bit if it does *not* occur. Let $\mathsf{v}_1, \ldots, \mathsf{v}_{q_c}$ be the distinct outputs of CMhash successively involved in $\mathcal{A}$'s queries throughout the game and let $\mathsf{v}^\star$ be the chameleon hash value involved in $\mathcal{A}$'s fake claim/disclaimer τ, which is part of $\mathsf{statement}$. We know that $\mathsf{v}^\star \in \{\mathsf{v}_1, \ldots, \mathsf{v}_{q_c}\}$ unless the failure event introduced in Game_3 occurs. For each string $\mathsf{v} = \mathsf{v}[1] \ldots \mathsf{v}[\ell] \in \{0,1\}^\ell$, we now define the function $J(\mathsf{v}) = \mu\kappa - \rho_0 - \sum_{i=1}^{\ell} \rho_i \mathsf{v}[i]$ and consider the event F_4 that

$$J(\mathsf{v}^\star) = 0 \qquad \wedge \qquad \bigwedge_{\mathsf{v}_j \in \{\mathsf{v}_1, \ldots, \mathsf{v}_{q_c}\} \setminus \{\mathsf{v}^\star\}} J(\mathsf{v}_j) \neq 0.$$

We observe that the exponents $\rho_0, \rho_1, \ldots, \rho_L$ are completely independent of the adversary's view. The analysis of [39,7] shows that

$$\Pr[S_4 \wedge F_4] \geq \Pr[S_3]^2 / (27 \cdot q_c \cdot (\ell + 1)).$$

Hence, we *a fortiori* have $\Pr[S_4] \geq \frac{\Pr[S_3]^2}{27 \cdot q_c \cdot (\ell+1)}$.

This follows from the fact that, for any set of queries, a lower bound on the probability of event F_4 is $1/(2q_c(\ell+1))$.

Game_5 : In this game, we bring one more modification to the distribution of param. Namely, the challenger $\mathcal{B}$ picks $\vec{g_1} = (g_1, 1, g)$ and $\vec{g_2} = (1, g_2, g)$ as before but, instead of choosing the vectors $\{\vec{h}_i\}_{i=0}^{\ell}$ as previously, $\mathcal{B}$ sets them as

$$\begin{aligned} \vec{h}_0 &= \vec{g_1}^{\zeta_{0,1}} \cdot \vec{g_2}^{\zeta_{0,2}} \cdot (1,1,g)^{\mu \cdot \kappa - \rho_0} \\ \vec{h}_i &= \vec{g_1}^{\zeta_{i,1}} \cdot \vec{g_2}^{\zeta_{i,2}} \cdot (1,1,g)^{-\rho_i}, \qquad i \in \{1, \ldots, \ell\} \end{aligned} \tag{18}$$

which amounts to setting $\zeta_{0,3} = \zeta_{1,3} = \ldots = \zeta_{\ell,3} = 0$. If the DLIN assumption holds, $\mathcal{A}$'s should not be able to tell the difference between Game_5 and Game_4. Otherwise, we would be able to build an algorithm $\mathcal{B}^{\mathrm{DLIN}}$ such that $|\Pr[S_5] - \Pr[S_4]| \leq \mathbf{Adv}(\mathcal{B}^{\mathrm{DLIN}})$.

In Game_5, we show that a successful forger $\mathcal{A}$ implies an algorithm solving the inverse computational Diffie-Hellman problem and *a fortiori* contradicting the DLIN assumption. Specifically, we build an algorithm $\mathcal{B}^{\text{inv-CDH}}$ that takes as input group elements $(g, g^a) \in \mathbb{G}^2$ and computes $g^{1/a}$ with non-negligible probability, which is known to be as hard as solving the Diffie-Hellman problem. We thus have[10] $\Pr[S_5] \leq (\mathbf{Adv}^{\mathrm{CDH}}(\lambda))^2$.

To describe $\mathcal{B}^{\text{inv-CDH}}$, we first recall that, when the adversary $\mathcal{A}$ terminates, it outputs a statement (τ, pk_j), where pk_j is the public key of some user in $\mathsf{database}$ that did not fell under $\mathcal{A}$'s control. Moreover, (τ, pk_j) is not a claim that $\mathcal{A}$ trivially obtained by probing the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle on the input $(\psi^\star, L^\star, \mathsf{pk}_j)$. To solve its problem instance, $\mathcal{B}^{\text{inv-CDH}}$ prepares param by generating $(\vec{g_1}, \vec{g_2}, \{\vec{h}_i\}_{i=0}^{\ell})$ as specified in Game_5. In addition, $\mathcal{B}^{\text{inv-CDH}}$ chooses $\omega_0, \omega_1, \iota_0, \iota_1 \xleftarrow{R} \mathbb{Z}_p$ and sets $\Lambda_0 = g^{\iota_0} \cdot (g^a)^{\omega_0}$ and $\Lambda_1 = g^{\iota_1} \cdot (g^a)^{\omega_1}$. The vector $\vec{f}$ is chosen so as to have a perfectly sound Groth-Sahai CRS $\mathbf{f} = (\vec{g_1}, \vec{g_2}, \vec{f})$.

During the game, the way to simulate the USERS oracle depends on the index $i \in \{1, \ldots, q_u\}$ of the query.

- If $i = i^\star$, $\mathcal{B}^{\text{inv-CDH}}$ defines $\mathsf{pk}_{i^\star} = (X_{i^\star,1}, X_{i^\star,2}, \Gamma_{i^\star,1}, \Gamma_{i^\star,2}) = \left(g_1^{x_{i^\star,1}} g^{z_{i^\star}}, g_2^{x_{i^\star,2}} g^{z_{i^\star}}, g^a, g^{\gamma_{i^\star,2}}\right)$ for random $x_{i^\star,1}, x_{i^\star,2}, z_{i^\star}, \gamma_{i^\star,2} \xleftarrow{R} \mathbb{Z}_p$. In other words, $\mathcal{B}^{\text{inv-CDH}}$ implicitly sets $\gamma_{i^\star,2} = a \in \mathbb{Z}_p$. Note that, although $\mathcal{B}^{\text{inv-CDH}}$ does not know $a = \log_g(\Gamma_{i^\star,1})$, it can perfectly simulate the USERS oracle as it can compute $\Gamma_{i^\star,0} = (g^a)^{\gamma_{i^\star,2}}$ which has to be verifiably encrypted during the JOIN protocol.

[10] Recall that any algorithm solving the inverse CDH problem with probability ε implies an algorithm (see [5] for details) solving the standard CDH problem with probability ε^2.

– If $i \neq i^\star$, $\mathcal{B}^{\text{inv-CDH}}$ defines $\mathsf{pk}_i = (X_{i,1}, X_{i,2}, \Gamma_{i,1}, \Gamma_{i,2}) = \big(g_1^{x_{i,1}} g^{z_i}, g_2^{x_{i,2}} g^{z_i}, g^{\gamma_{i,1}}, g^{\gamma_{i,2}}\big)$ for randomly chosen $x_{i,1}, x_{i,2}, z_i, \gamma_{i,1}, \gamma_{i,2} \overset{R}{\leftarrow} \mathbb{Z}_p$. In this case, $\mathcal{B}^{\text{inv-CDH}}$ entirely knows $\mathsf{sk}_i = (x_{i,1}, x_{i,2}, z_i, \gamma_{i,1}, \gamma_{i,2})$.

When $\mathcal{A}$ invokes the user corruption oracle CORR for an index $i \in \{1, \ldots, q_u\}$, $\mathcal{B}^{\text{inv-CDH}}$ halts and outputs a random bit if $i = i^\star$ as it must have failed to correctly guess the index of the target user. If $i \neq i^\star$, $\mathcal{B}^{\text{inv-CDH}}$ can consistently answer by returning sk_i which is at its disposal. We also note that $\mathcal{B}^{\text{inv-CDH}}$ can also perfectly answer $\mathcal{A}$'s queries to the decryption oracle $\mathsf{DEC}(.,.)$ as it always knows the decryption components $(x_{i,1}, x_{i,2}, z_i)$ of honest users private keys sk_i.

When it comes to simulate the $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ oracle, $\mathcal{B}^{\text{inv-CDH}}$ proceeds as follows. When $\mathcal{A}$ submits a triple (ψ, L, i), $\mathcal{B}^{\text{inv-CDH}}$ parses ψ as $\mathsf{VK}||(T_1, T_2, T_3, T_4)||\psi_{\mathsf{LY}}||\psi_{\mathsf{K}_1}||\psi_{\mathsf{K}_2}||\sigma$. With overwhelming probability, it can compute $T_{\delta,1} = (T_4/T_1^{\iota_0 \cdot \mathsf{VK} + \iota_1})^{1/(\omega_0 \cdot \mathsf{VK} + \omega_1)}$: indeed, (ω_0, ω_1) are independent of $\mathcal{A}$'s view, so that we can only have $\omega_0 \cdot \mathsf{VK} + \omega_1 = 0$ with negligible probability. In order to simulate the rest of τ, $\mathcal{B}^{\text{inv-CDH}}$ uses the observation that the CRS $(\vec{g_1}, \vec{g_2}, \vec{h}_{\mathsf{v}})$ is such that

$$\vec{h}_{\mathsf{v}} = \vec{h}_0 \odot \prod_{i=1}^{\ell} \vec{h}_i^{\mathsf{v}[i]} = \vec{g_1}^{J_1(\mathsf{v})} \odot \vec{g_2}^{J_2(\mathsf{v})} \odot (1, 1, g)^{J(\mathsf{v})}, \tag{19}$$

where $J_1(\mathsf{v}) = \zeta_{0,1} + \sum_{i=1}^{\ell} \zeta_{i,1} \mathsf{v}[i]$, $J_2(\mathsf{v}) = \zeta_{0,2} + \sum_{i=1}^{\ell} \zeta_{i,2} \mathsf{v}[i]$ and $J(\mathsf{v}) = \mu\kappa - \rho_0 - \sum_{i=1}^{\ell} \rho_i \mathsf{v}[i]$. We know that, if the event F_4 occurs, we have $J(\mathsf{v}_i) \neq 0$ for each $\mathsf{v}_i \in \{\mathsf{v}_1, \ldots, \mathsf{v}_{q_c}\}$ where $\mathsf{v}_1, \ldots, \mathsf{v}_{q_c}$ are the chameleon hash values involved in $\mathsf{CLAIM}/\mathsf{DISCLAIM}$ queries. This implies that, for each $i \in \{1, \ldots, q_c\}$, the triple $(J_1(\mathsf{v}_i), J_2(\mathsf{v}_i), J(\mathsf{v}_i))$ can be used as a simulation trapdoor to simulate a NIZK proof $(\vec{C}_{\Gamma_{-1}}, \vec{C}_{\mathcal{X}_\tau}, \pi_{\tau,1}, \pi_{\tau,2})$ of knowledge of g^{1/γ_1}.

When $\mathcal{A}$ outputs a claim/disclaimer $\tau^\star$ of its own, we know that the resulting chameleon hash value $\mathsf{v}^\star$ will satisfy $J(\mathsf{v}^\star) = 0$ if the event F_4 occurs. From (19), we see that $(\vec{g_1}, \vec{g_2}, \vec{h}_{\mathsf{v}})$ is an extractable Groth-Sahai CRS. Since the corresponding proofs $\pi^\star_{\tau^\star,1}, \pi^\star_{\tau^\star,2}$ are perfectly binding, $\mathcal{B}^{\text{inv-CDH}}$ can thus use $(\log_g(g_1), \log_g(g_2))$ as an extraction trapdoor to extract $g^{1/\gamma_1} = g^{1/a}$ from its commitment $\vec{C}_{\Gamma_{-1}}$.

When counting probabilities throughout the sequence of games, we find

$$\begin{aligned}\mathbf{Adv}^{\text{claiming-soundness}}(\mathcal{A}) \leq q_u \cdot \Big(&\mathbf{Adv}^{\text{DLIN}}(\lambda) + \mathbf{Adv}^{\text{CM-hash}}(\lambda) \\ &+ \big(27 \cdot q_c \cdot (\ell + 1) \cdot \big(\mathbf{Adv}^{\text{DLIN}}(\lambda) + (\mathbf{Adv}^{\text{CDH}}(\lambda))^2\big)\big)^{1/2}\Big)\end{aligned}$$

as an upper bound on the adversary's advantage. Since the CDH assumption is implied by the DLIN assumption, the announced result follows. □