Group Encryption: Non-Interactive Realization in the Standard Model

Julien Cathalo[1] *, Benoît Libert[1] **, and Moti Yung[2]

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

Abstract. Group encryption (GE) schemes, introduced at Asiacrypt'07, are an encryption analogue of group signatures with a number of interesting applications. They allow a sender to encrypt a message (in the CCA2 security sense) for some member of a PKI group concealing that member's identity (in a CCA2 security sense, as well); the sender is able to convince a verifier that, among other things, the ciphertext is valid and some anonymous certified group member will be able to decrypt the message. As in group signatures, an opening authority has the power of pinning down the receiver's identity. The initial GE construction uses interactive proofs as part of the design (which can be made non-interactive using the random oracle model) and the design of a fully non-interactive group encryption system is still an open problem. In this paper, we give the first GE scheme, which is a pure encryption scheme in the standard model, *i.e.*, a scheme where the ciphertext is a single message and proofs are non-interactive (and do not employ the random oracle heuristic). As a building block, we use a new public key certification scheme which incurs the smallest amount of interaction, as well.

Keywords. Group encryption, anonymity, provable security.

1 Introduction

Group encryption (GE) schemes, introduced by Kiayias, Tsiounis and Yung [29], are the encryption analogue of group signatures [16]. The latter primitives basically allow a group member to sign messages in the name of a group without revealing his identity. In a similar spirit, GE systems aim to hide the identity of a ciphertext's recipient and still guarantee that he belongs to a population of registered members in a group administered by a group manager (GM). A sender can generate an anonymous encryption of some plaintext m intended for a receiver holding a public key that was certified by the GM (message security and receiver anonymity being both in the CCA2 sense). The ciphertext is prepared while leaving an opening authority (OA) the ability to "open" the ciphertext (analogously to the opening operation in group signatures) and uncover the receiver's name. At the same time, the sender should be able to convince a verifier that (1) the ciphertext is a valid encryption under the public key of some group member holding a valid certificate; (2) if necessary, the opening authority will be able to find out who the receiver is; (3) (optionally) the plaintext is a witness satisfying some public relation.

Motivations. The GE primitive was motivated by various privacy applications such as anonymous trusted third parties or oblivious retriever storage. Many cryptographic protocols such as fair exchange, fair encryption or escrow encryption, involve trusted third parties that remain offline most of the time and are only involved to resolve problems. Group encryption allows one to verifiably encrypt some message to such a trusted third party while hiding his identity among a set of possible

* This author's research was supported by the Belgian Walloon Region project ALAWN (Programme Wist 2).
** This author acknowledges the Belgian National Fund for Scientific Research (F.R.S.-F.N.R.S.) for their financial support.

trustees. For instance, a user can encrypt a key (e.g., in an "international key escrow system") to his own national trusted representative without letting the ciphertext reveal the latter's identity, which could leak information on the user's citizenship. At the same time, everyone can be convinced that the ciphertext is heading for an authorized trustee.

Group encryption also finds applications in ubiquitous computing, where anonymous credentials must be transferred between peer devices belonging to the same group. Asynchronous transfers may require to involve an untrusted storage server to temporarily store encrypted credentials. In such a situation, GE schemes may be used to simultaneously guarantee that (1) the server retains properly encrypted valid credentials that it cannot read; (2) credentials have a legitimate anonymous retriever; (3) if necessary, an authority will be able to determine who the retriever is.

By combining cascaded group encryptions using multiple trustees and according to a sequence of identity discoveries and transfers, one can also implement group signatures where signers can flexibly specify how a set of trustees should operate to open their signatures.

Prior Works. Kiayias, Tsiounis and Yung (KTY) [29] formalized the concept of group encryption and provided a suitable security modeling. They presented a modular design of GE system and proved that, beyond zero-knowledge proofs, anonymous public key encryption schemes with CCA2 security, digital signatures, and equivocal commitments are necessary to realize the primitive. They also showed how to efficiently instantiate their general construction using Paillier's cryptosystem [35] (or, more precisely, a modification of the Camenisch-Shoup [13] variant of Paillier). While efficient, their scheme is not a single message encryption, since it requires the sender to interact with the verifier in a Σ-protocol to convince him that the aforementioned properties are satisfied. Interaction can be removed using the Fiat-Shamir paradigm [20] (and thus the random oracle model [4]), but only heuristic arguments [22] (see also [14]) are then possible in terms of security.

Independently, Qin *et al.* [36] considered a closely related primitive with non-interactive proofs and short ciphertexts. However, they avoid interaction by explicitly employing a random oracle and also rely on strong interactive assumptions. As we can see, none of these schemes is a truly non-interactive encryption scheme without the random oracle idealization.

Our Contribution. As already noted in various contexts such as anonymous credentials [2], rounds of interaction are expensive and even impossible at times as, in some applications, proofs should be verifiable by third parties that are not present when provers are available. In the setting of group encryption, this last concern is even more constraining as it requires the sender, who may be required to repeat proofs with many verifiers, to maintain a state and remember the random coins that he uses to encrypt *every* single ciphertext. In the frequent situation where many encryptions have to be generated using independent random coins, this becomes a definite bottleneck.

This paper solves the above problems and describes the first realization of group encryption which is a fully non-interactive encryption scheme with CCA2-security and anonymity in the standard model. In our scheme, senders do not need to maintain a state: thanks to the Groth-Sahai [27] non-interactive proof systems, the proof of a ciphertext can be generated once-and-for-all at the same time as the ciphertext itself. Furthermore, using suitable parameters and for a comparable security level, we can also shorten ciphertexts by a factor of 2 in comparison with the KTY scheme. As far as communication goes, the size of proofs allows decreasing by more than 75% the number of transmitted bits between the sender and the verifier.

Since our goal is to avoid interaction, we also design a joining protocol (*i.e.*, a protocol whereby the user effectively becomes a group member and gets his public key certified by the GM) which requires the smallest amount of interaction: as in the Kiayias-Yung group signature [30], only two messages have to be exchanged between the GM and the user and the latter need not to prove anything about his public key. In particular, rewinding is not necessary in security proofs and the join protocol can be safely executed in a concurrent environment, when many users want to register at the same time. The join protocol uses a non-interactive public key certification scheme where discrete-logarithm-type public keys can be signed as if they were ordinary messages (and without knowing the matching private key) while leaving the ability to efficiently prove knowledge of the certificate/public key using the Groth-Sahai techniques. To certify users without having to rewind[3] in security proofs, the KTY scheme uses groups of hidden order (and more precisely, Camenisch-Lysyanskaya signatures [12]). In public order groups, to the best of our knowledge, our construction is the first certification method that does not require any form of proof of knowledge of private keys. We believe it to be of independent interest as it can be used to construct group signatures (in the standard model) where the joining mechanism tolerates concurrency in the model of [30] without demanding more than two moves of interaction.

Organization. In section 2, we describe the intractability assumptions that we need and recall the KTY model of group encryption. Section 3 explains the building blocks of our construction and notably describes our certification scheme. Our GE system is depicted in section 4.

2 Background

In the paper, when S is a set, $x \xleftarrow{\$} S$ 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.

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 rely on an assumption introduced in [7] that allows constructing efficient non-interactive proofs as pointed out in [27].

Definition 1. *The* **Decision Linear Problem** *(DLIN) in $\mathbb{G}$, is to distinguish the distribution of linear tuples $D_1 = \{(g, g^a, g^b, g^{ac}, g^{bd}, g^{c+d}) | a, b, c, d \xleftarrow{\$} \mathbb{Z}_p^*\}$ from the distribution of random tuples $D_2 = \{(g, g^a, g^b, g^{ac}, g^{bd}, g^z) | a, b, c, d, z \xleftarrow{\$} \mathbb{Z}_p^*\}$. The* **Decision Linear Assumption** *is the intractability of DLIN for any PPT algorithm $\mathcal{D}$.*

This problem amounts to deciding whether vectors $\vec{g_1} = (g^a, 1, g)$, $\vec{g_2} = (1, g^b, g)$ and $\vec{g_3}$ are linearly dependent or not. We also consider a related computational problem which bears similarities with simultaneous pairing problems [26, 25].

Definition 2. *The* **Simultaneous Double Pairing problem** *(S2P) in $\mathbb{G}$ is, given a tuple of elements $(g_1, g_2, g_{1,c}, g_{2,d}) \in \mathbb{G}^4$, to find a non-trivial triple $(u, v, w) \in \mathbb{G}^3 \backslash \{(1_{\mathbb{G}}, 1_{\mathbb{G}}, 1_{\mathbb{G}})\}$ such that $e(g_1, u) = e(g_{1,c}, w)$ and $e(g_2, v) = e(g_{2,d}, w)$.*

[3] Although the simulator does not need to rewind proofs of knowledge in [29], users still have to interactively prove the validity of their public key.

Like the simultaneous triple pairing assumption [25], the hardness of this problem is implied by the DLIN assumption: given $(g, g_1, g_2, g_1^c, g_2^d, \eta \stackrel{?}{=} g^{c+d})$ any algorithm that, on input of (g_1, g_2, g_1^c, g_2^d), outputs a non-trivial (u, v, w) such that $e(g_1, u) = e(g_1^c, w)$, $e(g_2, v) = e(g_2^d, w)$ allows telling whether $\eta = g^{c+d}$ by testing if $e(g, u \cdot v) = e(\eta, w)$ (since $u = w^c$ and $v = w^d$).

We also use the Hidden Strong Diffie-Hellman (HSDH) assumption introduced in [10] as a strengthening of the Strong Diffie-Hellman assumption [6].

Definition 3. *The* **ℓ-Hidden Strong Diffie-Hellman problem** *(ℓ-HSDH) in $\mathbb{G}$ consists in, given $(g, \Omega = g^\omega, u) \stackrel{\$}{\leftarrow} \mathbb{G}^3$ and triples $(g^{1/(\omega+s_i)}, g^{c_i}, u^{c_i})$ with $c_1, \ldots, c_\ell \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$, finding another triple $(g^{1/(\omega+c)}, g^c, u^c)$ such that $c \neq c_i$ for $i = 1, \ldots, \ell$.*

We finally need the following variant of the Diffie-Hellman assumption.

Definition 4. *The* **Flexible Diffie-Hellman problem** *(FlexDH) is, given $(g, g^a, g^b) \in \mathbb{G}^3$, where $a, b \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$, to find a triple (C, C^a, C^{ab}) such that $C \neq 1_{\mathbb{G}}$.*

A potentially easier problem considered in [33] only requires to output (C, C^{ab}) on input of the same values. The latter problem was proved generically hard in prime order groups [33]. In bilinear groups, any algorithm solving either of these two problems would make it easy to recognize g^{abc} on input of (g, g^a, g^b, g^c), which is a problem suggested for the first time in [8, Section 8].

2.2 Model and Security Notions

Group encryption 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 $\mathsf{GE} = \big(\mathsf{SETUP}, \mathsf{JOIN}, \langle \mathcal{G}_r, \mathcal{R}, \mathsf{sample}_{\mathcal{R}} \rangle, \mathsf{ENC}, \mathsf{DEC}, \mathsf{OPEN}, \langle \mathcal{P}, \mathcal{V} \rangle\big)$ 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 to explicitly write it.

$\mathsf{JOIN} = (\mathsf{J}_{\mathsf{user}}, \mathsf{J}_{\mathsf{GM}})$ is an interactive protocol between the GM and the prospective user. As in [30], we will restrict this protocol to have minimal interaction and consist of only two messages: the first one 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 that makes the user's group membership effective. We do not require the user to prove knowledge of his private key sk or anything else about it. In our construction, valid keys will be publicly recognizable and users do not need to prove their validity. 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 $(x, w) \in \mathcal{R}$ (made 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 will be the case in our scheme). The testing procedure $\mathcal{R}(x, w)$ returns 1 whenever $(x, w) \in \mathcal{R}$. To encrypt a witness w such that $(x, w) \in \mathcal{R}$ for some public x, the sender fetches the pair $(\mathsf{pk}, \mathsf{cert}_{\mathsf{pk}})$ from $\mathsf{database}$ and runs the randomized 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 $\perp$. Finally, 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 public key pk.

The security model considers four properties termed correctness, message security, anonymity and soundness. In the definitions hereafter, we sometimes 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. The correctness property requires that the following experiment returns 1 with overwhelming probability.

Experiment $\mathbf{Expt}^{\mathrm{correctness}}(\lambda)$

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

Message Security. The message secrecy 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.

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. She has oracle access to the decryption function DEC of that receiver. At the challenge phase, she probes the challenge oracle for a label and a pair $(x, w) \in \mathcal{R}$ of her choice. After the challenge phase, she 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 [29], 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 5. *A GE 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 $\mathbf{Expt}^{\mathrm{sec}}_{\mathcal{A}}(\lambda)$

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

Anonymity. In anonymity attacks, the adversary controls the whole system but the opening authority and performs a kind of chosen-ciphertext attack on the encryption scheme of the OA. She registers two keys $\mathsf{pk}_0, \mathsf{pk}_1$ in $\mathsf{database}$ and, for a pair $(x, w) \in \mathcal{R}$ of her choosing, obtains an encryption of w under pk_b for some $b \in \{0,1\}$ chosen by the challenger. She is granted access to decryption oracles w.r.t. both keys $\mathsf{pk}_0, \mathsf{pk}_1$. In addition, she may invoke the following oracles:

- $\mathsf{CH}^b_{\mathsf{anon}}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_0, \mathsf{pk}_1, w, L)$: is a challenge oracle that is only queried once by the adversary. 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.
- $\mathsf{USER}(\mathsf{pk}_{\mathsf{GM}})$: is a stateful oracle simulating two executions of $\mathsf{J}_{\mathsf{user}}$ to introduce two honest users in the group. It uses a string keys where the outputs of the two executions are written.
- $\mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}}, .)$: is a stateless oracle that simulates the opening algorithm on behalf of the OA and, on input of a GE ciphertext, returns the receiver's public key.

Definition 6. *A GE 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 $\mathbf{Expt}^{\mathrm{anon}}_{\mathcal{A}}(\lambda)$

$\mathsf{param} \leftarrow \mathsf{SETUP}_{\mathsf{init}}(\lambda)$; $(\mathsf{pk}_{\mathsf{OA}}, \mathsf{sk}_{\mathsf{OA}}) \leftarrow \mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$;
$(\mathsf{aux}, \mathsf{pk}_{\mathsf{GM}}) \leftarrow \mathcal{A}(\mathsf{param}, \mathsf{pk}_{\mathsf{OA}})$; $\mathsf{aux} \leftarrow \mathcal{A}^{\mathsf{USER}(\mathsf{pk}_{\mathsf{GM}}), \mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}},.)}(\mathsf{aux})$;
If $\mathsf{keys} \neq (\mathsf{pk}_0, \mathsf{sk}_0, \mathsf{cert}_{\mathsf{pk}_0}, \mathsf{pk}_1, \mathsf{sk}_1, \mathsf{cert}_{\mathsf{pk}_1})(\mathsf{aux})$ *return* 0;
$(\mathsf{aux}, x, w, L, \mathsf{pk}_{\mathcal{R}}) \leftarrow \mathcal{A}^{\mathsf{OPEN}(\mathsf{sk}_{\mathsf{OA}},.), \mathsf{DEC}(\mathsf{sk}_0,.), \mathsf{DEC}(\mathsf{sk}_1,.)}(\mathsf{aux})$;
If $(x, w) \notin \mathcal{R}$ *return* 0;
$b \xleftarrow{\$} \{0,1\}$; $(\psi, coins_\psi) \leftarrow \mathsf{CH}^b_{\mathsf{anon}}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_0, \mathsf{pk}_1, w, L)$;
$b' \leftarrow \mathcal{A}^{\mathcal{P}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}_b, \mathsf{cert}_{\mathsf{pk}_b}, x, w, \psi, L, coins_\psi), \mathsf{OPEN}^{\neg\langle \psi, L\rangle}(\mathsf{sk}_{\mathsf{OA}},.), \mathsf{DEC}^{\neg\langle \psi, L\rangle}(\mathsf{sk}_0,.), \mathsf{DEC}^{\neg\langle \psi, L\rangle}(\mathsf{sk}_1,.))}(\mathsf{aux}, \psi)$;
If $b = b'$ *return* 1 *else return* 0;

As shown in [29], GE schemes satisfying the above notion necessarily subsume a key-private (a.k.a. receiver anonymous) [3, 28] cryptosystem.

Soundness. In a soundness attack, the adversary creates the group of receivers by interacting with the honest GM. Her goal is to produce a ciphertext ψ and a convincing proof that ψ is valid w.r.t. a relation $\mathcal{R}$ of her choice but either (1) the opening reveals a receiver's public key pk that does not belong to any group member; (2) the output pk of OPEN is not a valid public key (*i.e.*, $\mathsf{pk} \notin \mathcal{PK}$, where $\mathcal{PK}$ is the space of valid public keys); (3) the ciphertext C is not in the space $\mathcal{C}^{x, L, \mathsf{pk}_{\mathcal{R}}, \mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}}$ of valid ciphertexts. 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 simulates J_{GM}. This oracle maintains a repository $\mathsf{database}$ where registered public keys and their certificates are stored.

Definition 7. *A GE scheme is sound if, for any PPT adversary $\mathcal{A}$, the experiment below returns* 1 *with negligible probability.*

$$
\begin{array}{l}
\text{Experiment } \mathbf{Expt}_{\mathcal{A}}^{\text{soundness}}(\lambda) \\
\quad \mathsf{param} \leftarrow \mathsf{SETUP_{init}}(\lambda);\ (\mathsf{pk_{OA}}, \mathsf{sk_{OA}}) \leftarrow \mathsf{SETUP_{OA}}(\mathsf{param}); \\
\quad (\mathsf{pk_{GM}}, \mathsf{sk_{GM}}) \leftarrow \mathsf{SETUP_{GM}}(\mathsf{param}); \\
\quad (\mathsf{pk}_{\mathcal{R}}, x, \psi, \pi_\psi, L, \mathsf{aux}) \leftarrow \mathcal{A}^{\mathsf{REG}(\mathsf{sk_{GM}},\cdot)}(\mathsf{param}, \mathsf{pk_{GM}}, \mathsf{pk_{OA}}, \mathsf{sk_{OA}}); \\
\quad \textit{If } \mathcal{V}(\psi, L, \pi_\psi, \mathsf{pk_{GM}}, \mathsf{pk_{OA}}) = 0 \textit{ return } 0; \\
\quad \mathsf{pk} \leftarrow \mathsf{OPEN}(\mathsf{sk_{OA}}, \psi, L); \\
\quad \textit{If } \big((\mathsf{pk} \notin \mathsf{database}) \vee (\mathsf{pk} \notin \mathcal{PK}) \vee (\psi \notin \mathcal{C}^{x, L, \mathsf{pk}_{\mathcal{R}}, \mathsf{pk_{GM}}, \mathsf{pk_{OA}}, \mathsf{pk}})\big) \\
\qquad \textit{then return } 1 \textit{ else return } 0;
\end{array}
$$

2.3 Groth-Sahai Proof Systems

In the following notations, for equal-dimension vectors $\vec{A}$ and $\vec{B}$ containing group elements, $\vec{A} \odot \vec{B}$ stands for their component-wise product.

When based on the DLIN assumption, the Groth-Sahai (GS) proof systems [27] use a common reference string comprising 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 $X \in \mathbb{G}$, one sets $\vec{C} = (1, 1, X) \odot \vec{g_1}^{\,r} \odot \vec{g_2}^{\,s} \odot \vec{g_3}^{\,t}$ with $r, s, t \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$. When the proof system is prepared to give perfectly sound proofs, $\vec{g_3}$ is set as $\vec{g_3} = \vec{g_1}^{\,\xi_1} \odot \vec{g_2}^{\,\xi_2}$ with $\xi_1, \xi_2 \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$. Commitments $\vec{C} = (g_1^{r+\xi_1 t}, g_2^{s+\xi_2 t}, X \cdot g^{r+s+t(\xi_1+\xi_2)})$ are then Boneh-Boyen-Shacham (BBS) ciphertexts that can be decrypted using $\alpha_1 = \log_g(g_1)$, $\alpha_2 = \log_g(g_2)$. In the witness indistinguishability (WI) setting, vectors $\vec{g_1}, \vec{g_2}, \vec{g_3}$ are linearly independent and $\vec{C}$ is a perfectly hiding commitment. Under the DLIN assumption, the two kinds of CRS are indistinguishable.

To commit to an exponent $x \in \mathbb{Z}_p$, one computes $\vec{C} = \vec{\varphi}^{\,x} \odot \vec{g_1}^{\,r} \odot \vec{g_2}^{\,s}$, with $r, s \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$, using a CRS comprising vectors $\vec{\varphi}, \vec{g_1}, \vec{g_2}$. In the soundness setting $\vec{\varphi}, \vec{g_1}, \vec{g_2}$ are linearly independent vectors (typically $\vec{\varphi} = \vec{g_3} \odot (1, 1, g)$ where $\vec{\varphi} = \vec{g_1}^{\,\xi_1} \odot \vec{g_2}^{\,\xi_2}$) whereas, in the WI setting, choosing $\vec{\varphi} = \vec{g_1}^{\,\xi_1} \odot \vec{g_2}^{\,\xi_2}$ gives a perfectly hiding commitment since $\vec{C}$ is always a BBS encryption of $1_{\mathbb{G}}$.

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

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

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

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

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

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 $\gamma_{ij} \in \mathbb{G}$, for $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$.

Multi-exponentiation equations admit zero-knowledge proofs at no additional cost. On a simulated CRS (prepared for the WI setting), a trapdoor makes it is 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 but usually come at some expense. In the paper, we only resort to such NIZK simulators in one occasion.

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.

3 Building Blocks

Our certification scheme uses a trapdoor commitment to group elements as an important ingredient to dispense with proofs of knowledge of users' private keys.

3.1 A Trapdoor Commitment to Group Elements

We need a trapdoor commitment scheme that allows committing to elements of a group $\mathbb{G}$ where bilinear map arguments are taken. Commitments will have to be themselves elements of $\mathbb{G}$, which prevents us from using Groth's scheme [25] where commitments lie in the range $\mathbb{G}_T$ of the pairing.

Such commitments can be obtained using the perfectly hiding Groth-Sahai commitment based on the linear assumption recalled in section 2.3. This commitment uses a common reference string describing a prime order group $\mathbb{G}$ and a generator $f \in \mathbb{G}$. The commitment key consists of vectors $(\vec{f_1}, \vec{f_2}, \vec{f_3})$ chosen as $\vec{f_1} = (f_1, 1, f)$, $\vec{f_2} = (1, f_2, f)$ and $\vec{f_3} = \vec{f_1}^{\xi_1} \odot \vec{f_2}^{\xi_2} \odot (1, 1, f)^{\xi_3}$, with $f_1, f_2 \xleftarrow{\$} \mathbb{G}$, $\xi_1, \xi_2, \xi_3 \xleftarrow{\$} \mathbb{Z}_p^*$. To commit to a group element $X \in \mathbb{G}$, the sender picks $\phi_1, \phi_2, \phi_3 \xleftarrow{\$} \mathbb{Z}_p^*$ and sets $\vec{C}_X = (1, 1, X) \odot \vec{f_1}^{\phi_1} \odot \vec{f_2}^{\phi_2} \odot \vec{f_3}^{\phi_3}$, which, if $\vec{f_3}$ is parsed as $(f_{3,1}, f_{3,2}, f_{3,3})$, can be written $\vec{C}_X = (f_1^{\phi_1} \cdot f_{3,1}^{\phi_3}, f_2^{\phi_2} \cdot f_{3,2}^{\phi_3}, X \cdot f^{\phi_1+\phi_2} \cdot f_{3,3}^{\phi_3})$. Due to the use of GS proofs, commitment openings need to only consist of group elements (and no scalar). To open $\vec{C}_X = (C_1, C_2, C_3)$, the sender reveals $(D_1, D_2, D_3) = (f^{\phi_1}, f^{\phi_2}, f^{\phi_3})$ and X. The receiver is convinced that the committed value was X by checking that

$$\begin{cases} e(C_1, f) = e(f_1, D_1) \cdot e(f_{3,1}, D_3) \\ e(C_2, f) = e(f_2, D_2) \cdot e(f_{3,2}, D_3) \\ e(C_3, f) = e(X \cdot D_1 \cdot D_2, f) \cdot e(f_{3,3}, D_3). \end{cases}$$

If a cheating sender can come up with distinct openings of $\vec{C}_X$, we can easily solve a $S2P$ instance $(g_1, g_2, g_{1,c}, g_{2,d})$. Namely, the commitment key is set as $(f_1, f_2, f_{3,1}, f_{3,2}) = (g_1, g_2, g_{1,c}, g_{2,d})$ and $f, f_{3,3}$ are chosen at random. When the adversary outputs $(X, (D_1, D_2, D_3))$ and $(X', (D_1', D_2', D_3'))$, we must simultaneously have $e(f_1, D_1/D_1') = e(f_{3,1}, D_3'/D_3)$, $e(f_2, D_2/D_2') = e(f_{3,2}, D_3'/D_3)$ and $e((XD_1D_2)/(X'D_1'D_2'), f) = e(f_{3,3}, D_3'/D_3)$. Hence, a solution to the S2P instance is obtained by setting $(u, v, w) = (D_1/D_1', D_2/D_2', D_3'/D_3)$, which is a non-trivial triple as long as $X' \neq X$.

We also observe that, using the trapdoor (ξ_1, ξ_2, ξ_3), the receiver can equivocate commitments. Given a commitment $\vec{C}_X$ and its opening $(X, (D_1, D_2, D_3))$, one can trapdoor open $\vec{C}_X$ to any other $X' \in \mathbb{G}$ (and without knowing $\log_g(X')$) by computing

$$D_1' = D_1 \cdot (X'/X)^{\xi_1/\xi_3}, \qquad D_2' = D_2 \cdot (X'/X)^{\xi_2/\xi_3}, \qquad D_3' = (X/X')^{1/\xi_3} \cdot D_3.$$

3.2 A Public Key Certification Scheme

We use a primitive that we call *non-interactive certification scheme*, which can be viewed as a signature scheme that only allows signing public keys from a specific public key space $\mathcal{PK}$. These keys should be signed while retaining algebraic properties that make it possible to prove knowledge of a public key and its corresponding certificate in an efficient way. In particular, signing hashed public keys is proscribed. In the interactive setting, several papers (e.g., [5, 24]) describe efficient interactive protocols where a public key is jointly generated by a user and a certification authority in such a way that the user eventually obtains a certified public key and no one else learns the underlying private key. In this paper, we aim at minimizing the amount of interaction and let users generate their public key entirely on their own before requesting their certification. Ideally, we would like to be able to sign public keys without even requiring users to prove knowledge of their private key and, in particular, without having to first rewind a proof of knowledge so as to extract the user's private key in the security proof.

A certification scheme consists of algorithms ($\mathsf{Setup}, \mathsf{Certify}, \mathsf{CertVerify}$). The first one is run by a certification authority (CA) that, on input of global parameters cp, generates a key pair $(SK, PK) \leftarrow \mathsf{Setup}(\mathsf{cp})$. On input of cp, SK and a user's public key pk, $\mathsf{Certify}$ generates a certificate $\mathsf{cert}_{\mathsf{pk}}$. The procedure Verify takes as input cp, PK, pk and $\mathsf{cert}_{\mathsf{pk}}$ and outputs either 0 or 1.

Correctness mandates that $\mathsf{CertVerify}(\mathsf{cp}, PK, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}) = 1$ when $\mathsf{cert}_{\mathsf{pk}} \leftarrow \mathsf{Certify}(\mathsf{cp}, SK, \mathsf{pk})$. The (strong) unforgeability [1] requirement is the same as in signature schemes. The adversary is supplied with a CA's public key PK and access to a certification oracle $\mathsf{Certify}(SK, .)$ that can be queried for arbitrary public keys $\mathsf{pk} \in \mathcal{PK}$. Her goal is to produce a new pair $(\mathsf{pk}^*, \mathsf{cert}^*_{\mathsf{pk}^*})$ (*i.e.*, if pk^* was queried to $\mathsf{Certify}(SK, .)$, the output must have been different from $\mathsf{cert}^*_{\mathsf{pk}^*}$).

In the description, we assume common public parameters cp consisting of of bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$, for a security parameter λ, and a generator $g \stackrel{\$}{\leftarrow} \mathbb{G}$. We also assume that certified public keys always consist of a fixed number n of group elements (*i.e.*, $\mathcal{PK} = \mathbb{G}^n$).

Intuition. The scheme borrows from the Boyen-Waters group signature [10] in the use of the Hidden Strong Diffie-Hellman assumption. A simplified version of this scheme involves a CA that holds a public key $PK = (\Omega = g^\omega, A = (g, g)^\alpha, u, u_0, u_1 = g^{\beta_1}, \ldots, u_n = g^{\beta_n})$, for private elements $SK = (\omega, \alpha, \beta_1, \ldots, \beta_n)$, where n denotes the number of groups elements that certified public keys consist of. To certify a public key $\mathsf{pk} = (X_1 = g^{x_1}, \ldots, X_n = g^{x_n})$, the CA chooses an exponent $c_{\mathsf{ID}} \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$ and computes $S_1 = (g^\alpha)^{1/(\omega + c_{\mathsf{ID}})}$, $S_2 = g^{c_{\mathsf{ID}}}$, $S_3 = u^{c_{\mathsf{ID}}}$, $S_4 = (u_0 \cdot \prod_{i=1}^n X_i^{\beta_i})^{c_{\mathsf{ID}}}$ and $S_5 = (S_{5,1}, \ldots, S_{5,n}) = (X_1^{c_{\mathsf{ID}}}, \ldots, X_n^{c_{\mathsf{ID}}})$. Verification then checks whether $e(S_1, \Omega \cdot S_2) = A$ and $e(S_2, u) = e(g, S_3)$ as in [10]. It must also be checked that $e(S_4, g) = e(u_0, S_2) \cdot \prod_{i=1}^n e(u_i, S_{5,i})$ and $e(S_{5,i}, g) = e(X_i, S_2)$ for $i = 1, \ldots, n$.

The security of this simplified scheme can only be proven if, when answering certification queries, the simulator can control the private keys $(x_1, \ldots, x_n)$ and force them to be random values of its choice. To allow the simulator to sign arbitrary public keys without knowing the private keys, we modify the scheme so that the CA rather signs commitments (calculated as in the trapdoor commitment of section 3.1) to public key elements $X_1, \ldots, X_n$. In the security proof, the simulator first generates a signature on n fake commitments $\vec{C}_i = (C_{i,1}, C_{i,2}, C_{i,3})$ that are all generated in such a way that it knows $\log_g(C_{i,j})$ for $i = 1, \ldots, n$ and $j = 1, 2, 3$. Using the trapdoor of the commitment scheme, it can then open $\vec{C}_i$ to any arbitrary $X_i \in \mathbb{G}$ without knowing $\log_g(X_i)$.

This use of the trapdoor commitment is reminiscent of a technique (notably used in [18]) to construct signature schemes in the standard model using chameleon hash functions [32]: the simulator first signs messages of its choice using a basic signature scheme and then "equivocates" the chameleon hashes to make them correspond to adversarially-chosen messages.

Setup(cp)**:** given common public parameters $\mathsf{cp} = \{g, \mathbb{G}, \mathbb{G}_T\}$, select $u, u_0 \xleftarrow{\$} \mathbb{G}$, $\alpha, \omega \xleftarrow{\$} \mathbb{Z}_p^*$ and set $A = e(g,g)^\alpha$, $\Omega = g^\omega$. Pick $\beta_{i,1}, \beta_{i,2}, \beta_{i,3} \xleftarrow{\$} \mathbb{Z}_p^*$ and set $\overline{u}_i = (u_{i,1}, u_{i,2}, u_{i,3}) = (g^{\beta_{i,1}}, g^{\beta_{i,2}}, g^{\beta_{i,3}})$ for $i = 1, \ldots, n$. Choose $f, f_1, f_2, f_{3,1}, f_{3,2}, f_{3,3} \xleftarrow{\$} \mathbb{G}$ that define a commitment key consisting of vectors $\vec{f_1} = (f_1, 1, f)$, $\vec{f_2} = (1, f_2, f)$ and $\vec{f_3} = (f_{3,1}, f_{3,2}, f_{3,3})$. Define the private key to be $SK = \big(\alpha, \omega, \{\overline{\beta}_i = (\beta_{i,1}, \beta_{i,2}, \beta_{i,3})\}_{i=1,\ldots,n}\big)$ and the public key as

$$PK = \Big(\mathbf{f} = (\vec{f_1}, \vec{f_2}, \vec{f_3}),\ A = e(g,g)^\alpha,\ \Omega = g^\omega,\ u,\ u_0,\ \{\overline{u}_i\}_{i=1,\ldots,n}\Big).$$

Certify($\mathsf{cp}, SK, \mathsf{pk}$)**:** parse SK as $\big(\alpha, \omega, \{\overline{\beta}_i\}_{i=1,\ldots,n}\big)$, pk as $(X_1, \ldots, X_n)$ and do the following.

1. For each $i \in \{1, \ldots, n\}$, pick $\phi_{i,1}, \phi_{i,2}, \phi_{i,3} \xleftarrow{\$} \mathbb{Z}_p^*$ and compute a commitment

$$C_i = (C_{i,1}, C_{i,2}, C_{i,3}) = (f_1^{\phi_{i,1}} \cdot f_{3,1}^{\phi_{i,3}},\ f_2^{\phi_{i,2}} \cdot f_{3,2}^{\phi_{i,3}},\ X_i \cdot f^{\phi_{i,1}+\phi_{i,2}} \cdot f_{3,3}^{\phi_{i,3}})$$

   and the matching de-commitment $(D_{i,1}, D_{i,2}, D_{i,3}) = (f^{\phi_{i,1}}, f^{\phi_{i,2}}, f^{\phi_{i,3}})$.

2. Choose $c_{\mathsf{ID}} \xleftarrow{\$} \mathbb{Z}_p^*$ and compute $S_1 = (g^\alpha)^{1/(\omega + c_{\mathsf{ID}})}$, $S_2 = g^{c_{\mathsf{ID}}}$, $S_3 = u^{c_{\mathsf{ID}}}$ as well as

$$S_4 = \Big(u_0 \cdot \prod_{i=1}^{n} (C_{i,1}^{\beta_{i,1}} \cdot C_{i,2}^{\beta_{i,2}} \cdot C_{i,3}^{\beta_{i,3}})\Big)^{c_{\mathsf{ID}}}$$
$$S_5 = \{(S_{5,i,1}, S_{5,i,2}, S_{5,i,3})\}_{i=1,\ldots,n} = \{(C_{i,1}^{c_{\mathsf{ID}}}, C_{i,2}^{c_{\mathsf{ID}}}, C_{i,3}^{c_{\mathsf{ID}}})\}_{i=1,\ldots,n}$$

Return $\mathsf{cert}_{\mathsf{pk}} = \Big(\{(C_{i,1}, C_{i,2}, C_{i,3}), (D_{i,1}, D_{i,2}, D_{i,3})\}_{i=1,\ldots,n}, S_1, S_2, S_3, S_4, S_5\Big)$.

CertVerify($\mathsf{cp}, PK, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}$)**:** parse pk as $(X_1, \ldots, X_n)$ and $\mathsf{cert}_{\mathsf{pk}}$ as above. Return 1 if, for indices $i = 1, \ldots, n$, it holds that $X_i \in \mathbb{G}$ and

$$e(C_{i,1}, f) = e(f_1, D_{i,1}) \cdot e(f_{3,1}, D_{i,3}) \tag{1}$$
$$e(C_{i,2}, f) = e(f_2, D_{i,2}) \cdot e(f_{3,2}, D_{i,3}) \tag{2}$$
$$e(C_{i,3}, f) = e(X_i \cdot D_{i,1} \cdot D_{i,2}, f) \cdot e(f_{3,3}, D_{i,3}), \tag{3}$$

and if the following checks are also satisfied. Otherwise, return 0.

$$e(S_1, \Omega \cdot S_2) = A \tag{4}$$
$$e(S_2, u) = e(g, S_3) \tag{5}$$
$$e(S_4, g) = e(u_0, S_2) \cdot \prod_{i=1}^{n} \big(e(u_{i,1}, S_{5,i,1}) \cdot e(u_{i,2}, S_{5,i,2}) \cdot e(u_{i,3}, S_{5,i,3})\big), \tag{6}$$
$$e(S_{5,i,j}, g) = e(C_{i,j}, S_2) \qquad \text{for } i = 1, \ldots, n,\ j = 1, 2, 3 \tag{7}$$

A certificate comprises $9n+4$ group elements. It would be interesting to avoid this linear dependency on n without destroying the algebraic properties that render the scheme compatible with the Groth-Sahai techniques.

Regarding the security of the scheme, the following theorem is proved in appendix A.

Theorem 1. *The scheme is a secure non-interactive certification system if the HSDH, FlexDH and S2P problems are all hard in* $\mathbb{G}$.

We believe that the above certification scheme is of interest in its own right. For instance, it can be used to construct non-frameable group signatures that are secure in the concurrent join model of [30] without resorting to random oracles. To the best of our knowledge, the Kiayias-Yung construction [30] has remained the only scalable group signature where joining supports concurrency at both ends while requiring the smallest amount of interaction. In the standard model, our certification scheme thus appears to provide the first[4] way to achieve the same result. In this case, we have $n = 1$ (since prospective group members only need to certify one group element if non-frameability is ensured by signing messages using Boneh-Boyen signatures [6] in the same way as in Groth's group signature [24]) so that membership certificates comprise 13 group elements and their shape is fully compatible with GS proofs.

3.3 Public Key Encryption Schemes Based on the Linear Problem

We need cryptosystems based on the DLIN assumption. The first one is Shacham's variant [37] of Cramer-Shoup [17] and, since it is key-private [3], we use it to encrypt witnesses. We also use Kiltz's tag-based encryption (TBE) scheme [31], where the validity of ciphertexts is publicly verifiable, to encrypt receivers' public keys under the public key of the opening authority.

Shacham's Linear Cramer-Shoup. If we assume public generators g_1, g_2, g that are parts of public parameters, each receiver's public key is made of $n = 6$ group elements

$$\begin{array}{lll} X_1 = g_1^{x_1} g^x & X_3 = g_1^{x_3} g^y & X_5 = g_1^{x_5} g^z \\ X_2 = g_2^{x_2} g^x & X_4 = g_2^{x_4} g^y & X_6 = g_2^{x_6} g^z. \end{array}$$

To encrypt a plaintext $m \in \mathbb{G}$ under the label L, the sender picks $r, s \xleftarrow{\$} \mathbb{Z}_p^*$ and computes

$$\psi_{\mathsf{CS}} = (U_1, U_2, U_3, U_4, U_5) = \Big(g_1^r,\ g_2^s,\ g^{r+s},\ m \cdot X_5^r X_6^s,\ (X_1 X_3^\alpha)^r \cdot (X_2 X_4^\alpha)^s\Big),$$

where $\alpha = H(U_1, U_2, U_3, U_4, L) \in \mathbb{Z}_p^*$ is a collision-resistant hash[5]. Given (ψ_{CS}, L), the receiver computes α. He returns $\perp$ if $U_5 \neq U_1^{x_1+\alpha x_3} U_2^{x_2+\alpha x_4} U_3^{x+\alpha y}$ and $m = U_4/(U_1^{x_5} U_2^{x_6} U_3^z)$ otherwise.

Kiltz's Tag-Based Encryption Scheme. In [31], Kiltz described a TBE scheme based on the same assumption. The public key is $(Y_1, Y_2, Y_3, Y_4) = (g^{y_1}, g^{y_2}, g^{y_3}, g^{y_4})$ if $g \in \mathbb{G}$ is part of public parameters. To encrypt $m \in \mathbb{G}$ under a tag $t \in \mathbb{Z}_p^*$, the sender picks $w_1, w_2 \xleftarrow{\$} \mathbb{Z}_p^*$ and computes

$$\psi_{\mathsf{K}} = (V_1, V_2, V_3, V_4, V_5) = \Big(Y_1^{w_1},\ Y_2^{w_2},\ (g^t Y_3)^{w_1},\ (g^t Y_4)^{w_2},\ m \cdot g^{w_1+w_2}\Big)$$

[4] Non-frameable group signatures described in [19, 9] achieve concurrent security by having the prospective user generate an extractable commitment to some secret exponent (which the simulator can extract without rewinding using the trapdoor of the commitment) and prove that the committed value is the discrete log. of a public value. In the standard model, this technique requires interaction and the proof should be simulatable in zero-knowledge when proving security against framing attacks. Another technique [21] requires users to prove knowledge of their secret exponent using Groth-Sahai non-interactive proofs. It is nevertheless space-demanding as each bit of committed exponent requires its own extractable GS commitment.

[5] The proof of CCA2-security [17, 37] only requires a universal one-way hash function (UOWHF) [34] but collision-resistance is required by the proof of key-privacy in [3].

To decrypt such a ciphertext ψ_{K}, the receiver checks that $V_3 = V_1^{(t+y_3)/y_1}$, $V_4 = V_2^{(t+y_4)/y_2}$. If so, it outputs the plaintext $m = V_5/(V_1^{1/y_1} V_2^{1/y_2})$. Unlike ψ_{CS} in the linear Cramer-Shoup system, the well-formedness of ψ_{K} is publicly verifiable in bilinear groups. The Canetti-Halevi-Katz [15] paradigm turns this scheme into a full-fledged CCA2 scheme by deriving the tag t from the verification key VK of a one-time signature, the private key SK of which is used to sign $(V_1, V_2, V_3, V_4, V_5)$.

4 A GE Scheme with Non-Interactive Proofs

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 $\mathsf{sk}_{\mathcal{R}} = \varepsilon$.

The construction slightly departs from the modular design of [29] in that commitments to the receiver's public key and certificate are part of the proof (instead of the ciphertext), which simplifies the proof of message-security. The security of the scheme eventually relies on the HSDH, FlexDH and DLIN assumptions. All security proofs are available in appendix B.

$\mathsf{SETUP}_{\mathsf{init}}(\lambda)$: choose bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of order $p > 2^\lambda$, $g \stackrel{\$}{\leftarrow} \mathbb{G}$ and $g_1 = g^{\alpha_1}$, $g_2 = g^{\alpha_2}$ with $\alpha_1, \alpha_2 \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$. Define $\vec{g_1} = (g_1, 1, g)$, $\vec{g_2} = (1, g_2, g)$ and $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$ with $\xi_1, \xi_2 \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$, which form a CRS $\mathbf{g} = (\vec{g_1}, \vec{g_2}, \vec{g_3})$ for the perfect soundness setting. Select a strongly unforgeable (as defined in [1]) one time signature scheme $\Sigma = (\mathcal{G}, \mathcal{S}, \mathcal{V})$ and a random member $H : \{0,1\}^* \to \mathbb{Z}_p$ of a collision-resistant hash family. Public parameters consists of $\mathsf{param} = \{\lambda, \mathbb{G}, \mathbb{G}_T, g, \mathbf{g}, \Sigma, H\}$.

$\mathsf{SETUP}_{\mathsf{GM}}(\mathsf{param})$: runs the setup algorithm of the certification scheme described in section 3.2 with $n = 6$. The obtained public key consists of $\mathsf{pk}_{\mathsf{GM}} = \Big(\mathbf{f},\ A = e(g, g)^\alpha,\ \Omega = g^\omega,\ u,\ u_0,\ \{\overline{u}_i\}_{i=1,\dots,6}\Big)$ and the matching private key is $\mathsf{sk}_{\mathsf{GM}} = \big(\alpha, \omega, \{\overline{\beta}_i = (\beta_{i,1}, \beta_{i,2}, \beta_{i,3})\}_{i=1,\dots,6}\big)$.

$\mathsf{SETUP}_{\mathsf{OA}}(\mathsf{param})$: 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 tag-based encryption scheme [31], and the corresponding private key as $\mathsf{sk}_{\mathsf{OA}} = (y_1, y_2, y_3, y_4)$.

JOIN: the user sends a linear Cramer-Shoup public key $\mathsf{pk} = (X_1, \dots, X_6) \in \mathbb{G}^6$ to the GM and obtains a certificate

$$\mathsf{cert}_{\mathsf{pk}} = \big(\{(C_{i,1}, C_{i,2}, C_{i,3}), (D_{i,1}, D_{i,2}, D_{i,3})\}_{i=1,\dots,6}, S_1, S_2, S_3, S_4, S_5\big).$$

$\mathsf{ENC}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, W, L)$: to encrypt $W \in \mathbb{G}$ such that $((X, Y), W) \in \mathcal{R}$ (for public elements $X, Y \in \mathbb{G}$), parse $\mathsf{pk}_{\mathsf{GM}}$, $\mathsf{pk}_{\mathsf{OA}}$ and pk as above and do the following.

1. Generate a one-time signature key pair $(\mathsf{SK}, \mathsf{VK}) \leftarrow \mathcal{G}(\lambda)$.
2. Choose $r, s \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$ and compute a linear CS encryption of W, the result of which is denoted by ψ_{CS}, under the label $L_1 = L||\mathsf{VK}$ as per section 3.3 (and using the collision-resistant hash function specified by param).
3. For $i = 1, \dots, 6$, choose $w_{i,1}, w_{i,2} \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$ and encrypt X_i under $\mathsf{pk}_{\mathsf{OA}}$ using Kiltz's TBE with the tag VK as described in section 3.3 . Let ψ_{K_i} be the ciphertexts.
4. Set the GE ciphertext ψ as $\psi = \mathsf{VK}||\psi_{\mathsf{CS}}||\psi_{\mathsf{K}_1}|| \cdots ||\psi_{\mathsf{K}_6}||\sigma$ where σ is a one-time signature obtained as $\sigma = \mathcal{S}(\mathsf{SK}, (\psi_{\mathsf{CS}}||\psi_{\mathsf{K}_1}|| \cdots ||\psi_{\mathsf{K}_6}||L))$.

Return (ψ, L) and $coins_\psi$ consist of $\{(w_{i,1}, w_{i,2})\}_{i=1,\dots,6}$, (r, s). If the one-time signature of [23] is used, VK and σ take 3 and 2 group elements, respectively, so that ψ comprises 40 group elements.

$\mathcal{P}(\mathsf{pk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{pk}, \mathsf{cert}_{\mathsf{pk}}, (X,Y), W, \psi, L, coins_\psi)$: parse $\mathsf{pk}_{\mathsf{GM}}$, $\mathsf{pk}_{\mathsf{OA}}$, pk and ψ as above. Conduct the following steps.

1. Generate commitments (as explained in section 2.3) to the $9n+4 = 58$ group elements that $\mathsf{cert}_{\mathsf{pk}}$ consists of. The resulting overall commitment $com_{\mathsf{cert}_{\mathsf{pk}}}$ contains 184 group elements.
2. Generate GS commitments to the public key elements $\mathsf{pk} = (X_1, \ldots, X_6)$ and obtain the set $com_{\mathsf{pk}} = \{com_{X_i}\}_{i=1,\ldots,6}$, which consists of 18 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}. For each $i = 1, \ldots, 6$, relations (1)-(3) cost 9 elements to prove (and thus 54 elements altogether). The quadratic equation (4) takes 9 elements and linear ones (5)-(6) both require 3 elements. Finally, (7) is a set of 18 linear equations which demand 54 elements altogether. The whole proof $\pi_{\mathsf{cert}_{\mathsf{pk}}}$ thus takes 123 group elements.
4. For $i = 1, \ldots, 6$, generate a NIZK proof $\pi_{eq\text{-}key,i}$ that com_{X_i} (which is part of com_{pk}) and ψ_{K_i} are encryptions of the same X_i. If ψ_{K_i} comprises

$$(V_{i,1}, V_{i,2}, V_{i,5}) = (Y_1^{w_{i,1}}, Y_2^{w_{i,2}}, X_i \cdot g^{w_{i,1}+w_{i,2}})$$

and com_{X_i} is parsed as $(c_{X_{i1}}, c_{X_{i2}}, c_{X_{i3}}) = (g_1^{\theta_{i1}} \cdot g_{3,1}^{\theta_{i3}},\ g_2^{\theta_{i2}} \cdot g_{3,2}^{\theta_{i3}},\ X_i \cdot g^{\theta_{i1}+\theta_{i2}} \cdot g_{3,3}^{\theta_{i3}})$, where $w_{i,1}, w_{i,2} \in coins_\psi$, $\theta_{i1}, \theta_{i2}, \theta_{i3} \in \mathbb{Z}_p^*$ and $\vec{g_3} = (g_{3,1}, g_{3,2}, g_{3,3})$, this amounts to prove knowledge of values $w_{i,1}, w_{i,2}, \theta_{i1}, \theta_{i2}, \theta_{i3} \in \mathbb{Z}_p^*$ such that

$$\Big(\frac{V_{i,1}}{c_{X_{i1}}}, \frac{V_{i,2}}{c_{X_{i2}}}, \frac{V_{i,3}}{c_{X_{i3}}}\Big) = \big(Y_1^{w_{i,1}} \cdot g_1^{-\theta_{i1}} \cdot g_{3,1}^{-\theta_{i3}},\ Y_2^{w_{i,2}} \cdot g_2^{-\theta_{i2}} \cdot g_{3,2}^{-\theta_{i3}},\ g^{w_{i,1}+w_{i,2}-\theta_{i1}-\theta_{i2}} \cdot g_{3,3}^{-\theta_{i3}}\big).$$

Committing to exponents $w_{i,1}, w_{i,2}, \theta_{i1}, \theta_{i2}, \theta_{i3}$ introduces 90 group elements whereas the above relations only require two elements each. Overall, proof elements $\pi_{eq\text{-}key,1}, \ldots, \pi_{eq\text{-}key,6}$ incur 126 elements.
5. Generate a NIZK proof $\pi_{val\text{-}enc}$ that $\psi_{\mathsf{CS}} = (U_1, U_2, U_3, U_4, U_5)$ is a valid CS encryption. This requires to commit to underlying encryption exponents $r, s \in coins_\psi$ and prove that $U_1 = g_1^r$, $U_2 = g_2^s$, $U_3 = g^{r+s}$ (which only takes 3 times 2 elements as base elements are public) and $U_5 = (X_1 X_3^\alpha)^r \cdot (X_2 X_4^\alpha)^s$ (which takes 9 elements since base elements are themselves variables). Including commitments com_r and com_s to exponents r and s, $\pi_{val\text{-}enc}$ demands 21 group elements overall.
6. Generate a NIZK proof $\pi_{\mathcal{R}}$ that the ciphertext ψ_{CS} encrypts a group element $W \in \mathbb{G}$ such that $((X,Y),W) \in \mathcal{R}$. To this end, generate a commitment

$$com_W = (c_{W,1}, c_{W,2}, c_{W,3}) = (g_1^{\theta_1} \cdot g_{3,1}^{\theta_3},\ g_2^{\theta_2} \cdot g_{3,2}^{\theta_3},\ W \cdot g^{\theta_1+\theta_2} \cdot g_{3,3}^{\theta_3})$$

and prove that the underlying W is the same as the one for which $U_4 = W \cdot X_5^r \cdot X_6^s$ in ψ_{CS}. In other words, prove knowledge of exponents $r, s, \theta_1, \theta_2, \theta_3$ such that

$$\Big(\frac{U_1}{c_{W,1}}, \frac{U_2}{c_{W,2}}, \frac{U_4}{c_{W,3}}\Big) = \big(g_1^{r-\theta_1} \cdot g_{3,1}^{-\theta_3},\ g_2^{s-\theta_2} \cdot g_{3,2}^{-\theta_3},\ g^{-\theta_1-\theta_2} \cdot g_{3,3}^{-\theta_3} \cdot X_5^r \cdot X_6^s\big). \tag{8}$$

Commitments to r, s are already part of $\pi_{val\text{-}enc}$. Committing to $\theta_1, \theta_2, \theta_3$ takes 9 elements. Proving the first two relations of (8) requires 4 elements whereas the third one is quadratic and its proof is 9 elements. Proving the linear pairing-product relation $e(g, W) = e(X, Y)$ in NIZK[6] demands 9 elements. Since $\pi_{\mathcal{R}}$ includes com_W, it entails a total of 34 elements.

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

The entire proof $\pi_\psi = com_{\mathsf{cert_{pk}}}||com_{\mathsf{pk}}||\pi_{\mathsf{cert_{pk}}}||\pi_{eq\text{-}key,1}||\cdots||\pi_{eq\text{-}key,6}||\pi_{val\text{-}enc}||\pi_{\mathcal{R}}$ eventually takes 516 elements.

$\mathcal{V}(\mathsf{param}, \psi, L, \pi_\psi, \mathsf{pk_{GM}}, \mathsf{pk_{OA}})$: parse $\mathsf{pk_{GM}}$, $\mathsf{pk_{OA}}$, pk, ψ and π_ψ as above. Return 1 if and only if $\mathcal{V}(\mathsf{VK}, \sigma, (\psi_{\mathsf{CS}}||\psi_{\mathsf{K}_1}||\cdots||\psi_{\mathsf{K}_6}||L)) = 1$, all proofs verify and if $\psi_{\mathsf{K}_1}, \ldots, \psi_{\mathsf{K}_6}$ are all valid tag-based encryptions w.r.t. the tag VK.

$\mathsf{DEC}(\mathsf{sk}, \psi, L)$: parse the ciphertext ψ as $\mathsf{VK}||\psi_{\mathsf{CS}}||\psi_{\mathsf{K}_1}||\cdots||\psi_{\mathsf{K}_6}||\sigma$. Return $\perp$ in the event that $\mathcal{V}(\mathsf{VK}, \sigma, (\psi_{\mathsf{CS}}||\psi_{\mathsf{K}_1}||\cdots||\psi_{\mathsf{K}_6}||L)) = 0$. Otherwise, use sk to decrypt (ψ_{CS}, L).

$\mathsf{OPEN}(\mathsf{sk_{OA}}, \psi, L)$: parse the ciphertext ψ as $\mathsf{VK}||\psi_{\mathsf{CS}}||\psi_{\mathsf{K}_1}||\cdots||\psi_{\mathsf{K}_6}||\sigma$. Return $\perp$ if $\psi_{\mathsf{K}_1}, \ldots, \psi_{\mathsf{K}_6}$ are not all valid TBE ciphertexts w.r.t. the tag VK or if $\mathcal{V}(\mathsf{VK}, \sigma, (\psi_{\mathsf{CS}}||\psi_{\mathsf{K}_1}||\cdots||\psi_{\mathsf{K}_6}||L)) = 0$. Otherwise, decrypt $\psi_{\mathsf{K}_1}, \ldots, \psi_{\mathsf{K}_6}$ using $\mathsf{sk_{OA}}$ and return the resulting $\mathsf{pk} = (X_1, \ldots, X_6)$.

From an efficiency standpoint, the length of ciphertexts is about 1.25 kB in an implementation using symmetric pairings with a 256-bit group order, which is more compact than in the Paillier-based scheme of [29] where ciphertexts take 2.5 kB using 1024-bit moduli. Moreover, our proofs only require 16.125 kB, which is significantly cheaper than in the original GE scheme [29], where interactive proofs reach a communication cost of 70 kB to achieve a 2^{-50} knowledge error.

References

1. 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.
2. M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. P-signatures and noninteractive anonymous credentials. In *TCC'08*, LNCS 4948, pages 356–374, 2008.
3. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Asiacrypt'01*, LNCS 2248, pages 566–582, 2001.
4. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS'93*, pages 62–73, 1993.
5. A. Boldyreva, M. Fischlin, A. Palacio, and B. Warinschi. A closer look at PKI: Security and efficiency. In *PKC'07*, LNCS 4450, pages 458–475, 2007.
6. D. Boneh and X. Boyen. Short signatures without random oracles. In *Eurocrypt'04*, LNCS 3027, pages 56–73, 2004.
7. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Crypto'04*, LNCS 3152, pages 41–55, 2004.
8. 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.
9. X. Boyen and C. Delerablée. Expressive subgroup signatures. In *SCN'08*, LNCS 5229, pages 185–200, 2008.
10. X. Boyen and B. Waters. Full-domain subgroup hiding and constant-size group signatures. In *PKC'07* , LNCS 4450, pages 1–15, 2007.
11. J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In *Eurocrypt 2009*, LNCS 5479, pages 351–368, 2009.
12. J. Camenisch and A. Lysyanskaya. A Signature Scheme with Efficient Protocols. In *SCN'02*, LNCS 2576, pages 268–289, 2003.
13. J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *Crypto'03*, LNCS 2729, pages 126–144, 2003.
14. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *Journal of the ACM*, 51(4):557–594, 2004.
15. R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *Eurocrypt'04*, LNCS 3027, pages 207–222, 2004.
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. R. Cramer and V. Shoup. Signature schemes based on the strong rsa assumption. In *ACM CCS'99*, pages 46–51, 1999.
19. C. Delerablée and D. Pointcheval. Dynamic fully anonymous short group signatures. In *Viecrypt 2006*, LNCS 4341, pages 193–210, 2006.
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. G. Fuchsbauer and D. Pointcheval. Encrypting Proofs on Pairings and Its Application to Anonymity for Signatures. In *Pairing'09*, LNCS series, 2009.
22. S. Goldwasser and Y. Tauman-Kalai. On the (In)security of the Fiat-Shamir Paradigm In *FOCS'03*, pages 102–115, 2003.
23. J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *Asiacrypt'06*, LNCS 4284, pages 444–459, 2006.
24. J. Groth. Fully anonymous group signatures without random oracles. In *Asiacrypt'07*, LNCS 4833, pages 164–180, 2007.
25. J. Groth. Homomorphic trapdoor commitments to group elements. Cryptology ePrint Archive: Report 2009/007, 2009.
26. J. Groth and S. Lu. A non-interactive shuffle with pairing based verifiability. In *Asiacrypt'07*, LNCS 4833, pages 51–67, 2007.
27. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Eurocrypt'08*, LNCS 4965, pages 415–432, 2008.
28. S. Halevi. A Sufficient Condition for Key-Privacy. Cryptology ePrint Archive: Report 2005/005, 2005.
29. A. Kiayias, Y. Tsiounis, and M. Yung. Group encryption. In *Asiacrypt'07*, LNCS 4833, pages 181–199, 2007.
30. A. Kiayias and M. Yung. Group signatures with efficient concurrent join. In *Eurocrypt'05*, LNCS 3494, pages 198–214, 2005.
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. S. Kunz-Jacques and D. Pointcheval. About the security of MTI/C0 and MQV. In *SCN'06*, LNCS 4116, pages 156–172, 2006.
34. M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *STOC'89*, pages 33–43, 1989.
35. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt'99*, *LNCS* 1592, pages 223–238, 1999.
36. 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.
37. H. Shacham. A Cramer-Shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive: Report 2007/074, 2007.
38. V. Shoup. A proposal for the ISO standard for public-key encryption (version 2.1). manuscript, 2001. `http://shoup.net/`.

A Proof of Theorem 1

The security proof of the certification scheme considers three kinds of forgeries in the attack game.

- Type I forgeries: are such that the fake certificate $\mathsf{cert}^{\star}_{\mathsf{pk}^{\star}}$ contains a tuple of elements $(S_1^{\star}, S_2^{\star}, S_3^{\star})$ that never appeared in outputs of certification queries.
- Type II forgeries: are such that $\mathsf{cert}^{\star}_{\mathsf{pk}^{\star}}$ contains a triple $(S_1^{\star}, S_2^{\star}, S_3^{\star})$ that appeared in the output of some query but $\mathsf{cert}^{\star}_{\mathsf{pk}^{\star}}$ also contains commitments $\{(C_{i,1}^{\star}, C_{i,2}^{\star}, C_{i,3}^{\star})\}_{i=1,\ldots,n}$ that do not match those in the output of that query.
- Type III forgeries: are such that $(S_1^{\star}, S_2^{\star}, S_3^{\star})$ and $\{(C_{i,1}^{\star}, C_{i,2}^{\star}, C_{i,3}^{\star})\}_{i=1,\ldots,n}$ are all identical in the fake certificate $\mathsf{cert}^{\star}_{\mathsf{pk}^{\star}}$ and in the output of some certification query. On the other hand, the public key $\mathsf{pk}^{\star} = (X_1^{\star}, \ldots, X_n^{\star})$ is not the one that was certified in that query.

Type I forgeries are easily seen (see lemma 1) to break the HSDH assumption whereas lemma 2 and lemma 3 show that Type II and Type III forgeries give rise to algorithms solving the FlexDH and S2P problems, respectively. $\square$

Lemma 1. *Any Type I forger has advantage at most* $\mathbf{Adv}^{\text{Type-I}}(A) \leq \mathbf{Adv}^{\ell\text{-HSDH}}(\mathcal{B})$, *where* ℓ *is the number of certification queries.*

Proof. The proof is based on ideas from [10]. We outline an algorithm $\mathcal{B}$ that, on input of $\Omega = g^{\omega}$, $u \in \mathbb{G}$ and a set of ℓ triples $(A_i = g^{1/(\omega+c_i)}, B_i = g^{c_i}, C_i = u^{c_i}) \in \mathbb{G}^3$ with $c_1, \ldots, c_\ell \in_R \mathbb{Z}_p^*$, uses a Type I forger to find a triple $(g^{1/(\omega+c)}, g^c, u^c)$ such that $c \neq c_i$ for $i = 1, \ldots, \ell$. To generate the public key PK, $\mathcal{B}$ chooses $\beta_0 \xleftarrow{\$} \mathbb{Z}_p^*$, $\alpha \xleftarrow{\$} \mathbb{Z}_p^*$ and sets $u_0 = g^{\beta_0}$, $A = e(g,g)^{\alpha}$. It also defines $\{(u_{i,1} = g^{\beta_{i,1}}, u_{i,2} = g^{\beta_{i,2}}, u_{i,3} = g^{\beta_{i,2}})\}_{i=1,\ldots,n}$ using random triples $(\beta_{i,1}, \beta_{i,2}, \beta_{i,3}) \xleftarrow{\$} (\mathbb{Z}_p^*)^3$ for $i = 1, \ldots, n$ whereas $\vec{f_1}, \vec{f_2}, \vec{f_3}$ are defined by $f = g^{\theta_0}, f_1 = g^{\theta_1}, f_2 = g^{\theta_2}, f_{3,1} = g^{\theta_1\xi_1}, f_{3,2} = g^{\theta_2\xi_2}$ and $f_{3,3} = g^{\theta_0(\xi_1+\xi_2+\xi_3)}$ for random chosen $\theta_0, \theta_1, \theta_2, \xi_1, \xi_2, \xi_3 \xleftarrow{\$} \mathbb{Z}_p^*$.

To answer certification queries involving public keys $\mathsf{pk} = (X_1, \ldots, X_n)$, $\mathcal{B}$ first computes n commitments to $1_{\mathbb{G}}$. For $i = 1, \ldots, n$, it randomly picks $\phi_{i,1}, \phi_{i,2}, \phi_{i,3} \xleftarrow{\$} \mathbb{Z}_p^*$ and computes a commitment $\vec{C_i} = (f_1^{\phi_{i,1}} \cdot f_{3,1}^{\phi_{i,3}}, f_2^{\phi_{i,2}} \cdot f_{3,2}^{\phi_{i,3}}, f^{\phi_{i,1}+\phi_{i,2}} \cdot f_{3,3}^{\phi_{i,3}})$, which equals $(g^{\eta_{i,1}}, g^{\eta_{i,2}}, g^{\eta_{i,3}})$ where $\eta_{i,1} = \theta_1(\phi_{i,1} + \xi_1\phi_{i,3})$, $\eta_{i,2} = \theta_2(\phi_{i,2} + \xi_2\phi_{i,3})$ and $\eta_{i,3} = \theta_0((\phi_{i,1} + \phi_{i,2}) + (\xi_1 + \xi_2 + \xi_3)\phi_{i,3})$ are all known to $\mathcal{B}$. Certificate parts (S_1, S_2, S_3) are generated as (A_k^{α}, B_k, C_k) using the next available triple (A_k, B_k, C_k) (with $k \in \{1, \ldots, \ell\}$). Finally, as for remaining certificate elements $S_4 = (u_0 \cdot \prod_{i=1}^{n}(C_{i,1}^{\beta_{i,1}} \cdot C_{i,2}^{\beta_{i,2}} \cdot \mathcal{C}_{i,3}^{\beta_{i,3}}))^{c_k}$ and $S_5 = \{(C_{i,1}^{c_k}, C_{i,2}^{c_k}, C_{i,3}^{c_k})\}_{i=1,\ldots,n}$, they are calculated as $S_4 = B_k^{\beta_0 + \sum_{i=1}^{n}\sum_{j=1}^{3}\beta_{i,j}\eta_{i,j}}$ and $S_5 = \{(B_k^{\eta_{i,1}}, B_k^{\eta_{i,2}}, B_k^{\eta_{i,3}}\}_{i=1,\ldots,n}$, respectively. To complete the generation of $\mathsf{cert}_{\mathsf{pk}}$, $\mathcal{B}$ then trapdoor opens $\vec{C}_1, \ldots, \vec{C}_n$ to $(X_1, \ldots, X_n)$ using the trapdoor (ξ_1, ξ_2, ξ_3). More precisely, for $i = 1, \ldots, n$, it computes the de-commitments

$$(D'_{i,1}, D'_{i,2}, D'_{i,3}) = \big(f^{\phi_{i,1}} \cdot (X_i)^{\xi_1/\xi_3}, f^{\phi_{i,2}} \cdot (X_i)^{\xi_2/\xi_3}, f^{\phi_{i,3}} \cdot (1/X_i)^{1/\xi_3}\big).$$

The game ends with $\mathcal{A}$ outputting a pair $(\mathsf{pk}^\star, \mathsf{cert}^\star_{\mathsf{pk}^\star})$ such that $(S_1^\star, S_2^\star, S_3^\star)$ never appeared within outputs of certification queries. Hence, $(S_1^{\star 1/\alpha}, S_2^\star, S_3^\star)$ must solve the HSDH problem. ◻

Lemma 2. *Any Type II forger* $\mathcal{A}$ *making* ℓ *certification queries has no better advantage than* $\mathbf{Adv}^{\text{Type-II}}(A) \leq \ell \cdot (1 - \frac{1}{p}) \cdot \mathbf{Adv}^{\text{FlexDH}}(\mathcal{B})$.

Proof. We show how a Type II forger implies an algorithm $\mathcal{B}$ that finds a non-trivial triple (C, C^a, C^{ab}) on input of $(g, g_a = g^a, g_b = g^b)$. To generate PK, $\mathcal{B}$ chooses $\omega \xleftarrow{\$} \mathbb{Z}_p^*$, $\alpha_a, \alpha_u \xleftarrow{\$} \mathbb{Z}_p^*$ and sets $\Omega = g^{\omega}$, $A = e(g, (g_a \cdot g^{\omega})^{\alpha_a}))$ (so that $\alpha = \log_{e(g,g)}(A)$ is implicitly set as $\alpha = (a+\omega)\alpha_a$) and $u = g^{\alpha_u}$. The commitment key $(\vec{f_1}, \vec{f_2}, \vec{f_3})$ is obtained by choosing $\theta, \theta_1, \theta_2, \xi_1, \xi_2, \xi_3 \xleftarrow{\$} \mathbb{Z}_p^*$ and setting $f = g^{\theta}, f_1 = g^{\theta_1}, f_2 = g^{\theta_2}, f_{3,1} = g^{\theta_1\xi_1}, f_{3,2} = g^{\theta_2\xi_2}$ and $f_{3,3} = g^{\theta(\xi_1+\xi_2+\xi_3)}$.

In the setup phase, $\mathcal{B}$ also computes a set of n commitments to $1_{\mathbb{G}}$, say $\vec{C}_i^\dagger = (g^{\eta^\dagger_{i,1}}, g^{\eta^\dagger_{i,2}}, g^{\eta^\dagger_{i,3}})$ for $i = 1, \ldots, n$, and obtains them by drawing random exponents $\phi^\dagger_{i,1}, \phi^\dagger_{i,2}, \phi^\dagger_{i,3} \xleftarrow{\$} \mathbb{Z}_p^*$ and computing $\eta^\dagger_{i,1} = \theta_1(\phi^\dagger_{i,1} + \xi_1\phi^\dagger_{i,3})$, $\eta^\dagger_{i,2} = \theta_2(\phi^\dagger_{i,2} + \xi_2\phi^\dagger_{i,3})$ as well as $\eta^\dagger_{i,3} = \theta((\phi^\dagger_{i,1} + \phi^\dagger_{i,2}) + (\xi_1+\xi_2+\xi_3)\phi^\dagger_{i,3})$. It also retains $\phi^\dagger_{i,1}, \phi^\dagger_{i,2}, \phi^\dagger_{i,3}$ for later use. Next, $\mathcal{B}$ picks $\rho \xleftarrow{\$} \mathbb{Z}_p^*$ and two sets of n triples $(\rho_{i,1}, \rho_{i,2}, \rho_{i,3}) \xleftarrow{\$} (\mathbb{Z}_p^*)^3$, $(\gamma_{i,1}, \gamma_{i,2}, \gamma_{i,3}) \xleftarrow{\$} (\mathbb{Z}_p^*)^3$ and defines $u_{i,1} = g^{\rho_{i,1}} \cdot g_b^{\gamma_{i,1}}$, $u_{i,2} = g^{\rho_{i,2}} \cdot g_b^{\gamma_{i,2}}$, $u_{i,3} = g^{\rho_{i,3}} \cdot g_b^{\gamma_{i,3}}$, for $i = 1, \ldots, n$, and $u_0 = g^{\rho} \cdot g_b^{\gamma}$ with $\gamma = -\sum_{i=1}^{n}(\gamma_{i,1}\eta^\dagger_{i,1} + \gamma_{i,2}\eta^\dagger_{i,2} + \gamma_{i,3}\eta^\dagger_{i,3})$. This implicitly defines private key elements to be $\beta_{i,1} = \rho_{i,1} + b\gamma_{i,1}$, $\beta_{i,2} = \rho_{i,2} + b\gamma_{i,2}$ and $\beta_{i,3} = \rho_{i,3} + b\gamma_{i,3}$. At the outset of the game, $\mathcal{B}$ also chooses $\ell^\star \xleftarrow{\$} \{1, \ldots, \ell\}$.

When the certification of a public key $\mathsf{pk} = (X_1, \ldots, X_n)$ is queried, the query's treatment depends on the index $k \in \{1, \ldots, \ell\}$ of the query.

- If $k \neq \ell^\star$, $\mathcal{B}$ computes n commitments to $1_{\mathbb{G}}$ (say $\vec{C}_i = (g^{\eta_{i,1}}, g^{\eta_{i,2}}, g^{\eta_{i,3}})$ for $i = 1, \ldots, n$) as in the proof of lemma 1. It generates $\mathsf{cert}_{\mathsf{pk}}$ by setting $S_1 = (g_a \cdot g^{\omega})^{\alpha_a/(\omega + c_k)}$, $S_2 = g^{c_k}$, $S_3 = u^{c_k}$ for a random $c_k \xleftarrow{\$} \mathbb{Z}_p^*$. Since it knows $\prod_{i=1}^n (C_{i,1}^{\beta_{i,1}} \cdot C_{i,2}^{\beta_{i,2}} \cdot C_{i,3}^{\beta_{i,3}}) = \prod_{i=1}^n (u_{i,1}^{\eta_{i,1}} \cdot u_{i,2}^{\eta_{i,2}} \cdot u_{i,3}^{\eta_{i,3}})$ (thanks to $(\eta_{i,1}, \eta_{i,2}, \eta_{i,3})$), it can also compute $S_4 = (u_0 \cdot \prod_{i=1}^n (C_{i,1}^{\beta_{i,1}} \cdot C_{i,2}^{\beta_{i,2}} \cdot C_{i,3}^{\beta_{i,3}}))^{c_k}$. Finally, $S_5 = \{(C_{i,1}^{c_k}, C_{i,2}^{c_k}, C_{i,3}^{c_k})\}_{i=1,\ldots,n}$ is also computable from c_k and $\mathcal{B}$ uses ξ_1, ξ_2, ξ_3 to trapdoor open $\vec{C}_i$ to X_i and obtain de-commitments $(D'_{i,1}, D'_{i,2}, D'_{i,3})$, for $i = 1, \ldots, n$, as in lemma 1.
- If $k = \ell^\star$, $\mathcal{B}$ implicitly defines $c_{\ell^\star} = a$ and sets $S_1 = g^{\alpha_a}$, $S_2 = g_a$, $S_3 = g_a^{\alpha_u}$. Thanks to $\vec{C}_i^\dagger = (C_{i,1}^\dagger, C_{i,2}^\dagger, C_{i,3}^\dagger) = (g^{\eta_{i,1}^\dagger}, g^{\eta_{i,2}^\dagger}, g^{\eta_{i,3}^\dagger})$ that were chosen in the setup phase, $\mathcal{B}$ can compute

$$S_4 = \big(u_0 \cdot \prod_{i=1}^n ({C_{i,1}^\dagger}^{\beta_{i,1}} \cdot {C_{i,2}^\dagger}^{\beta_{i,2}} \cdot {C_{i,3}^\dagger}^{\beta_{i,3}})\big)^a = g_a^{\rho + \sum_{i=1}^n (\rho_{i,1}\eta_{i,1}^\dagger + \rho_{i,2}\eta_{i,2}^\dagger + \rho_{i,3}\eta_{i,3}^\dagger)}.$$

  Finally, $\mathcal{B}$ obtains $S_5 = \{({C_{i,1}^\dagger}^a, {C_{i,2}^\dagger}^a, {C_{i,3}^\dagger}^a) = (g_a^{\eta_{i,1}^\dagger}, g_a^{\eta_{i,2}^\dagger}, g_a^{\eta_{i,3}^\dagger})\}_{i=1,\ldots,n}$. For $i = 1, \ldots, n$, it trapdoor opens $\vec{C}_i^\dagger$ to public key elements X_i using the trapdoor (ξ_1, ξ_2, ξ_3) and the de-commitment $(f^{\phi_{i,1}^\dagger}, f^{\phi_{i,2}^\dagger}, f^{\phi_{i,3}^\dagger})$ that was associated with the commitment to $1_{\mathbb{G}}$. The resulting de-commitment $(D'_{i,1}, D'_{i,2}, D'_{i,3})$ and $\vec{C}_i^\dagger = (C_{i,1}^\dagger, C_{i,2}^\dagger, C_{i,3}^\dagger)$ are included in $\mathsf{cert}_{\mathsf{pk}}$.

Finally, $\mathcal{A}$ outputs a pair $(\mathsf{pk}^\star, \mathsf{cert}^\star_{\mathsf{pk}^\star})$ such that $(S_1^\star, S_2^\star, S_3^\star)$ appeared in the output of some certification query but $\mathsf{cert}^\star_{\mathsf{pk}^\star}$ comprises commitments $\{(C_{i,1}^\star, C_{i,2}^\star, C_{i,3}^\star)\}_{i=1,\ldots,n}$ that do not match those returned in that specific query. With probability $1/\ell$, this query happens to be the $\ell^{\star\text{th}}$ one (and $\mathcal{B}$ fails if this is not the case), so that $(S_1^\star, S_2^\star, S_3^\star) = (g^{\alpha_a}, g_a, g_a^{\alpha_u})$. Then, we must have

$$S_4^\star = \big(u_0 \cdot \prod_{i=1}^n ({C_{i,1}^\star}^{\beta_{i,1}} \cdot {C_{i,2}^\star}^{\beta_{i,1}} \cdot {C_{i,3}^\star}^{\beta_{i,3}})\big)^a, \qquad S_5^\star = \{({C_{i,1}^\star}^a, {C_{i,2}^\star}^a, {C_{i,3}^\star}^a)\}_{i=1,\ldots,n},$$

where $\{(C_{i,1}^\star, C_{i,2}^\star, C_{i,3}^\star)\}_{i=1,\ldots,n} \neq \{(C_{i,1}^\dagger, C_{i,2}^\dagger, C_{i,3}^\dagger)\}_{i=1,\ldots,n}$, in such a way that dividing out the value $S_4 = g_a^{\rho + \sum_{i=1}^n (\rho_{i,1}\eta_{i,1}^\dagger + \rho_{i,2}\eta_{i,2}^\dagger + \rho_{i,3}\eta_{i,3}^\dagger)}$ from $S_4^\star$ yields

$$T = \big(\prod_{i=1}^n (C_{i,1}^\star / C_{i,1}^\dagger)^{\rho_{i,1} + b\gamma_{i,1}} \cdot (C_{i,2}^\star / C_{i,2}^\dagger)^{\rho_{i,2} + b\gamma_{i,2}} \cdot (C_{i,3}^\star / C_{i,3}^\dagger)^{\rho_{i,3} + b\gamma_{i,3}}\big)^a$$

whereas the component-wise quotient of $S_5 = \{(g_a^{\eta_{i,1}^\dagger}, g_a^{\eta_{i,2}^\dagger}, g_a^{\eta_{i,3}^\dagger})\}_{i=1,\ldots,n}$ and S_5^* reveals a triple $\{(Z_{i,1}, Z_{i,2}, Z_{i,3}) = \big((C_{i,1}^\star / C_{i,1}^\dagger)^a, (C_{i,2}^\star / C_{i,2}^\dagger)^a, (C_{i,3}^\star / C_{i,3}^\dagger)^a\big)\}_{i=1,\ldots,n}$. Hence, $\mathcal{B}$ extracts

$$R_3 = \Big(\prod_{i=1}^n (\frac{C_{i,1}^\star}{C_{i,1}^\dagger})^{\gamma_{i,1}} \cdot (\frac{C_{i,2}^\star}{C_{i,2}^\dagger})^{\gamma_{i,2}} \cdot (\frac{C_{i,3}^\star}{C_{i,3}^\dagger})^{\gamma_{i,3}}\Big)^{ab} = T / \prod_{i=1}^n Z_{i,1}^{\rho_{i,1}} \cdot Z_{i,2}^{\rho_{i,2}} \cdot Z_{i,3}^{\rho_{i,3}},$$

$$R_2 = \prod_{i=1}^n Z_{i,1}^{\gamma_{i,1}} \cdot Z_{i,2}^{\gamma_{i,2}} \cdot Z_{i,3}^{\gamma_{i,3}} = \Big(\prod_{i=1}^n (\frac{C_{i,1}^\star}{C_{i,1}^\dagger})^{\gamma_{i,1}} \cdot (\frac{C_{i,2}^\star}{C_{i,2}^\dagger})^{\gamma_{i,2}} \cdot (\frac{C_{i,3}^\star}{C_{i,3}^\dagger})^{\gamma_{i,3}}\Big)^a$$

$$R_1 = \prod_{i=1}^n (\frac{C_{i,1}^\star}{C_{i,1}^\dagger})^{\gamma_{i,1}} \cdot (\frac{C_{i,2}^\star}{C_{i,2}^\dagger})^{\gamma_{i,2}} \cdot (\frac{C_{i,3}^\star}{C_{i,3}^\dagger})^{\gamma_{i,3}}$$

which must form a non-trivial triple (R_1, R_1^a, R_1^{ab}) with overwhelming probability. Indeed, since $\gamma_{i,1}, \gamma_{i,2}, \gamma_{i,3}$ are (information theoretically) independent of $\mathcal{A}$'s view, we can only have $R_1 = 1_{\mathbb{G}}$ by pure chance (with probability $1/p$). $\square$

Lemma 3. *Any Type III forger has advantage at most* $\mathbf{Adv}^{\text{Type-III}}(A) \leq \mathbf{Adv}^{\text{S2P}}(\mathcal{B})$.

Proof. From a Type III adversary $\mathcal{A}$, it is simple to break the binding property of the commitment scheme in section 3.1. Consider an algorithm $\mathcal{B}$ which is given a commitment key $(\vec{f_1}, \vec{f_2}, \vec{f_3})$ and prepares the rest of the public key according to the specification of the scheme in such a way that it can perfectly answer all certification queries.

At the end of the game, $\mathcal{A}$ outputs a pair $\mathsf{pk}^\star$ and $\mathsf{cert}^\star_{\mathsf{pk}^\star}$ such that $\mathsf{cert}_{\mathsf{pk}^\star}$ contains $(S_1^\star, S_2^\star, S_2^\star)$ and commitments $(C^\star_{i,1}, C^\star_{i,2}, C^\star_{i,3})$ that were both contained in the output of some certification query. On the other hand, the public key $\mathsf{pk}^* = (X_1^\star, \ldots, X_n^\star)$ must be different from the one $(X_1, \ldots, X_n)$ that was certified at that query. This necessarily provides $\mathcal{B}$ with two distinct openings $(X_i, (D_{1,i}, D_{2,i}, D_{3,i}))$, $(X_i^\star, (D^\star_{1,i}, D^\star_{2,i}, D^\star_{3,i}))$ (since $X_i \neq X_i^\star$ for at least one index $i \in \{1, \ldots, n\}$) of some commitment $(C^\star_{i,1}, C^\star_{i,2}, C^\star_{i,3})$, which violates the S2P assumption. $\square$

B Security Proofs for the Group Encryption Scheme

Correctness is straightforward and we focus on anonymity, message security and soundness.

Theorem 2. *The* GE *scheme satisfies anonymity assuming that* Σ *is strongly unforgeable, that* H *is collision-resistant and that the DLIN assumption holds in* $\mathbb{G}$.

Proof. We consider a sequence of games where the first game is the real experiment of definition 6 while the adversary $\mathcal{A}$ is essentially a key privacy attacker against the linear Cramer-Shoup system in the last game. In Game i, we call W_i the event that $\mathcal{A}$ wins.

Game 1: the challenger $\mathcal{B}$ generates param that includes a reference string $\mathbf{g}$ containing $\vec{g_1}$, $\vec{g_2}$ and $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$, with $\xi_1, \xi_2 \xleftarrow{\$} \mathbb{Z}_p^*$. The public key $\mathsf{pk}_{\mathsf{OA}} = (Y_1, Y_2, Y_3, Y_4)$ is given to $\mathcal{A}$ who generates $\mathsf{pk}_{\mathsf{GM}}$ on her own. By invoking the USER oracle, she certifies two distinct receivers' public keys $\mathsf{pk}_0 = (X_1^0, \ldots, X_6^0)$, $\mathsf{pk}_1 = (X_1^1, \ldots, X_6^1)$ chosen by $\mathcal{B}$ and makes a number of opening queries and decryption queries, which $\mathcal{B}$ handles using $\mathsf{sk}_{\mathsf{OA}}$ and sk_0, sk_1, respectively. At some point, she outputs $((X,Y), W, L, \mathsf{pk}_{\mathcal{R}})$ such that $((X,Y), W) \in \mathcal{R}$ and obtains, as a challenge, a group encryption $\psi^\star = \mathsf{VK}^\star || \psi^\star_{\mathsf{CS}} || \psi^\star_{\mathsf{K}_1} || \cdots || \psi^\star_{\mathsf{K}_6} || \sigma^\star$ of W under pk_b, for some bit $b \in \{0,1\}$ of $\mathcal{B}$'s choice. Then, she obtains proofs $\pi^\star_{\psi^\star}$ for $\psi^\star$ and makes new opening and decryption queries under the obvious restrictions. She finally outputs b' and we call W_1 the event that $b' = b$.

Game 2: is as game 1 but $\mathcal{B}$ aborts in the event F_2 that $\mathcal{A}$ queries the opening of a ciphertext $\psi = \mathsf{VK} || \psi_{\mathsf{CS}} || \psi_{\mathsf{K}_1} || \cdots || \psi_{\mathsf{K}_6} || \sigma$ 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 $|\Pr[W_2] - \Pr[W_1]| \leq \Pr[F_2] \in \mathsf{negl}(\lambda)$ if Σ is strongly unforgeable.

Game 3: we modify the generation of the common reference string $\mathbf{g} = (\vec{g_1}, \vec{g_2}, \vec{g_3})$ in param and choose the vector $\vec{g_3}$ as $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2} \odot (1,1,g)^{-1}$ (instead of $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$). Under the DLIN assumption, this change is not noticeable to $\mathcal{A}$ and $|\Pr[W_3] - \Pr[W_2]| \in \mathsf{negl}(\lambda)$.

Game 4: we change the generation of proofs $\pi^\star_{\psi^\star}$ and use the trapdoor of the CRS (*i.e.*, values ξ_1 and ξ_2 such that $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2} \odot (1,1,g)^{-1}$ and commitments to exponents are thus generated using $\vec{\varphi} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$) instead of some of the actual witnesses. Namely, $\pi^\star_{\mathit{eq\text{-}key},1}, \ldots, \pi^\star_{\mathit{eq\text{-}key},6}$ as well as $\pi^\star_{\mathit{val\text{-}enc}}$ and $\pi^\star_{\mathcal{R}}$ are simulated without using encryption exponents $\{w^\star_{i,1}, w^\star_{i,2}\}_{i=1,\ldots,6}$ (that are used to encrypt $\psi^\star_{\mathsf{K}_i}$) and $r^\star, s^\star$ (that are used to compute $\psi^\star_{\mathsf{CS}}$). Commitments $\mathit{com}^\star_{\mathit{pk}_b}$ and $\mathit{com}^\star_W$ are still generated using pk_b and W but commitments to $w^\star_{i,1}, w^\star_{i,2}$ and $r^\star, s^\star$ are replaced by commitments to 0. Yet, the trapdoor ξ_1, ξ_2 allows generating proofs that have the same distribution as real proofs (e.g., [11, Section 4.4] shows how to simulate linear multi-exponentiation equations whereas quadratic ones, such as the third relation of (8), are also simulatable without $r^\star, s^\star$) and it comes that $\Pr[W_4] = \Pr[W_3]$.

Game 5: we modify the generation of the challenge ciphertext $\psi^\star$ and let $\psi^\star_{\mathsf{K}_1}, \ldots, \psi^\star_{\mathsf{K}_6}$ be encryptions of random group elements instead of $X_1^b, \ldots, X_6^b$. Since exponents $\{w^\star_{i,1}, w^\star_{i,2}\}_{i=1,\ldots,6}$ are no longer used in Game 4, any significant change in the distribution of $\mathcal{A}$'s output would give rise to a selective-tag weak CCA2 attacker[7] against the tag-based encryption (recall that opening queries do not involve $\mathsf{VK}^\star$ unless the rejection rule of Game 2 applies). According to theorem 5.1 in [31], we have $|\Pr[W_5] - \Pr[W_4]| \in \mathsf{negl}(\lambda)$ if DLIN holds.

In Game 5, $\mathcal{A}$ is essentially playing a CCA2 anonymity attack against the linear Cramer-Shoup encryption scheme. Indeed, elements $\psi^\star_{\mathsf{K}_i}$ do not depend on b and, since proofs are given in the WI setting, they reveal no information on underlying witnesses (in particular, $com^\star_{\mathsf{pk}_b}$ and $com^\star_{\mathsf{cert}_{\mathsf{pk}_b}}$ are perfectly hiding commitments). As for the key privacy of the linear Cramer-Shoup encryption scheme, re-proving theorem 6 in [3] using DLIN in place of DDH is just an exercise and we eventually obtain $|\Pr[W_5] - 1/2| \in \mathsf{negl}(\lambda)$ if DLIN holds and if H is collision-resistant. □

Theorem 3. *The GE scheme satisfies message security assuming that Σ is strongly unforgeable, that H is a UOWHF and that the DLIN assumption holds in $\mathbb{G}$.*

Proof. We use a sequence of games. The first one mirrors the experiment of definition 5 where the challenger's bit b is 1 and the adversary obtains a encryption of the witness W 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 a fake CRS (constructing a simulator for $\mathsf{PROVE}(.)$ using a simulated CRS is part of the security analysis as stressed in [29]). 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 $(\vec{g_1}, \vec{g_2}, \vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2})$, with $\xi_1, \xi_2 \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$. The adversary generates public keys $\mathsf{pk}_{\mathsf{OA}}$ and $\mathsf{pk}_{\mathsf{GM}}$ on her own. The challenger and $\mathcal{A}$ run an execution of JOIN where $\mathcal{A}$ certifies the public key $\mathsf{pk} = (X_1, \ldots, X_6)$ 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 sk that matches pk. At some point, $\mathcal{A}$ outputs $((X,Y), W, L, \mathsf{pk}_{\mathcal{R}})$ such that $((X,Y),W) \in \mathcal{R}$ and obtains in return a group encryption $\psi^\star = \mathsf{VK}^\star||\psi^\star_{\mathsf{CS}}||\psi^\star_{\mathsf{K}_1}||\cdots||\psi^\star_{\mathsf{K}_6}||\sigma^\star$ of W under pk and L. Then, she obtains polynomially many proofs $\pi^\star_{\psi^\star}$ for $\psi^\star$ and makes new decryption queries under the obvious restrictions. She finally outputs b' and we call W_1 the event that $b' = 1$.

Game 2: we modify the generation of the common reference string $\mathbf{g} = (\vec{g_1}, \vec{g_2}, \vec{g_3})$ in param and choose $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2} \odot (1,1,g)^{-1}$ (instead of $\vec{g_3} = \vec{g_1}^{\xi_1} \odot \vec{g_2}^{\xi_2}$). Under the DLIN assumption, this change is not noticeable to $\mathcal{A}$ and $|\Pr[W_2] - \Pr[W_1]| \in \mathsf{negl}(\lambda)$.

Game 3: we modify the $\mathsf{DEC}(.)$ oracle and let $\mathcal{B}$ reject any ciphertext of the form $\psi = \mathsf{VK}||\cdots||\sigma$ such that $\mathsf{VK} = \mathsf{VK}^\star$ ($\mathsf{VK}^\star$ can be generated at the outset of the game). Let F_3 be the event that this rule causes $\mathcal{B}$ to reject a ciphertext that would not have been rejected in Game 2. As in the

[7] 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 she can query for any ciphertext-tag pair (C,t) such that $t \neq t^\star$. At the challenge phase, she chooses plaintexts m_0, m_1 and receives a ciphertext $C^\star$ encrypting m_b (under the tag $t^\star$) for some bit $b \stackrel{\$}{\leftarrow} \{0,1\}$ that $\mathcal{A}$ eventually aims to guess after further decryption queries.

proof of theorem 2, we have $|\Pr[W_3] - \Pr[W_2]| \leq \Pr[F_3] \in \mathsf{negl}(\lambda)$ if Σ is strongly unforgeable.

Game 4: we change the generation of proofs $\pi^\star_{\psi^\star}$ and use the trapdoor of the CRS instead of witnesses W and $coins_{\psi^\star} = \{(r^\star, s^\star), \{(w^\star_{i,1}, w^\star_{i,2})\}_{i=1,\ldots,6}\}$. More precisely, $\{\pi^\star_{eq\text{-}key,i}\}_{i=1,\ldots,6}$ (which prove that com_{X_i} and ψ_{K_i} hide the same X_i), as well as $\pi^\star_{val\text{-}enc}$ and $\pi^\star_{\mathcal{R}}$ (*i.e.*, the proofs that $\psi^\star_{\mathsf{CS}}$ is a valid ciphertext and that $\psi^\star_{\mathsf{CS}}$ and com_W contain the same W) are simulated without using encryption exponents $\{w^\star_{i,1}, w^\star_{i,2}\}_{i=1,\ldots,6}$ and $r^\star, s^\star$ and commitments to the latter values are replaced by commitments to 0. Also, the part of $\pi^\star_{\mathcal{R}}$ that proves relation $e(g, W) = e(X, Y)$ (and thus $((X, Y), W) \in \mathcal{R}$) is simulated in NIZK[8] by setting com_W as a commitment to $1_{\mathbb{G}}$. As in the proof of theorem 2, the trapdoor ξ_1, ξ_2 allows generating simulated proofs that are perfectly indistinguishable from real proofs, so that $\Pr[W_4] = \Pr[W_3]$.

Game 5: in the calculation of $\psi^\star$, we set $\psi^\star_{\mathsf{CS}}$ as an encryption of a random group element. Since $r^\star, s^\star$ are not used in Game 4, any significant change in $\mathcal{A}$'s behavior would imply a CCA2 attacker (in the modeling of CCA2-security for labeled cryptosystems [38]) against the linear Cramer-Shoup scheme (recall that decryption queries do not involve $\mathsf{VK}^\star$ unless the rejection rule of Game 3 applies, which prevents $\mathcal{A}$ from mauling $\psi^\star_{\mathsf{K}_i}$ while keeping the same $(\psi^\star_{\mathsf{CS}}, \mathsf{VK}^\star, L)$). The result of [37] implies that $|\Pr[W_5] - \Pr[W_4]| \in \mathsf{negl}(\lambda)$ if DLIN holds and H is a UOWHF.

Game 6: we change again the $\mathsf{DEC}(.)$ oracle and do not apply the rejection rule of Game 3 anymore. If Σ is strongly unforgeable, we must have $|\Pr[W_6] - \Pr[W_5]| \in \mathsf{negl}(\lambda)$.

We see that, from Game 4 onwards, the oracle $\mathsf{PROVE}(.)$ does not use witnesses $W, coins_{\psi^\star}$ any longer. Game 6 is thus the experiment of definition 5 where the challenger's bit b is 0. Putting the above altogether, we find $|\Pr[W_6] - \Pr[W_1]| \in \mathsf{negl}(\lambda)$, which establishes the result. $\square$

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.

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