An extended abstract of this work appeared at INDOCRYPT 2020. This is the full version.

Skipping the q in Group Signatures

Olivier Blazy[1]* Saqib A. Kakvi[2]

October 26, 2020

[1]XLim, Université de Limoges
olivier.blazy@unilim.fr
[2]Bergische Universität Wuppertal
kakvi@uni-wuppertal.de

Abstract

The notion of group signatures was introduced to allow group members to sign anonymously on behalf of a group. A group manager allows a user to join a group, and another will be able to open a signature to revoke its anonymity. Several schemes have already been proposed to fulfil these properties, however very few of them are proven in the standard model. Of those proven in the standard model, most schemes rely on a so-called q-assumptions. The underlying idea of a q-assumptions is that to prove the security of the scheme, we are given a challenge long enough to allow the simulator to answer queries. Another common solution is to rely on an interactive hypothesis. We provide one of the first schemes proven in the standard model, requiring a constant-size non-interactive hypothesis. We then compare its efficiency to existing schemes, and show that this trade-off is acceptable as most schemes with better efficiency rely on either an interactive or a q-hypothesis. The exception to this is the recent independent work Libert, Peters and Yung (CRYPTO 2015), who presented an efficient group signature scheme in the standard model relying on standard assumptions.

1 Introduction

A group signature scheme [Cv91] is a protocol which lets a member of a group individually issue signatures on behalf of the group, in an anonymous but traceable way. To this end, we have a trusted manager, called the Opener, who can revoke anonymity of the actual signer in case of abuse. Group signatures primarily guarantee *anonymity*, which means that nobody (except the Opener) can link the signature to the signer, but also *unlinkability*, which means that one cannot two signatures were produced by the same user or not. One application Group Signature use case is anonymous auction, where users are interested in knowing what the best offer is. Additionally, no bid (except the winning one) should be linkable to its owner.

Several steps have been made in the study of those protocols: Bellare, Micciancio and Warinschi [BMW03] gave formal definitions of the security properties of group signatures (the BMW model) and proposed a (impractical proof of concept) scheme under general assumptions. However, this model required that the size of the group be fixed a priori and may not change and are known as static groups. Later, Bellare, Shi and Zhang [BSZ05], and independently Kiyias and Yung [KY06], extended this model to dynamic groups (the BSZ model), which allows the group to grow arbitrarily large, emphasizing the importance of unforgeability and anonymity. To allow for this another group manager, called the Issuer, is required.

1.1 Previous Work

The first efficient proposed group signature schemes were proven in the Random Oracle Model [ACJT00, AST02, BBS04, BS04]. Camenisch and Lysyanskaya [CL04] proposed one of the first standard model schemes.

*The 1st author was supported by the French ANR project ID-FIX (ANR-16-CE39-0004).

Despite being fairly efficient, these schemes suffered from the drawback that the signatures were non-constant and would grow related to either the size of the group or the number of revoked users.

The first group signature with constant size was due to Groth [Gro06], but the constant exceptionally large and impractical. Soon after, another scheme was proposed by Boyen and Waters [BW06], with more plausible sizes. Groth then improved on the scheme of [Gro06] in [Gro07] and provided not only an efficient group signature scheme, but also presented a generic approach consisting in using a re-randomizable certificate to produce a certified signature. Kakvi then proposed in [Kak10] some improvements that lead to a more efficient SXDH instantiation. In another line of work, Delerablée and Pointcheval [DP06] proposed another short scheme based, which was improved and extended by Blazy and Pointcheval in [BP12]. However, one critical flaw shared by all these schemes was that they relied on q-type assumptions. We attempt to overcome this, by presenting a scheme without relying on q-type assumptions.

q-type assumptions are a common type of assumption in modern cryptographic schemes, due to their powerful nature. A large number of these are used due to the fact that we are unable to compute responses to oracle queries by adversaries in our reductions. In these, one must rely upon parametrized or so-called q-type assumption families. In these types of assumptions, we are given q intermediate values to aid in answering oracle queries. These assumptions tend to be quite bespoke assumptions for the scheme in question and may not be useful for anything else, which in turn means they will probably not be well studied. On occasion, authors will provide a proof in the generic group model of the hardness of an assumption; this is neither standard practice nor guarantee of the actual security of the assumption. This has also lead to "a dizzying array of tailor-made complexity assumptions." [Boy08].

As a result of this, several researchers have asked if it is at all possible to do away with q-type assumptions, or if there is some property in them that makes them indispensable. A study of sub-class of these assumptions, known as the "one-more" assumptions was undertaken by Bellare et al. [BNPS02] and later by Bresson et al. [BMV08]. The later framework of Boneh, Boyen and Goh [BBG05] which was later dubbed the "Uber-Assumption" [Boy08], the coining of which is attributed to Dan Boneh [Boy08, Footnote 1]. This framework for a unified description of all q-type assumptions was the first step in a full classification of q-type assumption.

There have been other works looking at removing q-type assumptions from cryptographic primitives. The question of generically removing q-assumptions was studied by Chase and Meiklejohn [CM14]. The approach of Chase and Meiklejohn [CM14] transforms schemes in a prime order pairing groups to schemes in composite order groups. This work was further extended by Chase, Maller and Meiklejohn [CMM16], covering more assumptions and having a better security loss, however, they still require composite order groups. In the other direction, Bresson, Monnerat and Vergnaud [BMV08] showed separation between q-type assumptions and their non-q or simple variants, using algebraic reductions.

In a recent independent work, Libert, Peters and Yung [LPY15] have proposed a group signatures scheme, which is secure without any q-type assumptions. They use the recent advances in structure preserving signatures [AFG$^+$10] as a principal building block. However, the drawback of this construction, as also noted by Libert, Peters and Yung, is that their scheme is not modular and cannot be easily adapted with new building blocks. In contrast, our construction is highly modular and achieves similar efficiency to the of Libert, Peters and Yung [LPY15].

Apart from the group signature schemes we discuss, there have been some advances in other directions regarding group signatures. There have been efficient schemes suggested in the Generic Group Model [CS18], the Random Oracle Model [OEH$^+$19, GL19, DS18] or based on other assumptions, such as lattices [GKV10, dLS18, LNWX18, KY19, LNWX17, LLM$^+$17] and code-based assumptions [ABCG16]. Additionally, Bootle et al. [BCC$^+$16] presented a model for fully dynamic group signatures, i.e. signatures that can not only grow but also shrink in size. We do not compare our scheme to any of these, as they are in a different setting to the one we consider.

1.2 Our Contribution.

In this paper, we present a simple and efficient construction of group signatures. In independent work, Libert, Peters and Yung [LPY15] recently presented a compact groups signature scheme based on standard, or “simple”, assumptions[1], parallel to our work [BK15]. We note that our signatures are of comparable size to that of Libert, Peters and Yung, but we have a stronger security notion. Where as the underlying signature scheme in [LPY15] is proven to be F-CMA secure, we prove full UF-CMA security. Note that both schemes have a loss linear in the number of signing queries.

We present simple and efficient constructions of group signatures. They can be proven under reasonable assumptions (variations of the SDH) and prove the security of both schemes in the standard model. In this paper we combine the use of a Delerablée-Pointcheval [DP06] certificate for Waters' signature [Wat05], and the Groth-Sahai [GS08] methodology. We describe our instantiation through the framework of Groth [Gro06, Gro07] for generic group signatures.

1.3 Organization

In the next section, we present the primitive of group signature and the security model, due to Bellare, Shi and Zhang [BSZ05]. Then, we present the basic tools upon which our instantiations rely. Eventually, we describe our schemes, in the SXDH setting, with the corresponding assumptions for the security analysis that is provided. For the sake of consistency, we then explain the results with the (intuitive) DLin instantiations of this scheme in Section 4. This scheme requires roughly the same number of group elements and, based on the chosen elliptic curve and the way one wants to verify the signatures, one may prefer one instantiation to the other. It also allows us to compare our signature with the previous one, and show that we are roughly as efficient as most of the modern schemes, even though we require neither a q-assumption nor an interactive one.

2 Preliminaries

2.1 Computational Assumptions.

Our protocols will work with a pairing-friendly elliptic curve, of prime order:

- $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are multiplicative cyclic groups of finite prime order p, and g_1, g_2 are generators of $\mathbb{G}_1, \mathbb{G}_2$;
- e is a map from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$, that is bilinear and non-degenerated, such that $e(g_1, g_2)$ is a generator of $\mathbb{G}_T$.

In particular, we consider Type 3 group, as per the definitions of Galbraith, Paterson and Smart [GPS08]. For our purposes, we will need the following assumptions.

Definition 1 (Symmetric eXternal Diffie-Hellman [BBS04])**.** Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic groups of prime order, $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map. The SXDH assumption states that the DDH assumption holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

Definition 2 (q-Strong Diffie-Hellman Assumption in $\mathbb{G}$ [BB04])**.** Let $\mathbb{G}$ be a cyclic group of order p generated by g. The q-SDH problem consists, given $(g, g^{\gamma}, g^{\gamma^2}, \dots, g^{\gamma^q})$, in computing a pair $(x, g^{1/\gamma+x})$.

Definition 3 (Decision Linear Assumption in $\mathbb{G}$ [BBS04])**.** Let $\mathbb{G}$ be a cyclic group of prime order, with generator g. The DLin assumption states that given $(g, g^x, g^y, g^{ax}, g^{by}, g^c)$, it is hard to decide if $c = a + b$ or not, for random $a, b, x, y \in \mathbb{Z}_p$.

Definition 4 (eXternal Decision Linear 2 Assumption [AFG$^+$10])**.** Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic groups of prime order, with generators (g_1, g_2), and $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map.The XDLin_2 assumption states that given a tuple of the form $(g_1, g_1^x, g_1^y, g_1^{ax}, g_1^{by}, g_2, g_2^x, g_2^y, g_2^{ax}, g_2^{by}, g_2^c)$, it is hard to decide if $c = a + b$ or not, for random $a, b, x, y \in \mathbb{Z}_p$.

[1] While q-assumptions are more and more common, they require a polynomial number of inputs and thus should be avoided to provide a drastic improvement in security.

Definition 5 (Advanced Computational Diffie-Hellman [BFPV11])**.**
Let us be given two cyclic groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order p with (g_1, g_2) as respective generators and e an admissible bilinear map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. The CDH^+ assumption states that given $(g_1, g_2, g_1^a, g_2^a, g_1^b)$, for random $a, b \in \mathbb{Z}_p$, it is hard to compute g_1^{ab}.

Definition 6 (q-Double Hidden Strong Diffie-Hellman [FPV09])**.** Let $\mathbb{G}_1, \mathbb{G}_2$ be multiplicative cyclic groups of order p generated by g_1, g_2 respectively. The q-DHSDH problem consists given $(g_1, k_1, g_2, g_2^{\gamma})$ and several tuples of the form $(g_1^{x_i}, g_2^{x_i}, g_1^{y_i}, g_2^{y_i}, (k_1 g_1^{y_i})^{1/(\gamma+x_i)})_{i\in[1,q]}$ in computing a tuple
$(g_1^{x}, g_2^{x}, g_1^{y}, g_2^{y}, (k_1 g_1^{y})^{1/\gamma+x})$ for a new pair (x, y).

For completeness, we state the non-q ($q = 0$) version of this assumption, which we will need for our security proof.

Definition 7 (Double Hidden Strong Diffie-Hellman in $\mathbb{G}_1, \mathbb{G}_2$)**.** Let $\mathbb{G}_1, \mathbb{G}_2$ be multiplicative cyclic groups of order p generated by g_1, g_2 respectively. The DHSDH problem consists of, given $(g_1, k_1, g_2, g_2^{\gamma})$ in computing a tuple of the form $(g_1^{x}, g_2^{x}, g_1^{y}, g_2^{y}, (k_1 g_1^{y})^{1/\gamma+x})$ for any pair (x, y).

It is critical to note that the DHSDH is *not* a q-type assumption. The q-DHSDH is an extension of the DHSDH assumption to a q-type assumption. Our goal is to remove any dependencies of number of queries from our assumptions, which is exactly was using the DHSDH instead of the q-DHSDH achieves. Additionally, the DHSDH is a strictly stronger problem that the q-DHSDH, as any algorithm that can solve the DHSDH problem can also solve the q-DHSDH problem.

2.2 Dynamic Group Signatures

We prove our scheme secure in the growing group security model of Bellare, Shi and Zhang [BSZ05], here on in referred to as the BSZ model. The model implicitly requires that all users have their own personal signing/verification key pairs, which are all registered in a Public Key Infrastructure(PKI). We thus assume that any user $\mathcal{U}_i$ wishing to join the group owns a public-secret key pair $(\mathsf{usk}[i], \mathsf{upk}[i])$, certified by the PKI. Within our group signature setting, we have two distinct[2] authorities or managers, namely:

- The *Issuer* who adds new uses to the group and issues them with a group signing key and the corresponding certificate,
- The *Opener*, it is able to "open" any signature and extract the identity of the signer.

A group signature scheme is defined by a sequence of (interactive) protocols, $\mathsf{GS} = (\mathsf{Setup}, \mathsf{Join}, \mathsf{Sig}, \mathsf{Verif}, \mathsf{Open}, \mathsf{Judge})$, which are defined as follows:

- $\mathsf{Setup}(1^{\lambda})$: Generates the group public key gpk, the issuer key ik for the Issuer, and the opening key ok for the Opener.
- $\mathsf{Join}(\mathcal{U}_i)$: This is an interactive protocol between a user $\mathcal{U}_i$ (who has their secret key $\mathsf{usk}[i]$) and the Issuer (using his private key ik). At the end of the protocol, the user obtains their group signing key $\mathsf{sk}[i]$, and the group manager adds the user to the registration list, Reg. We denote the set of registered users by I.
- $\mathsf{Sig}(\mathsf{gpk}, , m, \mathsf{sk}[i])$: Produces a group signature σ on the message m, under user U_i's group signing key $\mathsf{sk}[i]$.
- $\mathsf{Verif}(\mathsf{gpk}, m, \sigma)$: Verifies the validity of the group signature σ, with respect to the public key gpk. This algorithm thus outputs 1 if and only if the signature is valid.

[2] The BSZ model requires that both authorities must be distinct for certain notions of security. However, one could have them as the same entity in a relaxed version of the BSZ security model.

	Issuer	Opener
Traceability	Uncorrupted	Partially Corrupted
Anonymity	Fully Corrupted	Uncorrupted
Non-Frameabiity	Fully Corrupted	Fully Corrupted

Table 1: Maximum corruption level of managers allowed.

- $\mathsf{Open}(\mathsf{gpk}, m, \sigma, \mathsf{ok})$: If σ is valid, the Opener, using ok, outputs a user identity i assumed to be the signer of the signature with a proof τ of this accusation.
- $\mathsf{Judge}(g\mathsf{pk}, m, \sigma, i, \tau)$: Verifies that the opening of σ to the identity i was indeed correctly done.

We now recall the BSZ security model for group signatures. As stated before, group signatures guarantee *anonymity, unlinkability* and *non-frameability*, which we will explain below. We require that we have two group authorities, namely the *Issuer*, who will issue certificates to grant access to the group, and an *Opener* who will be able to revoke anonymity, and thus trace back the actual signers. For different security notions, we allow each authority to be independently corrupted to some extent. We say an authority is fully corrupted if it reveals its key and potentially deviates from its algorithm. We say an authority is partially corrupted if it reveals its key but does not deviate from its algorithm. An uncorrupted authority neither reveals its key nor deviates from its algorithm. We give the maximum corruption levels allowed for each security property in the table below

Additionally, we assume that each user $\mathcal{U}_i$ owns a pair $(\mathsf{usk}[i], \mathsf{upk}[i])$ certified by a Public Key Infrastructure (PKI). We now recall the security notions.

2.3 Correctness

The *correctness* notion guarantees that honest users should be able to generate valid signatures, and the Opener should then be able to get the identity of the signers, and provide a convincing proof for the Judge. In the following experiments that formalize the security notions, the adversary can run the Join protocol:

- Either through the joinP-oracle (passive join), which means that it creates an honest user for whom it does not know the secret keys: the index i is added to the HU (Honest Users) list. The adversary gets back the public part of the certificate $\mathsf{pk}[i]$;
- or through the joinA-oracle (active join), which means that it interacts with the group manager to create a user it will control: the index i is added to the CU (Corrupted Users) list. The adversary gets back the whole certificate $\mathsf{pk}[i]$, and $\mathsf{sk}[i]$.

For users whose secret keys are known to the adversary, we let the adversary play on their behalf. For honest users, the adversary can interact with them, granted some oracles:

- $\mathsf{corrupt}(i)$, if $i \in \mathsf{HU}$, provides the secret key $\mathsf{sk}[i]$ of this user. The adversary can now control it. The index i is then moved from HU to CU;
- $\mathsf{Sig}(i, m)$, if $i \in \mathsf{HU}$, plays as the honest user i would do in the signature process. Then i is appended to the list $\S[m]$.

2.4 Traceability

Traceability asserts that nobody should be able to produce a valid signature that cannot be opened in a valid and convincing way. We detail the traceability experiment in Figure 1 below.

We define the advantage of an adversary against traceability as:

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

and we say that a group signature scheme is *traceable* if, for any polynomial adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{tr}}_{\mathsf{GS},\mathcal{A}}(\lambda)$ is negligible.

```
Experiment $\mathsf{Exp}^{\mathsf{tr}}_{\mathsf{GS},\mathcal{A}}(\lambda)$
1. $(\mathsf{pk}, \mathsf{msk}, \mathsf{skO}) \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(m, \sigma) \leftarrow \mathcal{A}(\mathsf{pk} : \mathsf{joinA}, \mathsf{joinP}, \mathsf{corrupt}, \mathsf{Sig}, \mathsf{open})$
3. IF $\mathsf{Verif}(\mathsf{pk}, m, \sigma) = 0$, RETURN 0
4. IF $\exists j \notin \mathsf{CU} \cup \S[m]$,
         $\mathsf{Open}(\mathsf{pk}, m, \sigma, \mathsf{skO}) = (j, \Pi)$
     RETURN 1
5. ELSE RETURN 0
```

Figure 1: Traceability Experiment

2.5 Non-Frameability

Non-frameability guarantees that no dishonest player (even the authorities, *i.e.* the Issuer and the Opener, hence the keys msk and skO provided to the adversary) will be able to frame an honest user. That is to say an honest user that does not sign a message M should not be convincingly declared as a possible signer. We detail the non-frameability experiment in Figure 2 below.

```
Experiment $\mathsf{Exp}^{\mathsf{nf}}_{\mathsf{GS},\mathcal{A}}(\lambda)$
1. $(\mathsf{pk}, \mathsf{msk}, \mathsf{skO}) \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(m, \sigma) \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{msk}, \mathsf{skO} : \mathsf{joinP}, \mathsf{corrupt}, \mathsf{Sig})$
3. IF $\mathsf{Verif}(\mathsf{pk}, m, \sigma) = 0$ RETURN 0
4. IF $\exists i \in \mathsf{HU} \setminus \S[m]$,
         $\mathsf{Open}(\mathsf{pk}, m, \sigma, \mathsf{skO}) = (i, \Pi)$
     RETURN 1
5. ELSE RETURN 0
```

Figure 2: Non-Frameability Experiment

We define the advantage of an adversary against non-frameability as

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

and we say a group signature scheme is *non-frameable* if, for any polynomial adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{nf}}_{\mathsf{GS},\mathcal{A}}(\lambda)$ is negligible.

2.6 Anonymity

Anonymity states that the signer of a message remains anonymous. In particular, given two of honest users i_0 and i_1, the adversary should not have any significant advantage in guessing which one of them have issued a valid signature. The adversary can interact with honest users as before (with Sig and $\mathsf{corrupt}$), but the challenge signature is generated using the interactive signature protocol Sign, where the adversary plays the role of the corrupted users, but honest users are activated to play their roles.

We define the advantage of an adversary against anonymity as:

$$\mathsf{Adv}^{\mathsf{An}}_{\mathsf{GS},\mathcal{A}}(\lambda) = \Pr[\mathsf{Exp}^{\mathsf{An}-1}_{\mathsf{GS},\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{An}-0}_{\mathsf{GS},\mathcal{A}}(\lambda) = 1]$$

and we say that a group signature scheme is *anonymous* for any polynomial adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{An}}_{\mathsf{GS},\mathcal{A}}(\lambda)$ is negligible.

```
Experiment $\mathsf{Exp}^{\mathsf{An}-b}_{\mathsf{GS},\mathcal{A}}(\lambda)$
1. $(\mathsf{pk}, \mathsf{msk}, \mathsf{skO}) \leftarrow \mathsf{Setup}(1^\lambda)$
2. $(m, i_0, i_1) \leftarrow \mathcal{A}(\mathtt{FIND}, \mathsf{pk}, \mathsf{msk} : \mathsf{joinP}, \mathsf{corrupt}, \mathsf{Sig})$
3. $\sigma \leftarrow \mathsf{Sign}(\mathsf{pk}, i_b, m, \mathsf{sk}[i])$
4. $b' \leftarrow \mathcal{A}(\mathtt{GUESS}, \sigma : \mathsf{joinP}, \mathsf{corrupt}, \mathsf{Sig})$
5. IF $i_0 \notin \mathsf{HU}$ OR $i_1 \notin \mathsf{HU}$ RETURN $0$
6. RETURN $b'$
```

Figure 3: Anonymity Experiment

2.7 Certified Signatures

We use a primitive known as a certified signature scheme which was introduced by Boldyreva et al. [BFPW07]. A certified signature scheme is a signature scheme where the well-formedness of the public key is verifiable due to an additional certificate.

We use the BBS-like certification [BBS04] proposed by Delerablée and Pointcheval [DP06] to certify a Waters public key [Wat05]. When a receiver wishes to verify a certified signature, he will not only verify the signature, as per usual, but also verify the certificate of the well-formedness of the public key.

The security requirements for certified signatures is that we should neither be able to create a signature using a faked certificate key nor forge a signature for an already issued certificate. Although Boldyreva et al. provide more general security requirements, we use slightly simpler definitions, as in previous works. For a certified signature scheme to be secure, we require it to satisfy the following conditions:

- **Unfakeability:** No adversary should be able to produce a valid certificate for a key pair generated of his choice, even after having seen a polynomial number of certificates

- **Unforgeablity:** We require that the basic signature scheme satisfies at least the notion of existential unforgeability under weak message attack.

2.8 Groth-Sahai Commitments.

We will follow the Groth-Sahai methodology for SXDH-based commitment in the SXDH setting. The commitment key consists of $\mathbf{u} \in \mathbb{G}_1^{2\times 2}$ and $\mathbf{v} \in \mathbb{G}_2^{2\times 2}$. There exist two initializations of the parameters; either in the perfectly binding setting, or in the perfectly hiding one. Those initializations are indistinguishable under the SXDH assumption which will be used in the simulation. We denote by $\mathcal{C}(X)$ a commitment of a group element X. An element is always committed in the group ($\mathbb{G}_1$ or $\mathbb{G}_2$) it belongs to. If one knows the commitment key in the perfectly binding setting, one can extract the value of X, else it is perfectly hidden. We note $\mathcal{C}^{(1)}(x)$ a commitment of a scalar x embedded in $\mathbb{G}_1$ as g_1^x. If one knows the commitment key in the perfectly binding setting, on can extract the value of g_1^x else x is perfectly hidden. The same things can be done in $\mathbb{G}_2$, if we want to commit a scalar, embedding it in $\mathbb{G}_2$.

Proofs. Under the SXDH assumption, the two initializations of the commitment key (perfectly binding or perfectly hiding) are indistinguishable. The former provides perfectly sound proofs, whereas the latter provides perfectly witness hiding proofs. A Groth-Sahai proof, is a pair of elements $(\pi, \theta) \in \mathbb{G}_1^{2\times 2} \times \mathbb{G}_2^{2\times 2}$. These elements are constructed to help in verifying pairing relations on committed values. Being able to produce a valid pair implies knowing plaintexts verifying the appropriate relation.

We will use three kinds of relations:

- pairing products equation which requires 4 extra elements in each group;

- multi-scalar multiplication which requires 2 elements in one group and 4 in the other;

- quadratic equations which only requires 2 elements in each group.

If some of these equations are linear, some of the extra group elements are not needed, which leads to further optimizations.

In the following, we will generate two Common Reference Strings to handle commitments and proofs under this methodology through the following algorithm:

- $\mathsf{GS.KeyGen}(\mathsf{gk})$: generates two commitment keys, and the associated extraction key xk if it exists. In our protocol, ck_B will provide perfectly binding commitments in both group and while ck_H will provide perfectly hiding commitments in $\mathbb{G}_2$.Both commitment keys are added to the Common Reference String crs.

- $\mathsf{C.Commit}(\mathsf{ck}_*, A)$: allows to commit to an element A under the key ck_*, this produces a commitment, and the randomness r used for the commitment.

- $\mathsf{GS.Prove}(E, (\mathcal{C}, \mathsf{ck}_*))$: generates a Zero-Knowledge Groth-Sahai Proof of Knowledge π, that the plaintexts committed in $\mathcal{C}$ under ck_* satisfy the equation E. Such proofs require the commitment randomness r, and can only be done directly if the elements are committed under the same key [3].

- $\mathsf{GS.Verify}(\pi)$: verifies the validity of the proof π. To lighten the notation, we assume that a proof π implicitly contains the equation E and the commitment $(\mathcal{C}, \mathsf{ck}_*)$. Additionally, we will denote by $\mathsf{GS.Verify}(\pi_1, \pi_2, \ldots)$ the verification of several proofs, this can be done sequentially or using a batch technique, as presented in [BFI+10].

- $\mathsf{GS.Re\text{-}Randomize}(\mathcal{C}, \mathsf{ck}_*, \pi)$: re-randomizes the commitment $\mathcal{C}$, using $\mathsf{C.Re\text{-}Randomize}(\mathcal{C}, \mathsf{ck}_*)$, and then adapts the proof π. This step does not require the knowledge of the commitment randomness.

- $\mathsf{C.Extract}(\mathcal{C}, \mathsf{xk})$: extracts the plaintext A from $\mathcal{C}$ if $\mathcal{A}$ was committed in $\mathcal{C}$ under a binding key. The soundness of proof generated by Groth and Sahai methodology implies that if $\mathsf{GS.Verify}(E, (\mathcal{C}, \mathsf{ck}_B), \pi)$ holds, then we have that $\mathsf{C.Extract}(\mathcal{C}, \mathsf{xk})$ verifies the equation E.

2.9 A Classical Trick

Our construction will rely on a classical trick used on Groth-Sahai proofs; in several e-cash papers [CG07, BCKL09, LV09, FV10, BCF+11], the construction needs an anonymity property where the adversary should not be able to get any information on a coin while a judge should be able to extract the information while in the same CRS. Another application around this idea was presented by Fischlin, Libert and Manulis in [FLM11] where the authors used it to provide a non-interactive technique to commit to elements in the UC framework.

In those cases, the solution proposed, is to commit twice to the value X, once with a perfectly binding commitment key, and once with a perfectly hiding key, and then proving the committed value X is the same in both. (While this is necessarily true because of the perfectly hiding commitment, under the Co-Soundness of Groth-Sahai proof, this is hard to do without the knowledge of trapdoors in the commitment key). To then use this X in the rest of the scheme, one simply builds proof using the perfectly hiding commitment.

We will employ exactly this trick in the context of group signatures. Most schemes rely on a q-assumption, or even an interactive assumption, to prove anonymity of the scheme. We use this trick to be able to prove anonymity without using either, thus achieving our goal.

[3] This means that we have to be careful that, for a given equation, our commitments in $\mathbb{G}_2$ are done solely with ck_H or solely with ck_B.

algorithm $\mathsf{KeyGen}(1^k)$	**algorithm** Issue
$\mathsf{gk} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow_{\$} \mathsf{Gen}(1^k)$ $\ell = poly(\lambda)$ $\gamma \in_R \mathbb{Z}_p$, $\Gamma = g_2^{\gamma}$ $k_1 \in_R \mathbb{G}_1$ $\mathbf{u} \in_R \mathbb{G}_1^{\ell}$ //description of $\mathcal{F}$ return $(\mathsf{ak}, \mathsf{ck}) = ((\mathsf{gk}, \Gamma, k_1, \mathcal{F}), (\mathsf{ak}, \gamma))$	User Issuer $y' \in_R \mathbb{Z}_p$ $\xrightarrow{g_1^{y'}, g_2^{y'}}$ $x, y'' \in_R \mathbb{Z}_p$ $A = (k_1 g_1^{y'} g_1^{y''})^{\frac{1}{x+\gamma}}$ $\mathsf{cert} = (g_1^x, g_2^x, A)$ $\mathsf{sk} = y' + y''$ $\xleftarrow{y'', \mathsf{cert}}$ $\mathsf{pk} = (g_1^{\mathsf{sk}}, g_2^{\mathsf{sk}})$ return $(\mathsf{pk}, \mathsf{cert}, \mathsf{sk})$ return $(\mathsf{pk}, \mathsf{cert})$
algorithm $\mathsf{Sign}(\mathsf{pk}, \mathsf{sk}, m)$	**algorithm** $\mathsf{Verify}(\mathsf{pk}, \mathsf{ak}, \mathsf{cert}, m, \sigma)$
$s \in_R \mathbb{Z}_p$ $\sigma_1 = h^{\mathsf{sk}} \mathcal{F}(m)^s$ $\sigma_2 = g_1^s$ $\sigma_3 = g_2^s$ return $\sigma = (\sigma_1, \sigma_2, \sigma_3)$	return 1 if $e(\mathsf{cert}_1, g_2) = e(g_1, \mathsf{cert}_2) \wedge$ $e(\mathsf{cert}_3, \mathsf{ak}_2 \mathsf{pk}_2) = e(k_1, g_2) e(g_1, \mathsf{pk}_2) \wedge$ $e(\mathsf{cert}_2, g_2) = e(g_1, \sigma_3) \wedge$ $e(\sigma_2, g_2) = e(g_1, \sigma_3) \wedge$ $e(\sigma_1, g_2) = e(h_1, \mathsf{pk}_2) e(\mathcal{F}(m), \sigma_3)$ else return 0

Figure 4: The Delerablée-Pointcheval Certified Waters Signature Scheme.

3 Group Signature without q-Assumptions

3.1 Asymmetric Waters Signature Scheme

We briefly recall the asymmetric Waters signature scheme, which is a core building block in our scheme [Duc10, BFPV11].

- $\mathsf{Setup}(1^k)$: The scheme needs a (asymmetric) pairing-friendly environment $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$, where $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an admissible bilinear map, for groups $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$, of prime order p, generated by g_1, g_2 and $g_t = e(g_1, g_2)$ respectively. We will sign messages $M = (M_1, \ldots, M_k) \in \{0,1\}^k$. To this aim, we need a vector $\vec{u} = (u_0, \ldots, u_k) \overset{\$}{\leftarrow} \mathbb{G}_1^{k+1}$, and for convenience, we denote the *Waters Hash* as $\mathcal{F}(M) = u_0 \prod_{i=1}^{k} u_i^{M_i}$. We also need an additional generator $h_1 \overset{\$}{\leftarrow} \mathbb{G}_1$. The global parameters param consist of all these elements $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2, h_1, \vec{u})$.

- $\mathsf{KeyGen}(\mathsf{param})$: Chooses a random scalar $x \overset{\$}{\leftarrow} \mathbb{Z}_p$, which defines the public key as $(X_1, X_2) = (g_1^x, g_2^x)$, and the secret key as $\mathsf{sk} = Y = h_1^x$.

- $\mathsf{Sign}(\mathsf{sk} = Y, M; s)$: For some random $s \overset{\$}{\leftarrow} \mathbb{Z}_p$, define the signature as $\sigma = \big(\sigma_1 = Y(\mathcal{F}(M))^s, \sigma_2 = g_1^{-s}, \sigma_3 = g_2^{-s}\big)$.

- $\mathsf{Verif}((X_1, X_2), M, \sigma)$: Checks whether $e(\sigma_1, g_2) \cdot e(\mathcal{F}(M), \sigma_3) = e(h_1, X_2)$, and $e(\sigma_2, g_2) = e(g_1, \sigma_3)$.

3.2 Certified Signature Scheme

We will use a slight variant of the signature scheme due to Waters [Wat05] using certificates as described by Delerablée and Pointcheval [DP06], which we refer to as the DPW scheme from here on. We describe the scheme in Figure 4

The DPW Scheme was shown to be secure under the q-DHSDH, and CDH^+ assumptions. In Section 3.2, we will present a modification of this scheme so that we can prove the security under the DHSDH and CDH^+ assumptions.

```
algorithm KeyGen($1^k$)
$\mathsf{gk} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow_\$ \mathsf{Gen}(1^\lambda)$
$\gamma \in_R \mathbb{Z}_p$, $\Gamma = g_2^\gamma$
$k_1 \in_R \mathbb{G}_1, h_2, \mathcal{F} \in_R \mathbb{G}_2^{\ell+2}$
$(\mathsf{ck}_B, \mathsf{ck}_H, \mathsf{xk}) \leftarrow_\$ \mathsf{GS.KeyGen}(\mathsf{gk})$
$(\mathsf{ak}) = (\mathsf{gk}, \Gamma, k, \mathcal{F}, \mathsf{ck}_B, ck_H, \mathsf{crs})$
$(\mathsf{ck}) = (\mathsf{ak}, \gamma)$
return $(\mathsf{ak}, \mathsf{ck})$
algorithm Sign($\mathsf{pk}, \mathsf{sk}, m$)
$s \in_R \mathbb{Z}_p$
$\sigma_1 = h_2^{\mathsf{sk}} \mathcal{F}(m)^s$
$\sigma_2 = g_1^s$
$\sigma_3 = g_2^s$
return $\sigma = (\sigma_1, \sigma_2, \sigma_3)$
algorithm Verify($\mathsf{pk}, \mathsf{ak}, \mathsf{cert}, m, \sigma$)
return 1 if
    $\mathsf{GS.Verify}(\pi) == 1 \wedge$
    $e(\sigma_2, g_2) == e(g_1, \sigma_3) \wedge$
    $e(g_1, \sigma_1) == e(\mathsf{pk}_1, h_2) e(\sigma_2, \mathcal{F}(m))$
return 0 else
```

```
algorithm Join/Issue
User                                                   Issuer
$y' \in_R \mathbb{Z}_p$          $\xrightarrow{g_1^{y'}}$
                                                       $y'', x \in_R \mathbb{Z}_p$
                                                       $A = (k_1 g_1^{y'} g_1^{y''})^{\frac{1}{x+\gamma}}$
                                                       $\alpha = \mathsf{C.Commit}(\mathsf{ck}_B, A)$
                                                       $\chi = \mathsf{C.Commit}(\mathsf{ck}_H, g_2^x)$
                                                       $X_1 = \mathsf{C.Commit}(ck_B, g_1^x)$
                                                       $X_2 = \mathsf{C.Commit}(ck_B, g_2^x)$
                                                       $\pi_1 = \mathsf{GS.Prove}(\alpha, \chi)$
                                                       $\pi_2 = \mathsf{GS.Prove}(X_1, \chi)$
                                                       $\pi_3 = \mathsf{GS.Prove}(X_1, X_2)$
                                                       $X = (X_1, X_2)$
                                                       $\pi = (\pi_1, \pi_2, \pi_3)$
                                                       $\mathsf{cert} = (\alpha, \chi, X, \pi)$
                                 $\xleftarrow{y'', \mathsf{pk}, \mathsf{cert}}$      $\mathsf{pk} = (g_1^{y_i'} g_1^{y_i''}, g_2^{y_i'} g_2^{y_i''})$
$\mathsf{sk} = y' + y''$
if $\mathsf{pk} \neq (g_1^{\mathsf{sk}}, g_2^{\mathsf{sk}})$
    return $\perp$
if $\mathsf{GS.Verify}(\pi) \neq 1$
    return $\perp$
else
    return $(\mathsf{pk}, \mathsf{cert}, \mathsf{sk})$                    return $(\mathsf{pk}, \mathsf{cert})$
```

Figure 5: The Delerablée-Pointcheval Certified Waters Signature Scheme with Commitments.

We now present our variant of the Delerablée-Pointcheval Certified Waters Signature Scheme, using commitments, which we will call the DPWC scheme from here on. In the DPWC scheme, instead of sending the certificate, the certificate authority will send commitments to the certificate, and a proof that the certificate is well-formed. The receiver must now verify the proof of well-formedness instead of the certificate. We can now show that the hardness of forging a certificate can be reduced to the soundness of the commitment scheme, which in turn is based upon the SXDH. Due to technical reasons, we need two common reference strings, one which is perfectly hiding and one which is perfectly binding. We present the DPWC scheme in Figure 5.

Theorem 1. *The DPWC scheme is a certified signature scheme with perfect correctness for all messages $m \in \{0,1\}^\ell$. It is unfakeable under the DHSDH assumption and unforgeable under the CDH$^+$ assumption.*

Proof. The correctness of the scheme follows from the correctness of the Waters signatures, the Delerablée-Pointcheval certification and the correctness of the Groth-Sahai NIZK scheme.

Lemma 1. *If an adversary can (q', t', ε')-break the unfakeability of the scheme, then we can (t, ε)-solve the Double Hidden Strong Diffie-Hellman (DHSDH) problem, with*

$$t \approx t' \qquad \text{and} \qquad \varepsilon = \varepsilon'.$$

Proof. We receive as an initial input the DHSDH challenge of the form $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, k_1, g_2, g_2^\gamma, e)$. We then generate new commitment keys and keys for the proof system, thus giving us the extraction keys for the commitments and the ability to simulate the proofs, using the CRS trapdoor. We send the challenge along with the commitment keys and public parameters for the proof system to the adversary. Note that these form a valid DPWC public key. The adversary will then make q queries to the KeyReg oracle, which will allow it to act as a user and receive a key and certificate. We pick random values y'', x as before. Now since we do not possess a valid certification key, we must simulate the certificate.

To simulate a certificate, we pick a random value A and commit to it. We then simulate the zero-knowledge proofs of well-formedness using the simulation trapdoor. Since we never send the A values in the clear, under the indistinguishability of the commitments, the adversary will not distinguish the difference. Hence, we have a perfect simulation of the scheme. The adversary will then submit a forged certificate cert^*, a public key pk^*, message m^*, signature σ^*. We first verify the certified signature. If the adversary has produced a valid certified signature, then both the certificate and signature must be correctly formed. Using the extraction key on the binding commitments, we are able to extract the value A^* from the forged certificate, along with the values $g_1^{x^*}, g_2^{x^*}, g_1^{y^*}, g_2^{y^*}$. We then submit $(g_1^{x^*}, g_2^{x^*}, g_1^{y^*}, g_2^{y^*}, A^*)$ our solution to the DHSDH problem. We note that we win with exactly the same probability as the adversary. □

The unforgeability of the DPW scheme was shown to hold by Blazy et al. in [BFPV11]. We include their statement in here for completeness.

Lemma 2. *Given an adversary can (q', t', ε')-break the unforgeability of the scheme, then we can (t, ε)-solve the Advanced Computational Diffie-Hellman (CDH^+) problem, with*

$$t \approx t' \qquad \text{and} \qquad \varepsilon = \Theta(\varepsilon'/q'\sqrt{\ell})$$

where ℓ is the length of our messages.

Proof. The proof can be found in [BFPV11, Appendix D]. □

This completes the proof. □

3.3 Our Group Signature Construction

Now that we have the DPWC scheme, we can begin to construct our group signature scheme. The naïve approach would be to simply to provide each user with a DPWC certificate and key pair and use those to produce in the normal manner. However, we can immediately see that these signatures are no longer unlinkable, as a all the signatures from any user would have their DPWC certificate attached to it, along with the corresponding public key. This is remedied by treating the DPWC public key as part of the certificate and committing it as well during the Join/Issue protocol. When signing the user will re-randomize these commitments and the proofs.

However, the signatures are still linkable. This is due to the fact that given a pair of Waters signatures, one can check if they are signed using the same key or not. To resolve this problem, we use an idea due to Fischlin [Fis06] that a commitment to a signature and proof of well-formedness implies a signature. We apply this idea to the Waters signature and hence get commitments of our signature elements and proofs of their well-formedness. This "committed" signature and our re-randomized committed certificate and the relevant proofs are then sent as the group signature.

The Open procedure will use the extraction key xk to extract the certificate from a signature and then check if there is a registry entry with the same certificate. If a matching certificate is found, we know that this user must have made that signature and thus it can be opened to their index. To prove that the opening was done correctly, we simply prove that the commitment stored in the registry and the one commitment from the signature contain the same certificate.

Theorem 2. *The scheme described in Figure 6 is a group signature scheme with perfect correctness. The scheme satisfies anonymity, traceablity and non-frameability under the SXDH, DHSDH and CDH^+ assumptions.*

Proof. We will now provide proof sketches of the statements individually. We omit correctness, as it is clear.

```
algorithm KeyGen(1^k)
    $\mathsf{gk} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e) \leftarrow_{\$} \mathsf{Gen}(1^k)$
    $\gamma \in_R \mathbb{Z}_p$, $\Gamma = g_2^{\gamma}$
    $k_1 \in_R \mathbb{G}_1, h_2, \mathcal{F} \in_R \mathbb{G}_2^{\ell+2}$
    $(\mathsf{ck}_B, \mathsf{ck}_H, \mathsf{xk}) \leftarrow_{\$} \mathsf{GS.KeyGen}(\mathsf{gk})$
    $(\mathsf{ak}) = (\mathsf{gk}, \Gamma, k, \mathcal{F}, \mathsf{ck}_B, ck_H, \mathsf{crs})$
    $(\mathsf{ck}) = (\mathsf{ak}, \gamma)$
    return $(\mathsf{ak}, \mathsf{ck})$

algorithm Sign(gpk, sk, m)
    $s \in_R \mathbb{Z}_p$
    $\mathsf{cert}'_i \leftarrow_{\$} \mathsf{GS.Re\text{-}Randomize}(\mathsf{cert}_i)$
    $Y_1 = \mathsf{GS.Re\text{-}Randomize}(ck_B, g_1^{sk_i}, \pi'_{i,1})$
    $Y_2 = \mathsf{C.Commit}(ck_B, g_2^{sk_i})$
    $\sigma_1 = \mathsf{C.Commit}(h_2^{\mathsf{sk}_i}\mathcal{F}(m)^s)$
    $\sigma_2 = g_1^s$
    $\sigma_3 = g_2^s$
    $\tilde{\pi}_1 = \mathsf{GS.Prove}(Y_1, Y_2)$
    $\tilde{\pi}_2 = \mathsf{GS.Prove}(\sigma_1, \sigma_2, Y_2)$
    return $\sigma = (\sigma_1, \sigma_2, \sigma_3, \mathsf{cert}'_i, Y_1, Y_2, \tilde{\pi}_1, \tilde{\pi}_2)$
```

```
algorithm Issue(usk[i], ik)
    User                                                    Issuer
    $y'_i \in_R \mathbb{Z}_p$          $\xrightarrow{g^{y'_i}}$
                                                            $y''_i, x_i \in_R \mathbb{Z}_p$
                                                            $A_i = (k_1 g_1^{y'} g_1^{y''})^{\frac{1}{x_i+\gamma}}$
                                                            $\alpha_i = \mathsf{C.Commit}(\mathsf{ck}_B, A_i)$
                                                            $\chi_i = \mathsf{C.Commit}(\mathsf{ck}_H, g_2^{x_i})$
                                                            $X_{i,1} = \mathsf{C.Commit}(ck_B, g_1^{x_i})$
                                                            $X_{i,2} = \mathsf{C.Commit}(ck_B, g_2^{x_i})$
                                                            $\pi_{i,1} = \mathsf{GS.Prove}(\alpha_i, \chi_i)$
                                                            $\pi_{i,2} = \mathsf{GS.Prove}(X_{i,1}, \chi_i)$
                                                            $\pi_{i,3} = \mathsf{GS.Prove}(X_{i,1}, X_{i,2})$
                                                            $X_i = (X_{i,1}, X_{i,2})$
                                                            $\pi_i = (\pi_{i,1}, \pi_{i,2}, \pi_{i,3})$
                                                            $\mathsf{cert}_i = (\alpha_i, \chi_i, X_i, \pi_i)$
    $\mathsf{sk}_i = y'_i + y''_i$     $\xleftarrow{\mathsf{cert}_i, y''_i}$
    if $\mathsf{GS.Verify}(\pi_i) \neq 1$
        return $\perp$
    $s_i = \mathsf{Sign}(\mathsf{usk}[i], \mathsf{cert}_i)$     $\xrightarrow{s_i}$
    return $(\mathsf{cert}_i, \mathsf{sk}_i)$                  $\mathsf{Reg}[i] = (i, \mathsf{pk}[i], \mathsf{cert}_i, s_i)$
```

```
algorithm Open(gpk, ok, σ)
    $\mathsf{cert}^* \leftarrow \mathsf{C.Extract}(\mathsf{xk}, cert'_i)$
    for($i \in [1, n]$)
        $\widehat{cert} \leftarrow \mathsf{C.Extract}(\mathsf{ok}, \mathsf{cert}_i)$
        $\widehat{x} \leftarrow \mathsf{C.Extract}(\mathsf{ok}, \mathsf{Reg}[i]_4)$
        if $\widehat{cert} == \mathsf{cert}^*_1$
            $\tau = \mathsf{GS.Prove}(cert, \mathsf{cert}^*)$
            return $(i, \tau)$
    endfor
    return $(0, \perp)$
```

```
algorithm Verify(gpk, m, σ)
    return $\mathsf{GS.Verify}(\pi'_i, \tilde{\pi}_1, \tilde{\pi}_2) \wedge e(\sigma_2, g_2) == e(g_1, \sigma_3)$

algorithm Judge(pk, ak, cert, m, σ, τ)
    return $\mathsf{GS.Verify}(\tau)$
```

Figure 6: The Group Signature Scheme.

Lemma 3. *The scheme described in Figure 6 is a group signature scheme with (t, ϵ)-anonymity, assuming the Symmetric External Diffie-Hellman Assumption is $(t', \varepsilon_{\mathsf{SXDH}})$-hard, with*

$$t' \approx t \qquad \text{and} \qquad \epsilon \leqslant \varepsilon_{\mathsf{SXDH}}.$$

Proof. For the anonymity game, we that the issuer is fully corrupted, but the Opener is uncorrupted. This means that we give full control of the issuer, including the secret key to the adversary. However, the Opener is uncorrupted, thus we maintain full control of the opening key. In the initial game, we generate both ik and ok honestly. We give the adversary control of ik.

We now modify our game such that the Opener's key is no longer binding but in fact, hiding. This is indistinguishable to the adversary under the SXDH assumption. In particular, this means that any information contained in the certificates is hidden. This means that the game where the challenge bit $b = 0$ gives the adversary the same information as when $b = 1$. Thus, the adversary has no advantage. As the advantage in this game differs from the previous game by at most the advantage of an adversary against the SXDH, the adversary has a total advantage $\epsilon \leqslant \varepsilon_{\mathsf{SXDH}}$. □

Lemma 4. *The scheme described in Figure 6 is a group signature scheme with (t, ϵ)-traceability, assuming* *the Symmetric External Diffie-Hellman Assumption is* $(t', \varepsilon_{\mathsf{SXDH}})$*-hard and the Double Strong Hidden Diffie-Hellman Assumption is* $(t', \varepsilon_{\mathsf{DHSDH}})$*-hard, with*

$$t' \approx t \qquad \textit{and} \qquad \epsilon \leqslant 2 \cdot \varepsilon_{\mathsf{SXDH}} + \varepsilon_{\mathsf{DHSDH}}.$$

Proof. In the traceability game, we have a partially corrupted Opener and an uncorrupted Issuer. Thus, the adversary will know ok but the Issuer, will still behave in the prescribed manner. Initially we set up the game such that all keys are generated as expected, and we give the issuer key to the adversary. Since we have all the requisite keys, we can execute the requisite oracles in the standard manner.

Since the Opener is only partially corrupted, they will still perform all the openings faithfully and generate valid proofs of that. If a signature opens and is traced to a user, then the proof produced will be accepted by the Judge algorithm. If the adversary can produce a false proof, then they will have broken the soundness of the proof scheme. They can do this with an advantage at most $\varepsilon_{\mathsf{SXDH}}$. Thus, the adversary must produce a false certificate.

Our next modification is the CRS generation. We switch now to a simulating CRS which provides perfectly hiding commitments. The difference in advantage the adversary has in this game and the previous game is at most $\varepsilon_{\mathsf{SXDH}}$. As before, we execute all the oracles in the normal manner.

For our final game, we now simulate our issue protocol in the same manner as we did for the unfakeability proof of the DPWC scheme in Lemma 1. Thus, we have that any adversary that can produce a fake certificate will break the DHSDH, which he can do with advantage at most $\varepsilon_{\mathsf{DHSDH}}$.
Combining these gives us $\epsilon \leqslant 2 \cdot \varepsilon_{\mathsf{SXDH}} + \varepsilon_{\mathsf{DHSDH}}$, as required. □

Lemma 5. *The scheme described in Figure 6 is a group signature scheme with* (t, ϵ)*-non-frameability, assuming the Symmetric External Diffie-Hellman problem is* $(t', \varepsilon_{\mathsf{SXDH}})$*-hard, the Double Strong Hidden Diffie-Hellman problem is* $(t', \varepsilon_{\mathsf{DHSDH}})$*-hard and the Advanced Computational Diffie-Hellman problem is* $(t', \varepsilon_{\mathsf{CDH+}})$*-hard, with*

$$t \approx t' \qquad \textit{and} \qquad \epsilon \leqslant \varepsilon_{\mathsf{SXDH}} + \varepsilon_{\mathsf{DHSDH}} + \varepsilon_{\mathsf{CDH+}}$$

Proof. For an adversary to win the non-framebility game, they must produce a signature which will be correctly attributed to an honest user who did not produce this signature. To achieve this, an adversary must provide:

1. A valid signature under the user's public key

2. A valid proof that the signature is valid under the public key in the committed certificate

3. A valid committed certificate, with proofs

Item 3 can be easily obtained by the adversary as he is able to fully corrupt both the Opener and Issuer and obtain the correct certificates and the corresponding proof from there. Thus we now need to only consider how the Adversary produces the other two components. To this end, we consider two types of Adversaries, namely Type I and Type II adversaries, which win by generating item 1, respectively item 2, listed above.

Type I Non-Frameability Adversaries The first type of adversary, which we call $\mathcal{NF}_1$, is an adversary who wins in the non-frameability game by forging a signature for an honest user. Once the adversary has a valid forgery, he can easily obtain a committed certificate and re-randomize that. Having both these, the adversary can then honestly generate a proof that the signature is valid. We see that this signature is a valid group signature and will indeed be attributed to the targeted user. If a $\mathcal{NF}_1$ to succeeds, it can be turned into and adversary against the unforgeability of the DPWC signature scheme. We are able to generate all the keys, except for the signing keys. We use the oracle from the unforgeability game to produce the required signatures. We can simply use the adversary's forgery as our own and thus win the DPWC unforgeability game, which in turn would break the Advanced Computational Diffie-Hellman Assumption.

Component	σ_1	σ_2	α	χ	X_1	X_2	π_1	π_2	π_3	Y_1	Y_2	$\tilde{\pi}_1$	$\tilde{\pi}_2$	TOTAL
$\mathbb{G}$	3	1	3	3	3	3	≈ 13	≈ 13	2	3	3	2	3	≈ 55

Component	σ_1	σ_2	σ_3	α	χ	X_1	X_2	π_1	π_2	π_3	Y_1	Y_2	$\tilde{\pi}_1$	$\tilde{\pi}_2$	TOTAL
$\mathbb{G}_1$	0	1	0	2	0	2	0	2	2	2	2	0	2	4	19
$\mathbb{G}_2$	2	0	1	0	2	0	2	4	2	2	0	2	2	4	23

Type II Non-Frameability Adversaries The second type of adversary, which we call $\mathcal{NF}_2$, is an adversary who wins in the non-frameability game by creating a false proof for an incorrect signature. The adversary will try to prove that an arbitrary invalid signature is actually a correct signature and then proceeds to produce a proof that this invalid "signature" is indeed valid. If an $\mathcal{NF}_2$ adversary succeeds, they will have produced a NIZK proof on a false statement, which breaks the Soundness of the Groth-Sahai proof system, and thus the Symmetric External Diffie-Hellman Assumption.

In addition to the above types of adversary, we must also consider an adversary who fakes a certificate for the targeted user and then performs a Type I or Type II attack. The adversary in this game has the capability to write to the registry and hence can replace the user's old certificate with their faked one. After this, the user must perform a Type I or Type II attack as described above. Here we see that the adversary must first fake a certificate, hence breaking the unfakeability of the DPWC certified signature scheme, and thus the Double String Hidden Diffie-Hellman Assumption. After this, the adversary will proceed as a $\mathcal{NF}_1$ or $\mathcal{NF}_2$ and thus additionally break the Advanced Computational Diffie-Hellman Assumption or the Symmetric External Diffie-Hellman Assumption.

Thus, we get that $\epsilon \leqslant \varepsilon_{\mathsf{SXDH}} + \varepsilon_{\mathsf{CDH}^+} + \varepsilon_{\mathsf{DHSDH}}$, as required. □

This completes the proof. □

4 A Linear Version of Our Construction

Our previous scheme can be directly transposed in a symmetric group, with Linear Commitments.

Theorem 3. *The scheme described in Figure 6 is a group signature scheme with perfect correctness. The scheme satisfies anonymity, traceability and non-frameability under the DHSDH, DLin and CDH assumptions.*

This can be proven following the idea of the asymmetric instantiations. We omit the proofs, as they are of minimal interest.

On the efficiency of this scheme There is always a trade-off in efficiency while instantiating on a symmetric group a scheme designed for an asymmetric one. Verifying that two elements have the same discrete logarithm is way more efficient in a DLin setting because this becomes a linear equation while being a quadratic one in SXDH. However we will have equations with two CRS involved for the same group, and that is quite inefficient (approximately 13 elements for each proof).

The table above gives a rough estimation of the cost of the symmetric instantiation of our scheme, while not being so efficient it is still in the same order of magnitude as existing group signatures schemes. Once again our hypotheses are neither interactive nor relying on q-assumptions.

5 Efficiency Comparison and Conclusions

We now look at the efficiency of our scheme in comparison to the state of the art in signature schemes. We begin with a look at the exact size of our signatures. We list the size of each component of our signature in the table below.

```
algorithm KeyGen($1^k$)

$\mathsf{gk} = (p, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow_\$ \mathsf{Gen}(1^k)$
$\gamma \in_R \mathbb{Z}_p$, $\Gamma = g^\gamma$
$k, h, g_2 \in_R \mathbb{G}, \mathcal{F} \in_R \mathbb{G}^{\ell+1}$
$(\mathsf{ck}_B, \mathsf{ck}_H, \mathsf{xk}) \leftarrow_\$ \mathsf{GS.KeyGen}(\mathsf{gk})$
$(\mathsf{ak}) = (\mathsf{gk}, \Gamma, k, h, g_2, \mathcal{F}, \mathsf{ck}_B, ck_H, \mathsf{crs})$
$(\mathsf{ck}) = (\mathsf{ak}, \gamma)$
return $(\mathsf{ak}, \mathsf{ck})$
```

```
algorithm Sign($\mathsf{gpk}, \mathsf{sk}, m$)

$s \in_R \mathbb{Z}_p$
$\mathsf{cert}'_i \leftarrow_\$ \mathsf{GS.Re\text{-}Randomize}(\mathsf{cert}_i)$
$Y_1 = \mathsf{GS.Re\text{-}Randomize}(ck_B, g^{sk_i}, \pi'_{i,1})$
$Y_2 = \mathsf{C.Commit}(ck_B, g_2^{sk_i})$
$\sigma_1 = \mathsf{C.Commit}(h^y \mathcal{F}(m)^s)$
$\sigma_2 = g^s$
$\tilde{\pi}_1 = \mathsf{GS.Prove}(Y_1, Y_2)$
$\tilde{\pi}_2 = \mathsf{GS.Prove}(\sigma_1, \sigma_2, Y_2)$
return $\sigma = (\sigma_1, \sigma_2, \mathsf{cert}'_i, Y_1, Y_2, \tilde{\pi}_1, \tilde{\pi}_2)$
```

```
algorithm Issue
User                                                   Issuer
$y'_1 \in_R \mathbb{Z}_p$                  $\xrightarrow{g^{y'_1}}$
                                                       $y''_i, x_i \in_R \mathbb{Z}_p$
                                                       $A_i = (k_1 g_1^{y'} g_1^{y''})^{\frac{1}{x_i+\gamma}}$
                                                       $\alpha_i = \mathsf{C.Commit}(\mathsf{ck}_B, A_i)$
                                                       $\chi_i = \mathsf{C.Commit}(\mathsf{ck}_H, g^{x_i})$
                                                       $X_{i,1} = \mathsf{C.Commit}(\mathsf{ck}_B, g^{x_i})$
                                                       $X_{i,2} = \mathsf{C.Commit}(\mathsf{ck}_B, g_2^{x_i})$
                                                       $\pi_{i,1} = \mathsf{GS.Prove}(\alpha_i, \chi_i)$
                                                       $\pi_{i,2} = \mathsf{GS.Prove}(X_{i,1}, \chi_i)$
                                                       $\pi_{i,3} = \mathsf{GS.Prove}(X_{i,1}, X_{i,2})$
                                                       $X_i = (X_{i,1}, X_{i,2})$
                                                       $\pi_i = (\pi_{i,1}, \pi_{i,2}, \pi_{i,3})$
                                                       $\mathsf{cert}_i = (\alpha_i, \chi_i, X_i, \pi_i)$
$\mathsf{sk}_i = y' + y''$                 $\xleftarrow{\mathsf{cert}_i, y''_i}$
if $\mathsf{GS.Verify}(\pi_i) \neq 1$
      return $\perp$
$s_i = \mathsf{Sign}(\mathsf{sk}[i], \mathsf{cert}_i)$   $\xrightarrow{s_i}$
return $(\mathsf{cert}_i, \mathsf{sk}_i)$                 $\mathsf{Reg}[i] = (i, \mathsf{pk}[i], \mathsf{cert}_i, s_i)$
```

```
algorithm Open($\mathsf{gpk}, \mathsf{ok}, \sigma$)
$\mathsf{cert}^* \leftarrow \mathsf{C.Extract}(\mathsf{xk}, cert'_i)$
for($i \in [1, n]$)
      $\widehat{cert} \leftarrow \mathsf{C.Extract}(\mathsf{ok}, \mathsf{cert}_i)$
      $\widehat{x} \leftarrow \mathsf{C.Extract}(\mathsf{ok}, \mathsf{Reg}[i]_4)$
      if $\widehat{cert} == \mathsf{cert}_1^*$
            $\tau = \mathsf{GS.Prove}(cert, \mathsf{cert}^*)$
            return $(i, \tau)$
endfor
return $(0, \perp)$
```

```
algorithm Verify($\mathsf{gpk}, m, \sigma$)

return $\mathsf{GS.Verify}(\pi'_i, \tilde{\pi}_1, \tilde{\pi}_2)$
```

```
algorithm Judge($\mathsf{pk}, \mathsf{ak}, \mathsf{cert}, m, \sigma, \tau$)

return $\mathsf{GS.Verify}(\tau)$
```

Figure 7: The Symmetric Group Signature Scheme.

Similar to the work of Libert, Peters and Yung [LPY15], we compare not only the number of group elements, but the bit sizes, assuming classical instantiations on Barreto-Naehrig Curves [BN06], (one group of size 256 bits, and one of size 512 bits), we consider the smaller group to be the one with more elements. In our comparison, we only consider the schemes in Type 3 groups.

As we can see from the table, our signature sizes are comparable to that of the other schemes, but under standard assumptions. In particular, we have fewer elements than the scheme of Libert, Peters and Yung [LPY15], with the same final signature size. Our signatures are slightly larger than the schemes based on q-assumption one, but not by a large margin. Additionally, this does not take into account that we rely on simpler assumptions. Hence, we do not have the same overheads to achieving higher levels of security. We believe that overall this technique allows quick and efficient transformation at minimal cost, and quite likely be extended to other protocols.

[3] The scheme in [LPY15] requires a chameleon hash function. For simplicity, we assume the DLin-based chameleon hash due to Hofheinz and Jager [HJ12], explicitly stated by Blazy et al. [BKKP15, Appendix A]

				Total Signature Size	
Scheme	Assumptions	$\mathbb{G}_1$	$\mathbb{G}_2$	Total Elements	Bitsize
Adapted Groth [Gro07, LPY15]	SXDLin, q-SDH, q-U$'$	27	12	39	13056
Kakvi [Kak10] (Scheme 3)	SXDLin, q-SDH, q-U3a	24	15	39	13824
Kakvi [Kak10] (Scheme 4)	SXDLin, q-SDH, q-U3b	16	23	39	14080
Blazy and Pointcheval [BP12]	CDH$^+$, q-DDHI, q-DHSDH	21	16	37	13568
Libert, Peter and Yung [LPY15]	SXDH, XDLin$_2$, DLin^3	33	14	47	15616
This Work	CDH$^+$, DHSDH	19	23	42	15616

Table 2: Comparison of Group Signature Schemes secure in the Standard Model.

Acknowledgements

We would like to thank the anonymous reviewers of INDOCRYPT 2020 for their comments. We would also like to thank Ruqayya Shaheed for her editorial comments. Part of this work was done while both authors were employed at the Ruhr University Bochum.

References

[ABCG16] Quentin Alamélou, Olivier Blazy, Stéphane Cauchie, and Philippe Gaborit. A code-based group signature scheme. Cryptology ePrint Archive, Report 2016/1119, 2016. `http://eprint.iacr.org/2016/1119`. Cited in Sec. 1.1

[ACJT00] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 255–270, Santa Barbara, CA, USA, August 20–24, 2000. Springer, Heidelberg, Germany. `doi:10.1007/3-540-44598-6_16`. Cited in Sec. 1.1

[AFG$^+$10] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 209–236, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-14623-7_12`. Cited in Sec. 1.1, 4

[AST02] Giuseppe Ateniese, Dawn Xiaodong Song, and Gene Tsudik. Quasi-efficient revocation in group signatures. In Matt Blaze, editor, *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 183–197. Springer, 2002. Cited in Sec. 1.1

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73, Interlaken, Switzerland, May 2–6, 2004. Springer, Heidelberg, Germany. `doi:10.1007/978-3-540-24676-3_4`. Cited in Sec. 2

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg, Germany. `doi:10.1007/11426639_26`. Cited in Sec. 1.1

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Heidelberg, Germany. `doi:10.1007/978-3-540-28628-8_3`. Cited in Sec. 1.1, 1, 3, 2.7

[BCC$^+$16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, and Jens Groth. Foundations of fully dynamic group signatures. In Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors, *ACNS 16: 14th International Conference on Applied Cryptography and Network Security*, volume 9696 of *Lecture Notes in Computer Science*, pages 117–136, Guildford, UK, June 19–22, 2016. Springer, Heidelberg, Germany. `doi:10.1007/978-3-319-39555-5_7`. Cited in Sec. 1.1

[BCF$^+$11] Olivier Blazy, Sébastien Canard, Georg Fuchsbauer, Aline Gouget, Hervé Sibert, and Jacques Traoré. Achieving optimal anonymity in transferable e-cash with a judge. In Abderrahmane Nitaj and David Pointcheval, editors, *AFRICACRYPT 11: 4th International Conference on Cryptology in Africa*, volume 6737 of *Lecture Notes in Computer Science*, pages 206–223, Dakar, Senegal, July 5–7, 2011. Springer, Heidelberg, Germany. Cited in Sec. 2.9

[BCKL09] Mira Belenkiy, Melissa Chase, Markulf Kohlweiss, and Anna Lysyanskaya. Compact e-cash and simulatable VRFs revisited. In Hovav Shacham and Brent Waters, editors, *PAIRING 2009: 3rd International Conference on Pairing-based Cryptography*, volume 5671 of *Lecture Notes in Computer Science*, pages 114–131, Palo Alto, CA, USA, August 12–14, 2009. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-03298-1_9`. Cited in Sec. 2.9

[BFI$^+$10] Olivier Blazy, Georg Fuchsbauer, Malika Izabachène, Amandine Jambert, Hervé Sibert, and Damien Vergnaud. Batch Groth-Sahai. In Jianying Zhou and Moti Yung, editors, *ACNS 10: 8th International Conference on Applied Cryptography and Network Security*, volume 6123 of *Lecture Notes in Computer Science*, pages 218–235, Beijing, China, June 22–25, 2010. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-13708-2_14`. Cited in Sec. 2.8

[BFPV11] Olivier Blazy, Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Signatures on randomizable ciphertexts. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011: 14th International Conference on Theory and Practice of Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 403–422, Taormina, Italy, March 6–9, 2011. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-19379-8_25`. Cited in Sec. 5, 3.1, 3.2, 3.2

[BFPW07] Alexandra Boldyreva, Marc Fischlin, Adriana Palacio, and Bogdan Warinschi. A closer look at PKI: Security and efficiency. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *PKC 2007: 10th International Conference on Theory and Practice of Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 458–475, Beijing, China, April 16–20, 2007. Springer, Heidelberg, Germany. `doi:10.1007/978-3-540-71677-8_30`. Cited in Sec. 2.7

[BK15] Olivier Blazy and Saqib A. Kakvi. Group signatures without q-assumptions. Cryptology ePrint Archive, Report 2015/858, 2015. `http://eprint.iacr.org/2015/858`. Cited in Sec. 1.2

[BKKP15] Olivier Blazy, Saqib A. Kakvi, Eike Kiltz, and Jiaxin Pan. Tightly-secure signatures from chameleon hash functions. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 256–279, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer, Heidelberg, Germany. `doi:10.1007/978-3-662-46447-2_12`. Cited in Sec. 3

[BMV08] Emmanuel Bresson, Jean Monnerat, and Damien Vergnaud. Separation results on the "one-more" computational problems. In Tal Malkin, editor, *Topics in Cryptology – CT-RSA 2008*, volume 4964 of *Lecture Notes in Computer Science*, pages 71–87, San Francisco, CA, USA, April 7–11, 2008. Springer, Heidelberg, Germany. `doi:10.1007/978-3-540-79263-5_5`. Cited in Sec. 1.1

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes* *in Computer Science*, pages 614–629, Warsaw, Poland, May 4–8, 2003. Springer, Heidelberg, Germany. doi:10.1007/3-540-39200-9_38. Cited in Sec. 1

[BN06] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In Bart Preneel and Stafford Tavares, editors, *SAC 2005: 12th Annual International Workshop on Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331, Kingston, Ontario, Canada, August 11–12, 2006. Springer, Heidelberg, Germany. doi:10.1007/11693383_22. Cited in Sec. 5

[BNPS02] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. In Paul F. Syverson, editor, *FC 2001: 5th International Conference on Financial Cryptography*, volume 2339 of *Lecture Notes in Computer Science*, pages 319–338, Grand Cayman, British West Indies, February 19–22, 2002. Springer, Heidelberg, Germany. Cited in Sec. 1.1

[Boy08] Xavier Boyen. The uber-assumption family (invited talk). In Steven D. Galbraith and Kenneth G. Paterson, editors, *PAIRING 2008: 2nd International Conference on Pairing-based Cryptography*, volume 5209 of *Lecture Notes in Computer Science*, pages 39–56, Egham, UK, September 1–3, 2008. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-85538-5_3. Cited in Sec. 1.1

[BP12] Olivier Blazy and David Pointcheval. Traceable signature with stepping capabilities. In David Naccache, editor, *Quisquater Festschrift*, Lecture Notes in Computer Science. Springer, 2012. Full version available from the web page of the authors. Cited in Sec. 1.1, 5

[BS04] Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 2004: 11th Conference on Computer and Communications Security*, pages 168–177, Washington, DC, USA, October 25–29, 2004. ACM Press. doi:10.1145/1030083.1030106. Cited in Sec. 1.1

[BSZ05] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 136–153, San Francisco, CA, USA, February 14–18, 2005. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-30574-3_11. Cited in Sec. 1, 1.3, 2.2

[BW06] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 427–444, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany. doi:10.1007/11761679_26. Cited in Sec. 1.1

[CG07] Sébastien Canard and Aline Gouget. Divisible e-cash systems can be truly anonymous. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 482–497, Barcelona, Spain, May 20–24, 2007. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-72540-4_28. Cited in Sec. 2.9

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-28628-8_4. Cited in Sec. 1.1

[CM14] Melissa Chase and Sarah Meiklejohn. Déjà Q: Using dual systems to revisit q-type assumptions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 622–639, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-55220-5_34. Cited in Sec. 1.1

[CMM16] Melissa Chase, Mary Maller, and Sarah Meiklejohn. Déjà Q all over again: Tighter and broader reductions of q-type assumptions. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 655–681, Hanoi, Vietnam, December 4–8, 2016. Springer, Heidelberg, Germany. `doi:10.1007/978-3-662-53890-6_22`. Cited in Sec. 1.1

[CS18] Remi Clarisse and Olivier Sanders. Short group signature in the standard model. Cryptology ePrint Archive, Report 2018/1115, 2018. `https://eprint.iacr.org/2018/1115`. Cited in Sec. 1.1

[Cv91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *Advances in Cryptology – EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265, Brighton, UK, April 8–11, 1991. Springer, Heidelberg, Germany. `doi:10.1007/3-540-46416-6_22`. Cited in Sec. 1

[dLS18] Rafaël del Pino, Vadim Lyubashevsky, and Gregor Seiler. Lattice-based group signatures and zero-knowledge proofs of automorphism stability. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 574–591, Toronto, ON, Canada, October 15–19, 2018. ACM Press. `doi:10.1145/3243734.3243852`. Cited in Sec. 1.1

[DP06] Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In Phong Q. Nguyen, editor, *Progress in Cryptology - VIETCRYPT 06: 1st International Conference on Cryptology in Vietnam*, volume 4341 of *Lecture Notes in Computer Science*, pages 193–210, Hanoi, Vietnam, September 25–28, 2006. Springer, Heidelberg, Germany. Cited in Sec. 1.1, 1.2, 2.7, 3.2

[DS18] David Derler and Daniel Slamanig. Highly-efficient fully-anonymous dynamic group signatures. In Jong Kim, Gail-Joon Ahn, Seungjoo Kim, Yongdae Kim, Javier López, and Taesoo Kim, editors, *ASIACCS 18: 13th ACM Symposium on Information, Computer and Communications Security*, pages 551–565, Incheon, Republic of Korea, April 2–6, 2018. ACM Press. Cited in Sec. 1.1

[Duc10] Léo Ducas. Anonymity from asymmetry: New constructions for anonymous HIBE. In Josef Pieprzyk, editor, *Topics in Cryptology – CT-RSA 2010*, volume 5985 of *Lecture Notes in Computer Science*, pages 148–164, San Francisco, CA, USA, March 1–5, 2010. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-11925-5_11`. Cited in Sec. 3.1

[Fis06] Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 60–77, Santa Barbara, CA, USA, August 20–24, 2006. Springer, Heidelberg, Germany. `doi:10.1007/11818175_4`. Cited in Sec. 3.3

[FLM11] Marc Fischlin, Benoît Libert, and Mark Manulis. Non-interactive and re-usable universally composable string commitments with adaptive security. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 468–485, Seoul, South Korea, December 4–8, 2011. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-25385-0_25`. Cited in Sec. 2.9

[FPV09] Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Transferable constant-size fair e-cash. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors, *CANS 09: 8th International Conference on Cryptology and Network Security*, volume 5888 of *Lecture Notes in Computer Science*, pages 226–247, Kanazawa, Japan, December 12–14, 2009. Springer, Heidelberg, Germany. Cited in Sec. 6

[FV10] Georg Fuchsbauer and Damien Vergnaud. Fair blind signatures without random oracles. In Daniel J. Bernstein and Tanja Lange, editors, *AFRICACRYPT 10: 3rd International Conference on Cryptology in Africa*, volume 6055 of *Lecture Notes in Computer Science*, pages 16–33, Stellenbosch, South Africa, May 3–6, 2010. Springer, Heidelberg, Germany. Cited in Sec. 2.9

[GKV10] S. Dov Gordon, Jonathan Katz, and Vinod Vaikuntanathan. A group signature scheme from lattice assumptions. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 395–412, Singapore, December 5–9, 2010. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-17373-8_23`. Cited in Sec. 1.1

[GL19] Lydia Garms and Anja Lehmann. Group signatures with selective linkability. In Dongdai Lin and Kazue Sako, editors, *PKC 2019: 22nd International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 11442 of *Lecture Notes in Computer Science*, pages 190–220, Beijing, China, April 14–17, 2019. Springer, Heidelberg, Germany. `doi:10.1007/978-3-030-17253-4_7`. Cited in Sec. 1.1

[GPS08] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008. Cited in Sec. 2.1

[Gro06] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 444–459, Shanghai, China, December 3–7, 2006. Springer, Heidelberg, Germany. `doi:10.1007/11935230_29`. Cited in Sec. 1.1, 1.2

[Gro07] Jens Groth. Fully anonymous group signatures without random oracles. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 164–180, Kuching, Malaysia, December 2–6, 2007. Springer, Heidelberg, Germany. `doi:10.1007/978-3-540-76900-2_10`. Cited in Sec. 1.1, 1.2, 5

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432, Istanbul, Turkey, April 13–17, 2008. Springer, Heidelberg, Germany. `doi:10.1007/978-3-540-78967-3_24`. Cited in Sec. 1.2

[HJ12] Dennis Hofheinz and Tibor Jager. Tightly secure signatures and public-key encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 590–607, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany. `doi:10.1007/978-3-642-32009-5_35`. Cited in Sec. 3

[Kak10] Saqib A. Kakvi. Eficienct fully anonymous group signatures based on the Groth group signature scheme. Master's thesis, University College London, 2010. . Cited in Sec. 1.1, 5

[KY06] Aggelos Kiayias and Moti Yung. Secure scalable group signature with dynamic joins and separable authorities. *IJSN*, 1(1/2):24–45, 2006. URL: `http://dx.doi.org/10.1504/IJSN.2006.010821`, `doi:10.1504/IJSN.2006.010821`. Cited in Sec. 1

[KY19] Shuichi Katsumata and Shota Yamada. Group signatures without NIZK: From lattices in the standard model. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part III*, volume 11478 of *Lecture Notes in Computer Science*, pages 312–344, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany. `doi:10.1007/978-3-030-17659-4_11`. Cited in Sec. 1.1

[LLM+17] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Adaptive oblivious transfer with access control from lattice assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 533–563, Hong Kong, China, December 3–7, 2017. Springer, Heidelberg, Germany. `doi:10.1007/978-3-319-70694-8_19`. Cited in Sec. 1.1

[LNWX17] San Ling, Khoa Nguyen, Huaxiong Wang, and Yanhong Xu. Lattice-based group signatures: Achieving full dynamicity with ease. In Dieter Gollmann, Atsuko Miyaji, and Hiroaki Kikuchi, editors, *ACNS 17: 15th International Conference on Applied Cryptography and Network Security*, volume 10355 of *Lecture Notes in Computer Science*, pages 293–312, Kanazawa, Japan, July 10–12, 2017. Springer, Heidelberg, Germany. `doi:10.1007/978-3-319-61204-1_15`. Cited in Sec. 1.1

[LNWX18] San Ling, Khoa Nguyen, Huaxiong Wang, and Yanhong Xu. Constant-size group signatures from lattices. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 58–88, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Heidelberg, Germany. `doi:10.1007/978-3-319-76581-5_3`. Cited in Sec. 1.1

[LPY15] Benoît Libert, Thomas Peters, and Moti Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 296–316, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany. `doi:10.1007/978-3-662-48000-7_15`. Cited in Sec. 1.1, 1.2, 5, 5, 3

[LV09] Benoît Libert and Damien Vergnaud. Group signatures with verifier-local revocation and backward unlinkability in the standard model. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors, *CANS 09: 8th International Conference on Cryptology and Network Security*, volume 5888 of *Lecture Notes in Computer Science*, pages 498–517, Kanazawa, Japan, December 12–14, 2009. Springer, Heidelberg, Germany. Cited in Sec. 2.9

[OEH+19] Kazuma Ohara, Keita Emura, Goichiro Hanaoka, Ai Ishida, Kazuo Ohta, and Yusuke Sakai. Shortening the libert-peters-yung revocable group signature scheme by using the random oracle methodology. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 102-A(9):1101–1117, 2019. URL: `http://search.ieice.org/bin/summary.php?id=e102-a_9_1101`. Cited in Sec. 1.1

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg, Germany. `doi:10.1007/11426639_7`. Cited in Sec. 1.2, 2.7, 3.2