Short Group Signature in the Standard Model

Rémi Clarisse[1,2] and Olivier Sanders[1]

[1] Orange Labs, Applied Crypto Group, Cesson-Sévigné, France
[2] Université de Rennes 1, IRMAR, Rennes, France

Abstract. Group signature is a central tool for privacy-preserving protocols, ensuring authentication, anonymity and accountability. It has been massively used in cryptography, either directly or through variants such as direct anonymous attestations. However it remains a complex tool, especially in the standard model where each of its building blocks is quite costly to instantiate.
In this work, we propose a new group signature scheme proven secure in the standard model which significantly decreases the complexity with respect to the state-of-the-art. More specifically, we halve both the size and the computational cost compared to the most efficient alternative in the standard model. Moreover, our construction is also competitive against the most efficient ones in the random oracle model, thus closing the traditional efficiency gap between these two models.
Our construction is based on a tailored combination of two popular signatures, which avoids the explicit use of encryption schemes or zero-knowledge proofs. It is flexible enough to achieve security in different models and is thus suitable for most contexts.

1 Introduction

Group Signature, introduced by Chaum and van Heyst [19], enables members of a group to sign on behalf of the group. The point is that the signature is anonymous, *i.e.* it cannot be traced back to its issuer, except for a specific entity, the opening authority, which can "open" any valid group signature.

The fact that this primitive combines authentication, anonymity and accountability has proved very useful in several contexts such as public transport [23] or attestation for secure devices [12, 20]. But more than that, group signature has become the cornerstone of numerous privacy-preserving protocols. Indeed, primitives like direct anonymous attestation [20] or anonymous credentials [13] can be seen as variants of group signatures. Even more intricate schemes, such as divisible e-cash systems [15,36], are implicitly constructed upon group signatures. Such a central position naturally enhances any progress on this primitive, which has drawn attention from many cryptographers.

1.1 Related Works

Combining seemingly contradictory properties such as anonymity and authentication has proved tricky, the first really practical solution being provided by

Ateniese *et al* [2]. Few years later, Bellare, Micciancio and Warinschi [5] proposed the first security model for static group signature, which was later extended to the case of dynamic group signature [6]. Besides providing a way to assess existing schemes, these seminal works have introduced a generic construction that has become the implicit framework for most of the following group signatures.

Informally, a group member of this generic construction receives a certificate (a digital signature) τ on his public key upk when he joins the group. To compute a group signature on some message m, he first generates a digital signature σ on m (using the corresponding signing key usk) and then encrypts σ and τ. Finally, he provides a non-interactive zero-knowledge proof (NIZK) that every element is well formed. This three-steps approach is usually known as Sign-Encrypt-Prove (SEP) in the literature.

The strength of the SEP paradigm is that it is based on standard cryptographic primitives for which many instantiations exist. Unfortunately, it leads to quite complex constructions because of the security requirements placed on each building block, but primarily because of the complexity of the resulting NIZK proof. Indeed, the signer must prove, without revealing σ and τ, that the group signature is a valid encryption of the signature σ that has been generated using keys certified by the group manager.

Such a statement is difficult to prove, even in the random oracle model (ROM), and this becomes worse if one wants to achieve security in the standard model. Indeed, NIZK proofs are much more complex in this setting and even by using the Groth-Sahai methodology [29], one still gets group signatures containing dozens of elements (see *e.g.* [28]).

A natural question arising from this observation is whether it is possible to construct more efficient schemes by using a different paradigm. In [7], Bichsel *et al* proposed an interesting answer to this question. They indeed introduced a very efficient alternative, at the cost of a slightly weaker notion of anonymity. The latter allows them to circumvent the result of Abdalla and Warinschi [1] and thus to avoid encryption. More specifically, their idea was to remove encryption by using re-randomizable [14] certificates τ and by merging σ with the NIZK proofs, leading to a signature of knowledge. The resulting construction is very efficient (see Figure 2 at the end of the paper) and can be further improved by instantiating it with the randomizable signature scheme of Pointcheval and Sanders (PS) [34].

Another alternative based on equivalence-class signature [31] has recently been proposed by Derler and Slamanig [22]. It shares commonalities with [7], such as the absence of explicit encryption, but manages to achieve full anonymity at the cost of increased complexity. Unfortunately, both [7] and [22] inherently rely on signature of knowledge, usually by combining Schnorr's proofs [37] with the Fiat-Shamir heuristic [24], and so rather fit the random oracle model.

Very recently, Backes *et al* [3] proposed a different framework based on a new primitive called signatures with flexible public keys. It yields secure constructions in the standard model with improved efficiency compared to the state-of-the-art in this setting. However, the resulting group signatures are still three times

larger than the most efficient one in the ROM (see Figure 2) and require more computations to be generated.

More generally, designers of group signature schemes are confronted with the choice of either proving security in the standard model or favouring efficiency by relying on the random oracle model whose limits are known [16, 17].

1.2 Our Contribution

In this work we aim to close the gap between the efficiency of constructions in the ROM and the one of constructions in the standard model. We indeed propose a new group signature scheme in the standard model that halves the size and the computational complexity compared to the state-of-the-art [3]. More specifically our group signature only consists of 1536 bits which makes it very competitive, even against constructions in the ROM (see Section 5 for more details).

As [3, 7, 22], our construction departs from the SEP framework and heavily relies on the randomizability of its components. However, contrarily to [3, 7, 22] that assemble different building blocks (digital signature, NIZKs, etc) and so achieve some level of genericity, we are here interested in optimizing the combination so as to get the best possible efficiency.

Our work results from the observation that the equivalence-class signature of Fuchsbauer, Hanser and Slamanig (FHS) [25] nicely interacts with the PS signature scheme [34]. More specifically, assuming very slight modifications of the FHS public key and of the PS signatures, we are able to merge the verification equations of FHS signatures with the one of PS signatures. Such a merge is crucial for our construction: it indeed means that it is no longer necessary to provide a NIZK proof that the signatures are valid and related. One can indeed verify our group signatures by essentially running the verification algorithm of FHS signatures.

Intuitively, we modify the PS signature scheme in such a way that each signature is of the form $(g^r, g^{y \cdot r} X^{r/h_m})$ where g^y is the user's secret key, r is a random scalar, h_m is a public element that depends on the message m to be signed and X is a public element common to all members of the group. One can then note that each signature contains a different representative of the same equivalence class as (g, g^y) and so it is quite easy, given a FHS signature on this pair, to prove that the PS signature was generated using certified keys. It only remains to remove the term in X in the PS signature but this can easily be done by adding the necessary elements in the group public key.

Our group signature thus only consists of a PS signature and a FHS one which are both re-randomizable, leading to an anonymity proof under the DDH assumption. Moreover, we can prove that a non-registered user cannot generate a valid group signature unless he is able to forge FHS signatures. We only pay the price for our tailored construction in the proof of non-frameability, where we want to prove that no one can issue a forged group signature that can be traced back to an honest user. Indeed, we would like to directly rely on the security of PS signatures but this is impossible due to the modifications we introduced.

However, we show that we can tweak the original proof of PS signatures to suit our construction and so that we can rely on similar arguments for this property.

While being non-generic, our construction remains flexible enough to achieve security in different models. Interestingly, the different variants we consider in our paper achieve the same efficiency with respect to the group signature but mostly differ in the definition of the registration procedure. This concretely means that one can select the most suitable setting according to his context without any impact on the group signature itself. This also allows us in Section 5 to fairly compare our construction with the most relevant ones of the sate-of-the-art and so to highlight the benefits of our group signature in all cases.

1.3 Organisation

We describe in Section 2 the building blocks that we need to construct our group signature. Section 3 recalls the standard security model of group signatures. We describe our construction in Section 4 and compare it with the most relevant alternatives of the state-of-the-art in Section 5. The security proofs are provided in Appendix A.

2 Preliminaries

2.1 Bilinear Groups

Our construction requires bilinear groups whose definition is recalled below.

Definition 1. *Bilinear groups are a set of three groups* $\mathbb{G}_1$, $\mathbb{G}_2$, *and* $\mathbb{G}_T$ *of order* p *along with a map, called pairing,* $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *that is*

1. *bilinear: for any* $g \in \mathbb{G}_1, \widetilde{g} \in \mathbb{G}_2$, *and* $a, b \in \mathbb{Z}_p$, $e(g^a, \widetilde{g}^b) = e(g, \widetilde{g})^{ab}$;
2. *non-degenerate: for any* $g \in \mathbb{G}_1^*$ *and* $\widetilde{g} \in \mathbb{G}_2^*$, $e(g, \widetilde{g}) \neq 1_{\mathbb{G}_T}$;
3. *efficient: for any* $g \in \mathbb{G}_1$ *and* $\widetilde{g} \in \mathbb{G}_2$, $e(g, \widetilde{g})$ *can be efficiently computed.*

In this work we will only consider bilinear groups of prime order with *type 3* pairings [26], meaning that no efficiently computable homomorphism is known between $\mathbb{G}_1$ and $\mathbb{G}_2$. We stress that this is not a significant restriction since this yields the most efficient parameters [18,30]. To highlight the differences between $\mathbb{G}_1$ and $\mathbb{G}_2$, we will always denote elements of the latter with a $\widetilde{}$ (*e.g.* $\widetilde{g}$).

2.2 Signature

Our construction will use a digital signature scheme as a building block, along with a specific instantiation provided by Pointcheval and Sanders [34]. We will additionally require a variant of digital signature, called equivalence-class signature [25, 31], where, given a signature on a message m, it is possible to derive new signatures for messages belonging to the same equivalence class as m.

Digital Signature. A digital signature scheme Σ is defined by four algorithms:

- $\mathtt{Setup}(1^\lambda)$: on input a security parameter λ, this algorithm outputs the public parameters pp of the system;
- $\mathtt{Keygen}(pp)$: on input pp, this algorithm outputs a pair of signing and verification keys $(\mathsf{sk}, \mathsf{pk})$;
- $\mathtt{Sign}(\mathsf{sk}, m)$: on input the signing key sk and a message m, this algorithm outputs a signature σ;
- $\mathtt{Verify}(\mathsf{pk}, m, \sigma)$: on input the verification key pk, a message m and its alleged signature σ, this algorithm outputs 1 if σ is a valid signature on m under pk, and 0 otherwise.

The standard security notion for a signature scheme is *existential unforgeability under chosen message attacks* (EUF-CMA) [27]: it means that it is hard, even given access to a signing oracle, to output a valid pair (m, σ) for a message m never asked to the signing oracle.

PS Signature. In [34, 35], Pointcheval and Sanders propose a *randomizable* signature scheme, *i.e.* a scheme enabling to derive re-randomized versions σ' of any valid signature σ. An interesting feature of their signatures is that one cannot link σ and σ' without knowing the corresponding message. Actually, the original paper describe several versions of their signature scheme, offering different features. In this work we will use their variant supporting aggregation, although we will not use this specific feature, because it enables to decrease the size of the public key.

- $\mathtt{Setup}(1^\lambda)$: this algorithm outputs the parameters pp containing the description of type 3 bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ along with a set of generators $(g, \widetilde{g}) \in \mathbb{G}_1 \times \mathbb{G}_2$ and a pair $(X, \widetilde{X}) \leftarrow (g^x, \widetilde{g}^x)$ for some random scalar x;
- $\mathtt{Keygen}(pp)$: this algorithm generates a random scalar y and sets $(\mathsf{sk}, \mathsf{pk})$ as $(g^y, \widetilde{Y} = \widetilde{g}^y)$;
- $\mathtt{Sign}(\mathsf{sk}, m)$: a signature on a message m is generated by computing $(\sigma_1, \sigma_2) \leftarrow (g^r, X^r \cdot g^{r \cdot y \cdot m})$ for some random scalar r;
- $\mathtt{Verify}(\mathsf{pk}, m, (\sigma_1, \sigma_2))$: a signature (σ_1, σ_2) is valid on m if the following equation holds:
$$e(\sigma_1, \widetilde{X} \cdot \widetilde{Y}^m) = e(\sigma_2, \widetilde{g}).$$

One can note that anyone can re-randomize a signature by raising σ_1 and σ_2 to the same power t.

FHS Signature. In [25], Fuchsbauer, Hanser and Slamanig introduce an equivalence-class signature [31] for the following equivalence relation on tuples $(M_1, \ldots, M_n) \in \mathbb{G}_1^n$:

$$(M_1, \ldots, M_n) \sim (N_1, \ldots, N_n) \text{ if } \exists a \in \mathbb{Z}_p \text{ such that } N_i = M_i^a \ \forall i \in [1, n].$$

In this paper, we will only consider the case $n = 2$ so we directly define their signature scheme in this setting.

– $\mathtt{Setup}(1^\lambda)$: this algorithm outputs the parameters pp containing the description of type 3 bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ along with a set of generators $(g, \widetilde{g}) \in \mathbb{G}_1 \times \mathbb{G}_2$;
– $\mathtt{Keygen}(pp)$: this algorithm generates two random scalars α_1 and α_2 and sets sk as (α_1, α_2) and pk as $(\widetilde{A}_1, \widetilde{A}_2) = (\widetilde{g}^{\alpha_1}, \widetilde{g}^{\alpha_2})$;
– $\mathtt{Sign}(\mathsf{sk}, m)$: To sign a representative $(M_1, M_2) \in \mathbb{G}_1^2$ of some equivalence class, one selects a random $t \in \mathbb{Z}_p$ and computes $(\tau_1, \tau_2) \leftarrow ((M_1^{\alpha_1} M_2^{\alpha_2})^t, g^{1/t})$ along with $\widetilde{\tau} \leftarrow \widetilde{g}^{1/t}$;
– $\mathtt{Verify}(\mathsf{pk}, m, (\tau_1, \tau_2, \widetilde{\tau}))$: a signature $(\tau_1, \tau_2, \widetilde{\tau}) \in \mathbb{G}_1^2 \times \mathbb{G}_2$ is valid on the representative (M_1, M_2) if the following equations hold:
  • $e(\tau_1, \widetilde{\tau}) = e(M_1, \widetilde{A}_1) \cdot e(M_2, \widetilde{A}_2)$
  • $e(\tau_2, \widetilde{g}) = e(g, \widetilde{\tau})$

  We note that the signature $(\tau_1, \tau_2, \widetilde{\tau})$ is only valid on the representative (M_1, M_2). However, we can easily derive a signature on other representatives (M_1^r, M_2^r) of the same equivalence class, while re-randomizing the signature, by generating a random t' and computing $(\tau_1^{r \cdot t'}, \tau_2^{1/t'}, \widetilde{\tau}^{1/t'})$.

2.3 Public Key Encryption

Our construction will marginally use an encryption scheme Γ which is defined by the following algorithms.

– $\mathtt{Keygen}(1^\lambda)$: this algorithm outputs a pair of decryption and encryption keys $(\mathsf{sk}, \mathsf{pk})$;
– $\mathtt{Encrypt}(\mathsf{pk}, m)$: on input pk and a message m, this algorithm outputs a ciphertext c;
– $\mathtt{Decrypt}(\mathsf{sk}, c)$: on input the decryption key sk and a ciphertext c, this algorithm outputs a message m or $\perp$.

Several security notions exist for public key encryption but our group signature scheme will only need indistinguishability under adaptive chosen ciphertext attack (IND-CCA2). Informally it requires that no adversary, even given access to a decryption oracle, is able to distinguish an encryption of m_0 from an encryption of m_1, where m_0 and m_1 are two messages chosen by the adversary.

2.4 Computational Assumption

SXDH assumption. For $k \in \{1, 2\}$, the DDH assumption is hard in $\mathbb{G}_k$ if, given $(g, g^x, g^y, g^z) \in \mathbb{G}_k^4$, it is hard to distinguish whether $z = x \cdot y$ or z is random. The SXDH assumption holds if DDH is hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

2.5 Groth-Sahai Proof System

In [29], Groth and Sahai proposed a non-interactive proof system, in the common reference string (CRS) model, which captures most of the relations for bilinear groups. There are two types of setup for the CRS that yield either perfect

soundness or perfect witness indistinguishability, while being computationally indistinguishable (under the SXDH assumption, in our setting).

To prove that some variables satisfy a set of relations, the prover first commits to them (by using the elements from the CRS) and then computes one proof element per relation. Efficient non-interactive witness indistinguishable proofs are available for

- pairing-product equations, for variables $\{X_i\}_{i=1}^n \in \mathbb{G}_1$, $\{\widetilde{X}_i\}_{i=1}^n \in \mathbb{G}_2$ and constants $t_T \in \mathbb{G}_T$, $\{A_i\}_{i=1}^n \in \mathbb{G}_1$, $\{\widetilde{B}_i\}_{i=1}^n \in \mathbb{G}_2$, $\{a_{i,j}\}_{i,j=1}^n \in \mathbb{Z}_p$:

$$\prod_{i=1}^{n} e(A_i, \widetilde{X}_i) \prod_{i=1}^{n} e(X_i, \widetilde{B}_i) \prod_{i=1}^{n}\prod_{j=1}^{n} e(X_i, \widetilde{X}_j)^{a_{i,j}} = t_T;$$

- or multi-exponentiation equations, for variables $\{X_i\}_{i=1}^n \in \mathbb{G}_k$, $\{y_i\}_{i=1}^n \in \mathbb{Z}_p$ and constants $T \in \mathbb{G}_k$, $\{A_i\}_{i=1}^n \in \mathbb{G}_k$, $\{b_i\}_{i=1}^n \in \mathbb{Z}_p$, $\{a_{i,j}\}_{i,j=1}^n \in \mathbb{Z}_p$ for $k \in \{1, 2\}$:

$$\prod_{i=1}^{n} A_i^{y_i} \prod_{j=1}^{n} X_j^{b_j} \prod_{i=1}^{n}\prod_{j=1}^{n} X_j^{y_i \cdot a_{i,j}} = T.$$

The Groth-Sahai framework also supports non-interactive zero-knowledge (NIZK) proofs for any multi-exponentiation equation or for pairing-product equations such that $t_T = 1_{\mathbb{G}_T}$ or $t_T = \prod(A_i, \widetilde{B}_i)$.

3 Group Signature

For completeness, we recall in this section the security model for dynamic group signature from [6]. We only introduce some minor syntactic changes and discuss popular variants of the original security notions introduced by Bellare *et al.* A reader familiar with group signature can then safely jump to the description of such variants, in remark 2 at the end of this section.

3.1 Syntax

A group signature scheme is defined by the following algorithms that involve three types of entities: a group manager, an opening authority and users. Each of the latter is identified by a public index $i \in \mathbb{N}^*$.

- $\mathtt{Setup}(1^\lambda)$: On input a security parameter λ, this algorithm returns the public parameters pp of the system.
- $\mathtt{UKeygen}(pp)$: On input the public parameter pp, this algorithm returns a user's key pair $(\mathsf{sk}, \mathsf{pk})$. We assume that pk is public and that anyone can get an authentic copy of it for each user.
- $\mathtt{OKeygen}(pp)$: On input the public parameters pp, this algorithm returns the opening authority's key pair $(\mathsf{osk}, \mathsf{opk})$.

– $\texttt{GKeygen}(pp)$: On input the public parameters pp, this algorithm returns the group manager's key pair $(\mathsf{gsk}, \mathsf{gpk})$ along with a public register $\mathbf{Reg}$.
– $\texttt{Join}$: this is a two party interactive protocol between the group manager and a user i who wants to join the group. The input of the former is $(\mathsf{gsk}, \mathbf{Reg}, \mathsf{opk}, \mathsf{pk}_i)$ whereas the user takes as input $(\mathsf{gpk}, \mathsf{opk}, \mathsf{sk}_i)$. If the protocol does not fail, then the user gets a group signing key usk_i whereas the group manager updates $\mathbf{Reg}$. Else, both parties return $\perp$.
– $\texttt{Sign}(\mathsf{usk}_i, m)$: On input a group signing key usk_i and a message m, this algorithm returns a group signature σ on m.
– $\texttt{Verify}(\mathsf{gpk}, \sigma, m)$: On input the group manager's public key, a group signature σ and a message m, this algorithm returns a bit $b \in \{0, 1\}$.
– $\texttt{Open}(\mathsf{osk}, \mathsf{gpk}, \mathbf{Reg}, \sigma, m)$: On input the opening authority's secret key, the group manager's public key, the register $\mathbf{Reg}$, a group signature σ and a message m, this algorithm returns either 0, $\perp$ or an index $i \in \mathbb{N}^*$ along with a proof π.
– $\texttt{Judge}(\mathsf{gpk}, \mathbf{Reg}, \sigma, m, i, \pi)$: On input the group manager's public key, the register $\mathbf{Reg}$, a group signature σ, a message m, an index $i \in \mathbb{N}^*$ and a proof π, this algorithm returns a bit $b \in \{0, 1\}$.

3.2 Security Model

A group signature should achieve *correctness*, *anonymity*, *traceability* and *non-frameability*. We refer to [6] for a formal definition of correctness, but informally it means that any user who has joined the group should be able to produce valid signatures σ (*i.e.* one for which $\texttt{Verify}$ outputs 1) on any message m. Moreover, it should be possible to open such signatures, *i.e.* to recover the identity i of the signer, and to produce publicly verifiable proofs that i has indeed issued these signatures.

Anonymity, traceability and non-frameability are defined using the experiments of Figure 1. The first property requires that group signatures should be anonymous, except for the opening authority. Traceability requires that no one can produced a valid signature that cannot be traced back to some user through the $\texttt{Open}$ procedure. Finally, non-frameability means that no one can be falsely accused of having produced a signature. The corresponding experiments make use of the following oracles:

– $\mathcal{O}\texttt{Add}(i)$ is an oracle that can be used to add a new user i. It then runs $\texttt{UKeygen}(pp)$ to get $(\mathsf{sk}_i, \mathsf{pk}_i)$ and returns pk_i. If i has already been used in a previous query, then it returns $\perp$.
– $\mathcal{O}\texttt{Join}_U(i)$ is an oracle that plays the user's side of the $\texttt{Join}$ protocol. It can be used by an adversary $\mathcal{A}$ playing the role of a corrupt group manager. It returns $\perp$ if i has already joined the group or if user i does not exist.
– $\mathcal{O}\texttt{Corrupt}(i)$ is an oracle that returns all the secret keys of the user i. The user i is then said to be *corrupt*. Any non-corrupt user is considered as being *honest*.

$\mathtt{Exp}^{anon}_{\mathcal{A}}(1^\lambda)$ – Anonymity Security Game

1. $pp \leftarrow \mathtt{Setup}(1^\lambda)$
2. $(\mathsf{osk}, \mathsf{opk}) \leftarrow \mathtt{OKeygen}(pp)$
3. $(\mathsf{gsk}, \mathsf{gpk}) \leftarrow \mathtt{GKeygen}(pp)$
4. $b \xleftarrow{\$} \{0,1\}$
5. $b^* \leftarrow \mathcal{A}^{\mathcal{O}\mathtt{Add},\mathcal{O}\mathtt{Join}_U,\mathcal{O}\mathtt{Corrupt},\mathcal{O}\mathtt{Sign},\mathcal{O}\mathtt{Open},\mathcal{O}\mathtt{Ch}_b}(\mathsf{gsk}, \mathsf{opk})$
6. If $\mathcal{O}\mathtt{Open}$ is queried on the output of $\mathcal{O}\mathtt{Ch}_b$, then return 0
7. Return $(b = b^*)$

$\mathtt{Exp}^{tra}_{\mathcal{A}}(1^\lambda)$ – Traceability Security Game

1. $pp \leftarrow \mathtt{Setup}(1^\lambda)$
2. $(\mathsf{osk}, \mathsf{opk}) \leftarrow \mathtt{OKeygen}(pp)$
3. $(\mathsf{gsk}, \mathsf{gpk}) \leftarrow \mathtt{GKeygen}(pp)$
4. $(\sigma, m) \leftarrow \mathcal{A}^{\mathcal{O}\mathtt{Add},\mathcal{O}\mathtt{Join}_{GM},\mathcal{O}\mathtt{Corrupt},\mathcal{O}\mathtt{Sign},\mathcal{O}\mathtt{Open},}(\mathsf{gpk}, \mathsf{osk})$
5. If $\perp \leftarrow \mathtt{Open}(\mathsf{osk}, \mathsf{gpk}, \mathbf{Reg}, \sigma, m)$, then return 1
6. If $(i, \pi) \leftarrow \mathtt{Open}(\mathsf{osk}, \mathsf{gpk}, \mathbf{Reg}, \sigma, m)$ and $0 \leftarrow \mathtt{Judge}(\mathsf{gpk}, \mathbf{Reg}, \sigma, m, i, \pi)$, then return 1
7. Return 0

$\mathtt{Exp}^{nf}_{\mathcal{A}}(1^\lambda)$ – Non-Frameability Security Game

1. $pp \leftarrow \mathtt{Setup}(1^\lambda)$
2. $(\mathsf{osk}, \mathsf{opk}) \leftarrow \mathtt{OKeygen}(pp)$
3. $(\mathsf{gsk}, \mathsf{gpk}) \leftarrow \mathtt{GKeygen}(pp)$
4. $(\sigma, m, i, \pi) \leftarrow \mathcal{A}^{\mathcal{O}\mathtt{Add},\mathcal{O}\mathtt{Join}_U,\mathcal{O}\mathtt{Corrupt},\mathcal{O}\mathtt{Sign},\mathcal{O}\mathtt{Open},}(\mathsf{gsk}, \mathsf{osk})$
5. If σ has been returned by $\mathcal{O}\mathtt{Sign}$, then return 0
6. If i is corrupt, then return 0
7. Return $\mathtt{Judge}(\mathsf{gpk}, \mathbf{Reg}, \sigma, m, i, \pi)$

Fig. 1. Security Games for Group Signature

- $\mathcal{O}\mathtt{Join}_{GM}()$ is the counterpart of the $\mathcal{O}\mathtt{Join}_U$ oracle that can be used by a corrupt user to join the group.
- $\mathcal{O}\mathtt{Sign}(i, m)$ is an oracle that returns $\mathtt{Sign}(\mathsf{usk}_i, m)$, provided that i is an honest user that has already joined the group.
- $\mathcal{O}\mathtt{Open}(\sigma, m)$ is an oracle that returns $\mathtt{Open}(\mathsf{osk}, \mathsf{gpk}, \mathbf{Reg}, \sigma, m)$.
- $\mathcal{O}\mathtt{Ch}_b(i_0, i_1, m)$ is an oracle that takes as inputs the indices of two honest users and that returns $\mathtt{Sign}(\mathsf{usk}_{i_b}, m)$.

Let $\mathcal{A}$ be a probabilistic polynomial adversary. A group signature scheme is

- anonymous if $\mathtt{Adv}^{anon}(\mathcal{A}) = |\Pr[\mathtt{Exp}^{anon}_{\mathcal{A}}(1^\lambda) = 1] - 1/2|$ is negligible for any $\mathcal{A}$;
- traceable if $\mathtt{Adv}^{tra}(\mathcal{A}) = \Pr[\mathtt{Exp}^{tra}_{\mathcal{A}}(1^\lambda) = 1]$ is negligible for any $\mathcal{A}$;
- non-frameable if $\mathtt{Adv}^{nf}(\mathcal{A}) = \Pr[\mathtt{Exp}^{nf}_{\mathcal{A}}(1^\lambda) = 1]$ is negligible for any $\mathcal{A}$.

The BSZ model [6] places no restriction on the $\mathcal{O}\mathtt{Corrupt}$ queries in the anonymity experiment. This means that the adversary is allowed to corrupt the "challenge" users (*i.e.* those that are involved in $\mathcal{O}\mathtt{Ch}$ queries). This corresponds

to the strongest notion of anonymity, sometimes called *full anonymity* or *CCA-2 anonymity* (see *e.g.* [11,22]), where anonymity holds even if the users' secret keys leak.

Remark 2. The security model introduced by Bellare, Shi and Zhang (BSZ) [6] defines strong security properties that are sufficient in most contexts. However, it may be possible in some situations to relax some of them, usually leading to more efficient constructions. This is particularly true for the anonymity property for which popular variants exist, such as CPA anonymity [8,11] or selfless anonymity [7,9,34]. The former removes the oracle $\mathcal{O}\mathtt{Open}$ in the anonymity game but retains this property even if all users' secret keys leak. Contrarily, selfless anonymity allows $\mathcal{O}\mathtt{Open}$ queries but the users are no longer anonymous when their secret keys leak. These two notions are incomparable and so fit different contexts. The construction we describe in the next section achieves both of them. Interestingly, it also achieves full anonymity in the model introduced by Bellare, Micciancio and Warinschi [5] where the opening authority is also the group manager.

4 Our Construction

4.1 Intuition

A group signature usually contains two kinds of digital signatures that we will denote by σ and τ. The first one is issued on the message to be signed by the user using his own key pair $(\mathsf{usk}, \mathsf{upk})$. Intuitively, the unforgeability of the digital signature ensures that no adversary is able to produce a forged group signature which can be traced back to upk (non-frameability). The second one is issued by the group manager on usk (or upk) to differentiate key pairs of group members from those of unregistered users. Here, unforgeability ensures that only users that have joined the group can issue group signature, which is necessary to achieve traceability.

If non-frameability and traceability were the only two conditions expected from a group signature, then the latter would simply be $(\tau, \sigma, \mathsf{upk}, m)$. However, this cannot work when anonymity is also required so the standard practice has been to encrypt/commit at least τ and upk and then provide zero-knowledge proofs that these elements are well-formed.

The work of Bichsel et al [7] has shown that we can do better when τ is randomizable. Indeed, in such a case there is no need to encrypt τ, the latter can simply be re-randomized and sent unencrypted, leading to significant gains in efficiency. Their group signature can only achieve a weaker selfless anonymity notion in the ROM, but it seems a reasonable price to pay in view of the benefits.

Despite its novelty, [7] still shares commonalities with the standard framework of [6]. There is indeed still a modular composition of two signatures τ and σ with a proof of knowledge. The latter two can be merged (leading to a signature of knowledge) using the Fiat-Shamir heuristic [24] in the ROM, but the spirit remains the same. Modular systems are interesting since they can leverage any advance in the construction of their building blocks. For example, the scheme

of [7] can straightforwardly be improved by using PS signature [34] to instantiate τ, instead of CL signature [14] in the original construction. Unfortunately, the complexity of a modular construction is the sum of all its parts, so a natural question is whether it is possible to improve efficiency by optimizing the combination of the different building blocks for some specific instantiations.

In this section we construct the most efficient group signature in the standard model by noticing that the equivalence-class signature of Fuchsbauer *et al* [25] nicely interacts with the PS signature scheme [34]. Indeed let us recall the latter, and more specifically its variant designed to support aggregation. A (non-aggregated) signature on a message m in this case is given by $(\sigma_1, \sigma_2) = (g^r, X^r(g^{y \cdot m})^r)$ where r is some random scalar, $X = g^x$ is a public element and y is the signer's secret key. One can note that we can alternatively define σ_2 as $\sigma_2^{1/m} = X^{r/m}(g^y)^r$: any adversary able to forge such a signature can trivially be converted into an adversary against the original PS signature scheme.

Therefore, any signature issued by this user will be of the form $(\sigma_1, \sigma_2) = (g^r, X^{r/m}(g^y)^r)$. If we applied the standard methodology here, we would provide a signature τ on y (or g^y) and then prove in a zero-knowledge way that τ is valid on the key that has been used to generate (σ_1, σ_2). However, we can do better if we directly use the FHS signature scheme [25].

Indeed, for all r, if we discard the term $X^{r/m}$ in σ_2, it only remains $(g^r, g^{y \cdot r})$ which are different representatives of the same equivalent class. Thus, if we provide a FHS signature on $(g^r, g^{r \cdot y})$ one can *directly* check that (σ_1, σ_2) was generated using a certified key, without any proof of knowledge. Anonymity of the resulting construction simply follows from the ability to re-randomize FHS signature while changing the representative of the class.

It then only remains to explain how to remove $X^{r/m}$. Recall that a FHS signature on $(g^r, g^{r \cdot y})$ is a tuple $(\tau_1, \tau_2, \widetilde{\tau})$ such that:

- $e(\tau_1, \widetilde{\tau}) = e(g^r, \widetilde{A}_1) \cdot e(g^{r \cdot y}, \widetilde{A}_2)$
- $e(\tau_2, \widetilde{g}) = e(g, \widetilde{\tau})$

where $(\widetilde{A}_1, \widetilde{A}_2) = (\widetilde{g}^{\alpha_1}, \widetilde{g}^{\alpha_2})$ is the public key. Let us assume that we add $\widetilde{B} = \widetilde{X}^{\alpha_2}$ to this public key ($\widetilde{X} = \widetilde{g}^x$ is a part of the public key of the PS signature scheme). Then,

$$\begin{aligned} e(\sigma_1, \widetilde{A}_1) \cdot e(\sigma_2, \widetilde{A}_2) \cdot e(\sigma_1, \widetilde{B}^{-1/m}) &= e(g^r, \widetilde{A}_1) \cdot e(g^{y \cdot r}, \widetilde{A}_2) \\ &= e(\tau_1, \widetilde{\tau}), \end{aligned}$$

and the second equation remains unchanged. This means that we can check the validity of both the FHS and PS signatures at essentially the cost of verifying a FHS signature. Moreover, the fact that we merge the verification of these signatures makes zero-knowledge proofs unnecessary. Concretely, this means that our group signature only consists of $(\sigma_1, \sigma_2, \tau_1, \tau_2, \widetilde{\tau})$, *i.e.* 4 elements of $\mathbb{G}_1$ and 1 element of $\mathbb{G}_2$, and can be verified with only two pairing equations.

Interestingly, the fact that we avoid the classical signature of knowledge of y allows to achieve both CPA anonymity and selfless anonymity. Indeed, schemes based on randomizable signatures (see *e.g.* [14,34]) are usually proven anonymous under the DDH assumption in $\mathbb{G}_1$. Therefore, to enable opening, they usually force the users to provide some "trapdoor" $\widetilde{g}^y \in \mathbb{G}_2$ that allows the opening authority to break DDH on their specific signatures. When y is part of the user's signing key usk (which is necessary for a signature of knowledge of y), leakage of the latter means that the adversary can recover y and thus $\widetilde{g}^y$. Anonymity can then no longer hold in this case leading to the selfless anonymity notion.

In our case, we note that $g^y \in \mathbb{G}_1$ is enough to issue group signature, meaning that the user can discard y after generating his key. In case usk leaks, the adversary now recovers g^y, which is useless to break DDH. We can thus retain some level of anonymity (at least CPA anonymity) in this case.

4.2 The Protocol

- $\mathtt{Setup}(1^\lambda)$: Let $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ be the description of type 3 bilinear groups of prime order p, this algorithm first selects $g \xleftarrow{\$} \mathbb{G}_1^*$ and $\widetilde{g} \xleftarrow{\$} \mathbb{G}_2^*$, and then computes $(X, \widetilde{X}) \leftarrow (g^x, \widetilde{g}^x)$ for some random scalar x. It also generates the public parameters pp_Σ for a digital signature scheme Σ and selects a hash function $h : \{0,1\}^* \to \mathbb{Z}_p$. Finally, it generates a common reference string crs for the Groth-Sahai proof system [29] in the SXDH setting and then sets the public parameters as $pp = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \widetilde{g}, X, \widetilde{X}, crs, pp_\Sigma, h)$.
- $\mathtt{UKeygen}(pp)$: The user defines his own key pair as $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathtt{Keygen}(pp_\Sigma)$.
- $\mathtt{OKeygen}(pp)$: The opening authority generates a key pair for a public key encryption scheme Γ and defines osk as the decryption key and opk as the encryption key.
- $\mathtt{GKeygen}(pp)$: The group manager selects two random scalar α_1 and α_2 and then computes $(\widetilde{A}_1, \widetilde{A}_2, \widetilde{B}) \leftarrow (\widetilde{g}^{\alpha_1}, \widetilde{g}^{\alpha_2}, \widetilde{X}^{\alpha_2})$. He then initializes a public register $\mathbf{Reg}$ and returns $(\mathsf{gsk}, \mathsf{gpk}) \leftarrow ((\alpha_1, \alpha_2), (\widetilde{A}_1, \widetilde{A}_2, \widetilde{B}, \mathbf{Reg}))$.
- $\mathtt{Join}$: To join the group, a user i first selects two random scalars, u and y, and computes $(g^u, g^{u \cdot y})$ along with $C \leftarrow \Gamma.\mathtt{Encrypt}(\mathsf{opk}, \widetilde{g}^y)$. He then generates a NIZK proof π that C encrypts an element $\widetilde{g}^y \in \mathbb{G}_2$ such that

$$e(g^u, \widetilde{g}^y) = e(g^{u \cdot y}, \widetilde{g})$$

  Finally, he generates $\mu \leftarrow \Sigma.\mathtt{Sign}(\mathsf{sk}_i, (g^u || g^{u \cdot y} || C || \pi))$ and sends it, along with $(g^u, g^{u \cdot y}, C, \pi)$, to the group manager.
  Upon receiving these elements, the group manager checks the validity of the proof π and that $\Sigma.\mathtt{Verify}(\mathsf{pk}_i, \mu, (g^u || g^{u \cdot y} || C || \pi)) = 1$. If π and μ both pass the tests, then the group manager stores $(g^u, g^{u \cdot y}, C, \pi, \mathsf{pk}_i, \mu)$ in $\mathbf{Reg}[i]$, generates a random $t \xleftarrow{\$} \mathbb{Z}_p$ and returns $\tau_1' \leftarrow ((g^u)^{\alpha_1}(g^{u \cdot y})^{\alpha_2})^t$, $\tau_2 \leftarrow g^{1/t}$ and $\widetilde{\tau} \leftarrow \widetilde{g}^{1/t}$.
  The user then computes $\tau_1 \leftarrow (\tau_1')^{1/u}$ and sets $\mathsf{usk}_i = (\tau_1, \tau_2, \widetilde{\tau}, g^y)$.
- $\mathtt{Sign}(\mathsf{usk}_i, m)$: To sign a message m, the user first selects two random scalars r and s, and generates the following elements:

- $\tau_1' \leftarrow \tau_1^{r \cdot s}$
- $(\tau_2', \widetilde{\tau}') \leftarrow (\tau_2^{1/s}, \widetilde{\tau}^{1/s})$
- $(\sigma_1, \sigma_2) \leftarrow (g^r, X^{r/h(\tau_1'||\tau_2'||\widetilde{\tau}'||\sigma_1||m)} \cdot (g^y)^r)$

The group signature σ on m is then defined as $\sigma = (\tau_1', \tau_2', \widetilde{\tau}', \sigma_1, \sigma_2)$.

– $\texttt{Verify}(\mathsf{gpk}, \sigma, m)$: To verify a group signature σ on m, one checks that none of its elements is $1_{\mathbb{G}_1}$ or $1_{\mathbb{G}_2}$ and that the following equations hold:
- $e(\sigma_1, \widetilde{A}_1 \widetilde{B}^{-1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}) \cdot e(\sigma_2, \widetilde{A}_2) = e(\tau_1, \widetilde{\tau})$
- $e(\tau_2, \widetilde{g}) = e(g, \widetilde{\tau})$

in which case one outputs 1. Else, one returns 0.

– $\texttt{Open}(\mathsf{osk}, \mathsf{gpk}, \sigma, m)$: Before opening a signature, the opening authority first checks that it is valid. Else, it returns 0.
By using its secret key osk, the opening authority has the ability to decrypt any ciphertext C_i stored in $\mathbf{Reg}[i]$ and thus recover the elements $\widetilde{g}^{y_i} \in \mathbb{G}_2$ for all registered users. It can then checks, for each of them, whether the following equation holds:

$$e(\sigma_2, \widetilde{g}) \cdot e(\sigma_1, \widetilde{X}^{-1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}) = e(\sigma_1, \widetilde{g}^{y_i})$$

If there is no match, then the opening authority returns $\perp$. Else, let j be the corresponding user. The opening authority recovers the data $(g^{u_j}, g^{u_j \cdot y_j}, C_j, \pi_j, \mathsf{pk}_j, \mu_j)$ stored in $\mathbf{Reg}[j]$, commits to $\widetilde{g}^{y_j}$ and then outputs j along with a Groth-Sahai proof π that:
- $e(\sigma_2, \widetilde{g}) \cdot e(\sigma_1, \widetilde{X}^{-1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}) = e(\sigma_1, \widetilde{g}^{y_j})$
- $e(g^{u_j \cdot y_j}, \widetilde{g}) = e(g^{u_j}, \widetilde{g}^{y_j})$

– $\texttt{Judge}(\mathsf{gpk}, \sigma, m, i, \pi)$: To verify an opening, one checks that:
- $\texttt{Verify}(\mathsf{gpk}, \sigma, m) = 1$;
- $\Sigma.\texttt{Verify}(\mathsf{pk}_i, \mu_i, (g^{u_i}||g^{u_i \cdot y_i}||C_i||\pi_i)) = 1$;
- π is valid.

If at least one of these conditions is not satisfied, then one returns 0. Else, one returns 1.

Correctness. First note that at the end of the `Join` protocol, the user gets a FHS equivalence-class signature [25] on (g, g^y). Indeed, $(\tau_1, \tau_2, \widetilde{\tau}) = ((g^{\alpha_1} g^{y \cdot \alpha_2})^t, g^{1/t}, \widetilde{g}^{1/t})$.

To issue a group signature on m, the user first re-randomizes $(\tau_1, \tau_2, \widetilde{\tau})$ using s while updating the representative to $(g^r, g^{r \cdot y})$. The resulting tuple $(\tau_1', \tau_2', \widetilde{\tau}') = ((g^{r \cdot \alpha_1} g^{r \cdot y \cdot \alpha_2})^{t \cdot s}, g^{1/(t \cdot s)}, \widetilde{g}^{1/(t \cdot s)})$ is then still a FHS signature on the same equivalent class. He then generates a pair (σ_1, σ_2) where $(\sigma_1, \sigma_2^{m'})$ is a PS signature [34] on $m' = h(\tau_1'||\tau_2'||\widetilde{\tau}'||\sigma_1||m)$ using the same randomness r.

Therefore, such a group signature satisfies:

$$\begin{aligned} e(\sigma_1, \widetilde{A}_1 \widetilde{B}^{-1/m'}) \cdot e(\sigma_2, \widetilde{A}_2) &= e(g^r, \widetilde{g}^{\alpha_1 - \frac{x \cdot \alpha_2}{m'}}) \cdot e(g^{r(\frac{x}{m'} + y)}, \widetilde{g}^{\alpha_2}) \\ &= e(g, \widetilde{g})^{r(\alpha_1 + y \cdot \alpha_2)} \\ &= e(\tau_1', \widetilde{\tau}') \end{aligned}$$

and

$$e(\tau_2', \widetilde{g}) = e(g^{1/(t \cdot s)}, \widetilde{g}) = e(g, \widetilde{g}^{1/(t \cdot s)}) = e(g, \widetilde{\tau}')$$

Remark 3. Group signatures following the classical sign-encrypt-prove framework usually provide an efficient opening procedure. Indeed, the opening authority knows the corresponding decryption key and so can decrypt the ciphertext included in the group signature and then identify the signer. Unfortunately, there is no equivalent for constructions without encryption and in particular there is no longer a "master" key that the opening authority can use to break anonymity.

Constructions based on randomizable signatures [7, 22] circumvent this issue by forcing each user to provide a way to the opening authority to open their signatures. Concretely, during the `Join` protocol, each user must transmit some elements depending on their secret keys to this authority. Unfortunately this requirement does not fit the BSZ model [6] where `Join` is a two-party protocol between the user and the group manager. There are then two ways to solve this problem. Either we add the opening authority as an acting party in `Join` or we require that the user sends these elements to the group manager. The first solution is conceptually the simplest one but it modifies the original BSZ model. The second one is compatible with the latter but it requires additional primitives to ensure security. Indeed, the user cannot transmit such elements in clear (otherwise the group manager could break anonymity) so he must send them encrypted and prove (in a zero-knowledge way) that the resulting ciphertext is well-formed.

In this paper we choose to describe the most complex (second) solution since one can easily derive from it a group signature scheme complying with the first option. We will then need an IND-CCA2 secure public key encryption scheme that is compatible with NIZK proofs. In practice, one can choose for example [21, 32, 33] that nicely interact with Groth-Sahai proofs [29]. We note that efficiency is not really a concern here since this step of the `Join` protocol has no impact on the group signatures themselves.

Remark 4. We note that the security model of Bellare *et al* [6] already assumes a trusted `Setup` phase, so our construction perfectly fits this model on this point. However, this does not explain how to generate the public parameters in real-world conditions. In practice, it would be natural that the opening authority generates them. Regarding security, it would only be problematic for non-frameability if corruption of this entity occurred *before* `Setup`, but this is excluded by the model of [6]. We can also mitigate the risks by relying on a cooperative generation of the parameters, as in [15].

Security Analysis.

Theorem 5. – *Our group signature is traceable under the EUF-CMA security of the FHS signature scheme.*

– *In the generic group model, our group signature is non-frameable under the EUF-CMA security of Σ and the collision-resistance of h.*
– *Our group signature is CPA anonymous under the* SXDH *assumption and the IND-CCA2 security of Γ.*
– *Our group signature is selfless anonymous if it is non-frameable, if Γ is IND-CCA2 secure and if the* SXDH *assumption holds.*
– *Our group signature with merged opening authority and group manager is fully anonymous if it is traceable and if the* SXDH *assumption holds.*

The proofs are provided in Appendix A.

Remark 6. Theorem 5 shows that our scheme retains some security properties (namely CPA security) even when users' secret keys leak, contrarily to [7,34]. The fact that selfless anonymity depends on the non-frameability may seem surprising but this is due to the special opening process that the reduction $\mathcal{R}$ uses in our security proof. Informally, in the latter, $\mathcal{R}$ is able to open all signatures but the ones generated by the "challenge" user. To circumvent this problem $\mathcal{R}$ stores all the signatures it has produced on behalf of this user so that it will be able to recognize them if they are later submitted to the $\mathcal{O}\mathtt{Open}$ oracle. However, this works as long as the adversary is unable to forge signatures for this user, hence the non-frameability requirement.

The last statement of the theorem shows that we can achieve the strongest notion of anonymity if we additionally assume that the opening authority is also the group manager, as in the model of Bellare, Micciancio and Warinschi [5].

5 Efficiency

Scheme	Size (bits)	Cost	ROM ?	Model	Anonymity
BCNSW [7]	1280	3 e_1 + 1 e_T	yes	BMW	selfless
PS [34]	1024	2 e_1 + 1 e_T	yes	BMW	selfless
DS [22]	2048	5 e_1 + 1 e_2	yes	BSZ	CPA
DS* [22]	3328	5 e_1 + 6 e_2	yes	BSZ	full
BHKS [3]	3072	9 e_1 + 2 e_2	no	BMW	full
Ours	1536	5 e_1 + 1 e_2	no	BSZ	CPA & selfless
Ours*	1536	5 e_1 + 1 e_2	no	BMW	full

Fig. 2. Efficiency and Security comparison between related works and our construction. The signature size assumes that the bilinear groups are generated using Barreto-Naehrig curves [4], leading to a 256bit representation of the elements of $\mathbb{Z}_p$ and $\mathbb{G}_1$, and 512bit for the elements of $\mathbb{G}_2$. Cost refers to the computational cost of generating a signature. We only take into account the expensive operations, namely the exponentiations in $\mathbb{G}_1$ (e_1), $\mathbb{G}_2$ (e_2), and $\mathbb{G}_T$ (e_T). The BMW model is the one of [5] whereas the BSZ model is the one of [6] that we recall in Section 3. The last line of the table corresponds to our construction where the opening authority and the group manager are merged, which impacts security but not efficiency.

We compare in Figure 2 the efficiency of our scheme with the one of several constructions of the state-of-the-art. All of them are proven under interactive assumptions so we do not take this point into account in our comparison.

In the standard model, our group signature outperforms the recent construction of [3]: it halves both the signature size and the signature cost. We also note that it is competitive against the most efficient construction in the random oracle model [34]. Indeed, while the signature size remains larger, the computational cost is quite similar, according to [10], because we avoid the costly exponentiation in $\mathbb{G}_T$. More generally, our signer no longer needs to perform operation in $\mathbb{G}_T$ and so does not need to implement the artithmetic in $\mathbb{F}_{p^{12}}$, which is noticeable.

Conclusion

In this paper, we have introduced the most efficient group signature scheme in the standard model. Our construction is based on a tailored combination of the PS signature scheme and the FHS equivalence-class signature scheme, leading to a group signature consisting only of 4 elements in $\mathbb{G}_1$ and 1 in $\mathbb{G}_2$. Its security essentially relies on the one of these signature schemes which have been widely used in cryptographic protocols, although we need to slightly modify the proof of PS signature to fit our construction.

In the standard model, our scheme halves both the size and the computational cost compared to the most efficient alternative. It also significantly closes the gap with constructions in the ROM, showing that it is no longer necessary to choose between security and efficiency.

Acknowledgements

The authors are grateful for the support of the ANR through project ANR-16-CE39-0014 PERSOCLOUD.

References

1. Michel Abdalla and Bogdan Warinschi. On the minimal assumptions of group signature schemes. In Javier López, Sihan Qing, and Eiji Okamoto, editors, *ICICS 04*, volume 3269 of *LNCS*, pages 1–13. Springer, Heidelberg, October 2004.
2. Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 255–270. Springer, Heidelberg, August 2000.
3. Michael Backes, Lucjan Hanzlik, Kamil Kluczniak, and Jonas Schneider. Signatures with flexible public key: A unified approach to privacy-preserving signatures (full version). *IACR Cryptology ePrint Archive*, 2018:191, 2018.
4. Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In Bart Preneel and Stafford Tavares, editors, *SAC 2005*, volume 3897 of *LNCS*, pages 319–331. Springer, Heidelberg, August 2006.

5. 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, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 614–629. Springer, Heidelberg, May 2003.
6. Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, Heidelberg, February 2005.
7. Patrik Bichsel, Jan Camenisch, Gregory Neven, Nigel P. Smart, and Bogdan Warinschi. Get shorty via group signatures without encryption. In Juan A. Garay and Roberto De Prisco, editors, *SCN 10*, volume 6280 of *LNCS*, pages 381–398. Springer, Heidelberg, September 2010.
8. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg, August 2004.
9. Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 04*, pages 168–177. ACM Press, October 2004.
10. Joppe W. Bos, Craig Costello, and Michael Naehrig. Exponentiating in pairing groups. In Tanja Lange, Kristin Lauter, and Petr Lisonek, editors, *SAC 2013*, volume 8282 of *LNCS*, pages 438–455. Springer, Heidelberg, August 2014.
11. Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 427–444. Springer, Heidelberg, May / June 2006.
12. Ernie Brickell and Jiangtao Li. Enhanced privacy ID: A direct anonymous attestation scheme with enhanced revocation capabilities. *IEEE Trans. Dependable Sec. Comput.*, 9(3):345–360, 2012.
13. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg, May 2001.
14. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72. Springer, Heidelberg, August 2004.
15. Sébastien Canard, David Pointcheval, Olivier Sanders, and Jacques Traoré. Divisible E-cash made practical. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 77–100. Springer, Heidelberg, March / April 2015.
16. Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.
17. Ran Canetti, Oded Goldreich, and Shai Halevi. On the random-oracle methodology as applied to length-restricted signature schemes. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 40–57. Springer, Heidelberg, February 2004.
18. Sanjit Chatterjee and Alfred Menezes. On cryptographic protocols employing asymmetric pairings - the role of Ψ revisited. *Discrete Applied Mathematics*, 159(13):1311–1322, 2011.
19. David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer, Heidelberg, April 1991.
20. Liqun Chen and Jiangtao Li. Flexible and scalable digital signatures in TPM 2.0. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13*, pages 37–48. ACM Press, November 2013.

21. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 13–25. Springer, Heidelberg, August 1998.
22. 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*, pages 551–565. ACM Press, April 2018.
23. Nicolas Desmoulins, Roch Lescuyer, Olivier Sanders, and Jacques Traoré. Direct anonymous attestations with dependent basename opening. In Dimitris Gritzalis, Aggelos Kiayias, and Ioannis G. Askoxylakis, editors, *CANS 14*, volume 8813 of *LNCS*, pages 206–221. Springer, Heidelberg, October 2014.
24. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.
25. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Euf-cma-secure structure-preserving signatures on equivalence classes. *IACR Cryptology ePrint Archive*, 2014:944, 2014.
26. Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.
27. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.
28. Jens Groth. Fully anonymous group signatures without random oracles. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 164–180. Springer, Heidelberg, December 2007.
29. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Heidelberg, April 2008.
30. Aurore Guillevic. Comparing the pairing efficiency over composite-order and prime-order elliptic curves. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohassel, and Reihaneh Safavi-Naini, editors, *ACNS 13*, volume 7954 of *LNCS*, pages 357–372. Springer, Heidelberg, June 2013.
31. Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 491–511. Springer, Heidelberg, December 2014.
32. Charanjit S. Jutla and Arnab Roy. Relatively-sound NIZKs and password-based key-exchange. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 485–503. Springer, Heidelberg, May 2012.
33. Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 514–532. Springer, Heidelberg, May 2014.
34. David Pointcheval and Olivier Sanders. Short randomizable signatures. In Kazue Sako, editor, *CT-RSA 2016*, volume 9610 of *LNCS*, pages 111–126. Springer, Heidelberg, February / March 2016.
35. David Pointcheval and Olivier Sanders. Reassessing security of randomizable signatures. In Nigel P. Smart, editor, *CT-RSA 2018*, volume 10808 of *LNCS*, pages 319–338. Springer, Heidelberg, April 2018.

36. David Pointcheval, Olivier Sanders, and Jacques Traoré. Cut down the tree to achieve constant complexity in divisible E-cash. In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 61–90. Springer, Heidelberg, March 2017.
37. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Heidelberg, August 1990.

A Security Analysis

A.1 Proof of Anonymity

Our proofs of CPA anonymity and selfless anonymity are very similar and only differ in a game. In the following we will then consider an adversary against "anonymity" without specifying which property we consider except in Game 5 where the distinction is necessary. We discuss the case of full anonymity in Remark 7 at the end of this proof.

Let $\mathcal{A}$ be an adversary against the anonymity of our construction succeeding with probability ϵ. We define a sequence of games to show that this advantage is negligible. For each Game i we define $\mathtt{Adv}_i = |\mathtt{Pr}(S_i) - 1/2|$, where S_i is the probability that $\mathcal{A}$ succeeds in Game i. We additionally define $\mathtt{Adv}_{\mathsf{SXDH}}$ as the advantage against the SXDH problem.

Game 1. Our first game is exactly the one of anonymity of Figure 1 where the reduction $\mathcal{R}$ generates normally all the secret values and so is able to answer any oracle query. By definition, we have $\mathtt{Adv}_1 = \epsilon$.

Game 2. In our second game, $\mathcal{R}$ selects a random index $i^* \in [1, q_A]$, where q_A is a bound on the number of $\mathcal{O}\mathtt{Add}$ queries. $\mathcal{R}$ proceeds as usual but aborts if $\mathcal{A}$ queries (i_0, i_1, m) to the $\mathcal{O}\mathtt{Ch}$ oracle with $i_b \neq i^*$. The advantage of $\mathcal{A}$ in this new game is then at least $\frac{\epsilon}{q_A}$.

Game 3. In the third game, $\mathcal{R}$ generates a simulated common reference string crs and simulates all the zero-knowledge proofs. Any change in the behaviour of $\mathcal{A}$ can then be used against the zero-knowledge property of these proofs, which rely on SXDH in our setting. Therefore, $\mathtt{Adv}_3 \geq \mathtt{Adv}_2 - \mathtt{Adv}_{\mathsf{SXDH}}$.

Game 4. In the fourth game, $\mathcal{R}$ sets $\mathtt{opk}$ as the public key of a IND-CCA 2 experiment. It then uses the decryption oracle to decrypt the ciphertext C_i stored in $\mathbf{Reg}[i]$ for all users i and so can answer any query as usual. However, upon receiving the $\mathcal{O}\mathtt{Join}_U$ query on i^* (this query necessarily occurs because of Game 2), it proceeds normally except that it generates C as an encryption of a random element of $\mathbb{G}_2$ and simulates the proof. A change in the behaviour of $\mathcal{A}$ would imply an attack against the IND-CCA2 security of Γ, so we get $\mathtt{Adv}_4 \geq \mathtt{Adv}_3 - \mathtt{Adv}_{\mathsf{IND-CCA2}}$.

Game 5. In the fifth game, $\mathcal{R}$ stores every signature it generates on behalf of i^* in some register **Sig**. Upon receiving a $\texttt{Open}$ query for some pair (σ, m), it first checks whether $\sigma \in \textbf{Sig}$ in which case it returns i^* along with a simulated proof. Else, it returns $\texttt{Open}(\sigma, m)$.

We note that Game 5 is the same as Game 4 when we consider CPA anonymity since there is no $\mathcal{O}\texttt{Open}$ query in this case. For selfless anonymity, a difference only occurs when the adversary manages to submit a forged signature that can be traced back to i^*. However, such an adversary can straightforwardly be converted into an adversary against non-frameability. We then have $\texttt{Adv}_5 \geq \texttt{Adv}_4 - \texttt{Adv}^{nf}$.

Game 6. In the sixth game, $\mathcal{R}$ proceeds as in the previous game except that it answers to the $\mathcal{O}\texttt{Ch}$ query by returning a signature generated using a random key. The advantage of $\mathcal{A}$ can then only be 0. We prove below that the Games 5 and 6 cannot be distinguished under the SXDH assumption.

Proof. $\mathcal{R}$ receives a DDH challenge (g, g^a, g^b, g^z) in $\mathbb{G}_1$ and must then decide whether $z = a \cdot b$. It will then act as if $y = a$ for the secret key $\texttt{usk}_{i^*}$ of user i^*. This is not a problem since g^a is sufficient to issue group signatures and to join the group since Game 4. Moreover Game 5 ensures that $\mathcal{R}$ is able to answer any $\mathcal{O}\texttt{Open}$ query, even without knowing $\widetilde{g}^a$.

To answer the $\mathcal{O}\texttt{Ch}$ query for a message m, it selects a random $t \in \mathbb{Z}_p$ and computes a group signature σ as follows:

- $\tau_1 \leftarrow ((g^b)^{\alpha_1} \cdot (g^z)^{\alpha_2})^t$
- $(\tau_2, \widetilde{\tau}) \leftarrow (g^{1/t}, \widetilde{g}^{1/t})$
- $(\sigma_1, \sigma_2) \leftarrow (g^b, (g^b)^{x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)} \cdot (g^z))$

In any case, σ is a valid group signature on m, *i.e.* $\texttt{Verify}(\texttt{gpk}, \sigma, m)$ outputs 1. If $z = a \cdot b$, then σ is a valid signature issued by user i^* and $\mathcal{A}$ is still playing Game 5. Else, σ is a signature issued with a random key, independent of a and $\mathcal{A}$ is playing Game 6. Any change of behaviour of $\mathcal{A}$ between these two games can then be used against the DDH assumption in $\mathbb{G}_1$ and so against the SXDH assumption. We then have $\texttt{Adv}_6 \geq \texttt{Adv}_5 - \texttt{Adv}_{\mathsf{SXDH}}$. □

We then get the following result:

- $\frac{\epsilon}{q_A} \leq 2\ \texttt{Adv}_{\mathsf{SXDH}} + \texttt{Adv}_{\mathsf{IND-CCA2}}$ for any adversary succeeding against CPA anonymity with probability ϵ;
- $\frac{\epsilon}{q_A} \leq 2\ \texttt{Adv}_{\mathsf{SXDH}} + \texttt{Adv}_{\mathsf{IND-CCA2}} + \texttt{Adv}^{nf}$ for any adversary succeeding against selfless anonymity with probability ϵ;

which proves both CPA anonymity and selfless anonymity of our construction.

Remark 7. Let us now consider the case where the group manager and the opening authority are merged, as in [5]. As explained in remark 3, the use of IND-CCA2 encryption during the $\texttt{Join}$ protocol is only necessary when the opening authority is not involved in this process, which is no longer the case here. We can then discard Γ and remove Game 4 in the security proof.

In Game 5, $\mathcal{R}$ proceeds as follows. It still stores the signatures generated on behalf of i^* in **Sig** but now answer $\mathcal{O}\texttt{Open}$ queries on (σ, m) as follows:

– if $\sigma \in \mathbf{Sig}$, then it returns i^* along with a simulated proof π.
– if $\mathtt{Open}(\sigma, m)$ returns (i, π) or 0, then it forwards this answer to the adversary.
– if $\mathtt{Open}(\sigma, m)$ returns $\perp$, then it returns i^* along with a simulated proof π.

We note that a problem may only occur in the third case if the adversary managed to submit a group signature that cannot be traced back to a registered user. However, this would mean that $\mathcal{A}$ is a valid adversary against traceability, which is unlikely.

All the other games remain unchanged so we get the following result:

– $\frac{\epsilon}{q_A} \leq 2\,\mathtt{Adv}_{\mathsf{SXDH}} + \mathtt{Adv}^{tra}$ for any adversary succeeding against full anonymity with probability ϵ in the model of [5].

A.2 Proof of Traceability

We prove in this section that any untraceable group signature can be used to construct a forgery against the FHS equivalence-class signature scheme. More specifically, let $\mathcal{A}$ be an adversary succeeding against the traceability of our group signature with probability ϵ, then $\mathcal{A}$ can be converted into an adversary succeeding against the EUF-CMA security of FHS signature with the same probability.

Technically, $\mathcal{A}$ can succeed by returning a valid signature σ on m that either foils the opening process or that can be opened but for which it is impossible to produce a valid proof of opening. We can exclude the latter case in our construction because of the correctness and of the soundness of Groth-Sahai proofs.

Our reduction $\mathcal{R}$ generates the public parameters as usual except that it does not discard x after generating X and $\widetilde{X}$. It then gets the public key $\widetilde{A}_1$ and $\widetilde{A}_2$ from the EUF-CMA challenger and sets the group manager's public key as $(\widetilde{A}_1, \widetilde{A}_2, \widetilde{A}_2^x)$. By using its signing oracle, it is able to handle any Join query so the simulation is perfect. At the end of the game, $\mathcal{A}$ then outputs with probability ϵ an untraceable group signature σ on m. If we parse σ as $(\tau_1, \tau_2, \widetilde{\tau}, \sigma_1, \sigma_2)$, this means that:

1. $e(\sigma_1, \widetilde{A}_1 \widetilde{B}^{-1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}) \cdot e(\sigma_2, \widetilde{A}_2) = e(\tau_1, \widetilde{\tau})$
2. $e(\tau_2, \widetilde{g}) = e(g, \widetilde{\tau})$
3. $e(\sigma_2, \widetilde{g}) \cdot e(\sigma_1, \widetilde{X}^{-1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}) \neq e(\sigma_1, \widetilde{g}^{y_i})$ for all $\widetilde{g}^{y_i}$ stored (encrypted) in $\mathbf{Reg}[i]$

Equation 1 is equivalent to:

$$e(\sigma_1, \widetilde{A}_1) \cdot e(\sigma_2 \cdot \sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}, \widetilde{A}_2) = e(\tau_1, \widetilde{\tau})$$

which means (together with equation 2) that $(\tau_1, \tau_2, \widetilde{\tau})$ is a valid FHS signature on $(\sigma_1, \sigma_2 \cdot \sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)})$. However, $(\tau_1, \tau_2, \widetilde{\tau})$ will be considered as a valid forgery only if $(\sigma_1, \sigma_2 \cdot \sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)})$ does not belong to the equivalence class of a message submitted to the signing oracle.

Let $\mathcal{S} = \{(h_i, h_i^{y_i})\}_{i=1}^{q}$, for $h_i \in \mathbb{G}_1$ be the set of queried messages. All these messages were involved in a `Join` query during which the group manager received (and stored) $\widetilde{g}^{y_i}$ (encrypted). Therefore, if (μ_1, μ_2) belongs to the equivalent class of an element of $\mathcal{S}$, then $\exists i \in [1, q]$ such that $e(\mu_1, \widetilde{g}^{y_i}) = e(\mu_2, \widetilde{g})$.

Let us then assume that $(\sigma_1, \sigma_2 \cdot \sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)})$ satisfies the previous condition, *i.e.* that there is i such that:

$$e(\sigma_1, \widetilde{g}^{y_i}) = e(\sigma_2 \cdot \sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}, \widetilde{g})$$

We then have:

$$\begin{aligned} e(\sigma_1, \widetilde{g}^{y_i}) &= e(\sigma_2, \widetilde{g}) \cdot e(\sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}, \widetilde{g}) \\ &= e(\sigma_2, \widetilde{g}) \cdot e(\sigma_1, \widetilde{X}^{-1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)}) \end{aligned}$$

which contradicts equation 3. The pair $(\sigma_1, \sigma_2 \cdot \sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)})$ has then never been signed, nor any representative of the same equivalence class, which means that $(\tau_1, \tau_2, \widetilde{\tau})$ along with $(\sigma_1, \sigma_2 \cdot \sigma_1^{-x/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||m)})$ is a valid forgery against the EUF-CMA security of the FHS scheme.

A.3 Proof of Non-Frameability

A successful adversary $\mathcal{A}$ against the non-frameability of our scheme is able to forge a valid group signature $(\tau_1, \tau_2, \widetilde{\tau}, \sigma_1, \sigma_2)$ on a message m that can be traced back to some honest user i. We then distinguish three cases:

- Case 1: The elements $(g^u, g^{u \cdot y})$ stored in $\mathbf{Reg}[\mathrm{i}]$ are not the ones sent by user i when he joined the group.
- Case 2: The user i has issued a signature $(\tau_1', \tau_2', \widetilde{\tau}', \sigma_1', \sigma_2') \neq (\tau_1, \tau_2, \widetilde{\tau}, \sigma_1, \sigma_2)$ on m' such that $h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||\sigma_2||m) = h(\tau_1'||\tau_2'||\widetilde{\tau}'||\sigma_1'||\sigma_2'||m')$.
- Case 3: None of the previous events occur.

We recall that a user can be accused of having produced a signature only if the opening is valid, *i.e.* only if the `Judge` algorithm outputs 1. One step of the `Judge` algorithm is to check that the user i has indeed signed the elements stored in $\mathbf{Reg}[i]$, *i.e.* that there is a signature μ in $\mathbf{Reg}[i]$ such that $\Sigma.\mathtt{Verify}(\mathsf{pk}_i, \mu, (g^u||g^{u \cdot y}||C_i||\pi_i)) = 1$. Therefore, case 1 straightforwardly implies a forgery against Σ, which is assumed to be EUF-CMA.

Similarly, case 2 implies a collision of the hash function h so we can focus on the last case.

Intuitively, we would like to directly rely on the EUF-CMA security of PS signatures. Indeed, a valid forgery accusing i must satisfy

$$\begin{aligned} & e(\sigma_2, \widetilde{g}) \cdot e(\sigma_1, \widetilde{X}^{-1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||\sigma_2||m)}) = e(\sigma_1, \widetilde{g}^{y}) \\ \Leftrightarrow & e(\sigma_2, \widetilde{g}) = e(\sigma_1, \widetilde{X}^{1/h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||\sigma_2||m)} \cdot \widetilde{g}^{y}) \end{aligned}$$

which is essentially the verification equation of a PS signature on $h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||\sigma_2||m)$. Unfortunately, it seems hard to construct a reduction on this idea because the signing oracle of PS signatures is not sufficient to answer $\mathcal{O}\mathtt{Sign}$ queries. Indeed, the element $\tau_1 = (g^{r\cdot\alpha_1} g^{y\cdot r\cdot\alpha_2})^t$ cannot be deduced from a PS signature, even given (α_1, α_2). We then need to extend the original proof of PS signatures to show that EUF-CMA still holds when τ_1 is provided to the adversary.

Lemma 8. *Let $\mathcal{A}$ be an adversary whose forgery corresponds to case 3. In the generic group model, $\mathcal{A}$ cannot succeed with probability greater than $O(q_S(q_S + q_G)^2/p)$, where q_G is a bound on the group oracle queries and q_S is a bound on the $\mathcal{O}\mathtt{Sign}$ oracle queries.*

Proof. In the non-frameability game the adversary has access to gsk, osk and all the users' secret keys but the one (g^y) of user i. We also note that it does not know the scalar x that has been used to generate the public parameters $X = g^x$ and $\widetilde{X} = \widetilde{g}^x$. In the following, we will then associate group elements with polynomials whose formal variables are x and y along with the randomness used to answer join or signing queries. We first prove that $\mathcal{A}$ is unable to symbolically produce a forgery on behalf of i and then show that an accidental validity is very unlikely.

For $j \in [1, q_S]$, let $(\tau_{1,j}, \tau_{2,j}, \widetilde{\tau}_j, \sigma_{1,j}, \sigma_{2,j}) = ((g^{r_j\cdot\alpha_1} \cdot g^{r_j\cdot y\cdot\alpha_2})^{t_j}, g^{1/t_j}, \widetilde{g}^{1/t_j}, g^{r_j}, g^{r_j(x/h_j+y)})$ be the j-th group signature returned by $\mathcal{O}\mathtt{Sign}$ on input (i, m_j), where $h_j = h(\tau_{1,j}||\tau_{2,j}||\widetilde{\tau}_j||\sigma_{1,j}||\sigma_{2,j}||m_j)$.

In the generic group model, any group element must have been generated through queries to the oracle of the internal law of the group. This means that there are known coefficients $\{a_k, b_k, c_k, d_k\}_{k=1}^4$ and $\{a_{k,j}, b_{k,j}, c_{k,j}, d_{k,j}\}_{k=5,j=1}^{k=8,j=q_S}$ such that

$$\tau_1 = g^{a_1} \cdot X^{a_2} \cdot (g^u)^{a_3} \cdot (g^{u\cdot y})^{a_4} \cdot \prod_j \tau_{1,j}^{a_{5,j}} \cdot \tau_{2,j}^{a_{6,j}} \cdot \sigma_{1,j}^{a_{7,j}} \cdot \sigma_{2,j}^{a_{8,j}}$$

$$\tau_2 = g^{b_1} \cdot X^{a_2} \cdot (g^u)^{b_3} \cdot (g^{u\cdot y})^{b_4} \cdot \prod_j \tau_{1,j}^{b_{5,j}} \cdot \tau_{2,j}^{b_{6,j}} \cdot \sigma_{1,j}^{b_{7,j}} \cdot \sigma_{2,j}^{b_{8,j}}$$

$$\sigma_1 = g^{c_1} \cdot X^{c_2} \cdot (g^u)^{c_3} \cdot (g^{u\cdot y})^{c_4} \cdot \prod_j \tau_{1,j}^{c_{5,j}} \cdot \tau_{2,j}^{c_{6,j}} \cdot \sigma_{1,j}^{c_{7,j}} \cdot \sigma_{2,j}^{c_{8,j}}$$

$$\sigma_2 = g^{d_1} \cdot X^{d_2} \cdot (g^u)^{d_3} \cdot (g^{u\cdot y})^{d_4} \cdot \prod_j \tau_{1,j}^{d_{5,j}} \cdot \tau_{2,j}^{d_{6,j}} \cdot \sigma_{1,j}^{d_{7,j}} \cdot \sigma_{2,j}^{d_{8,j}}$$

and a'_1, a'_2 and $\{a'_{3,j}\}_{j=1}^{q_S}$ such that

$$\widetilde{\tau} = \widetilde{g}^{a'_1} \cdot \widetilde{X}^{a'_2} \prod_j \widetilde{\tau}_j^{a'_{3,j}}$$

Since we here consider Case 3, the fact that $(\tau_1, \tau_2, \widetilde{\tau}, \sigma_1, \sigma_2)$ is a valid signature on m that can be traced back to i means that:

1. $e(\sigma_1, \widetilde{A}_1 \widetilde{B}^{-1/h^*}) \cdot e(\sigma_2, \widetilde{A}_2) = e(\tau_1, \widetilde{\tau})$
2. $e(\tau_2, \widetilde{g}) = e(g, \widetilde{\tau})$
3. $e(\sigma_2, \widetilde{g}) \cdot e(\sigma_1, \widetilde{X}^{-1/h^*}) = e(\sigma_1, \widetilde{g}^y)$

where $h^* = h(\tau_1||\tau_2||\widetilde{\tau}||\sigma_1||\sigma_2||m)$. The last equation is equivalent to $e(\sigma_2, \widetilde{g}) = e(\sigma_1, \widetilde{g}^y \cdot \widetilde{X}^{1/h^*})$, which gives the following relation:

$$\begin{aligned}
&\sum_j [d_{5,j} \cdot r_j \cdot t_j(\alpha_1 + y \cdot \alpha_2) + d_{6,j}/t_j + d_{7,j} \cdot r_j + d_{8,j} \cdot r_j(x/h_j + y)] \\
&+ d_1 + x \cdot d_2 + u \cdot d_3 + u \cdot y \cdot d_4 = (y + x/h^*)(c_1 + x \cdot c_2 + u \cdot c_3 + u \cdot y \cdot c_4 \\
&+ \sum_j [c_{5,j} \cdot r_j \cdot t_j(\alpha_1 + y \cdot \alpha_2) + c_{6,j}/t_j + c_{7,j} \cdot r_j + c_{8,j} \cdot r_j(x/h_j + y))])
\end{aligned}$$

Due to the factor $(y + x/h^*)$, each monomial of the right member will be of degree at least 1 in x or y. We can therefore conclude that $d_1 = d_3 = d_{5,j} = d_{6,j} = d_{7,j} = 0 \ \forall j \in [1, q_S]$ and thus get :

$$\begin{aligned}
&\sum_j [d_{8,j} \cdot r_j(x/h_j + y)] + x \cdot d_2 + u \cdot y \cdot d_4 = (y + x/h^*)(c_1 + x \cdot c_2 + u \cdot c_3 \\
&+ u \cdot y \cdot c_4 + \sum_j [c_{5,j} \cdot r_j \cdot t_j(\alpha_1 + y \cdot \alpha_2) + c_{6,j}/t_j + c_{7,j} \cdot r_j + c_{8,j} \cdot r_j(x/h_j + y))])
\end{aligned}$$

Moreover, if we note that the left member does not contain any monomial of degree 2 in x or y, or any monomial in $x \cdot y$, we have $c_2 = c_4 = c_{5,j} = c_{8,j} = 0$ $\forall j$:

$$\begin{aligned}
&\sum_j [d_{8,j} \cdot r_j(x/h_j + y)] + x \cdot d_2 + u \cdot y \cdot d_4 \\
&= (y + x/h^*)(c_1 + u \cdot c_3 + \sum_j [c_{6,j}/t_j + c_{7,j} \cdot r_j])
\end{aligned}$$

The right member contains a term in $u \cdot x$ contrarily to the left one, which implies that $c_3 = 0$ and so that $d_4 = 0$. If we consider the resulting equation as an equality between polynomials in y we get

– degree 0 coefficients:

$$\sum_j [d_{8,j} \cdot r_j \cdot x/h_j] + x \cdot d_2 = x/h^*(c_1 + \sum_j [c_{6,j}/t_j + c_{7,j} \cdot r_j])$$

meaning that $c_{6,j} = 0$ (no other term involving t_j), $d_2 = c_1/h^*$ and $d_{8,j} = \frac{c_{7,j} \cdot h_j}{h^*}$.

– degree 1 coefficients:

$$\sum_j [d_{8,j} \cdot r_j \cdot y] = y(c_1 + \sum_j [c_{7,j} \cdot r_j])$$

which implies that $c_1 = 0$ and so $d_2 = 0$. Moreover, if we use the previous relation $d_{8,j} = \frac{c_{7,j} \cdot h_j}{h^*}$, we get $\frac{c_{7,j} \cdot h_j}{h^*} = c_{7,j}$ which is only possible for the values j such that $h_j = h^*$ when $c_{7,j} \neq 0$. However, we exclude this event in case 3, which means that $d_{8,j} = c_{7,j} = 0$ for all $j \in [1, q_s]$. Therefore, $\sigma_1 = \sigma_2 = 1_{\mathbb{G}_1}$ and the forgery is invalid.

Now, let us assess the probability of an accidental validity, *i.e.* when two different polynomials involved in group oracle queries evaluate to the same value. To remove the denominator $1/t_j$ we raise each polynomial to the power $\prod_j t_j$ and thus get polynomials of degree at most q_S+4. Since there are at most $5q_S+6+q_G$ polynomials, there are $(5q_S + 6 + q_G)^2/2$ pairs that could evaluate to the same value. By the Schwartz-Zippel lemma, this can occur with probability at most $(q_S + 4)(5q_S + 6 + q_G)^2/2p$, which is negligible.