Fully-Anonymous Short Dynamic Group Signatures Without Encryption

David Derler and Daniel Slamanig

IAIK, Graz Universtity of Technology, Austria
{david.derler|daniel.slamanig}@tugraz.at

Abstract. Group signatures are an important privacy-enhancing tool which allow members of a group to anonymously produce signatures on behalf of the group. Ideally, group signatures are dynamic and thus allow to dynamically enroll new members to a group. For such schemes Bellare et al. (CT-RSA'05) proposed a strong security model (BSZ model) that preserves anonymity of a group signature even if an adversary can see arbitrary key exposures or arbitrary openings of other group signatures. All previous constructions achieving this strong security notion follow the so called sign-encrypt-prove (SEP) paradigm. In contrast, all known constructions which avoid this paradigm and follow the alternative "without encryption" paradigm introduced by Bichsel et al. (SCN'10), only provide a weaker notion of anonymity (which can be problematic in practice). Until now, it was not clear if constructions following this paradigm, while also being secure in the strong BSZ model, even exist. In this paper we positively answer this question by providing a novel approach to dynamic group signature schemes following this paradigm, which is a composition of structure preserving signatures on equivalence classes (Asiacrypt'14) and other standard primitives. Our results are interesting for various reasons: We can prove our construction following this "without encryption" paradigm secure in the strong BSZ model without requiring random oracles. Moreover, when opting for an instantiation in the ROM, the so obtained scheme is extremely efficient and outperforms existing constructions following the SEP paradigm and being secure in the BSZ model regarding computational efficiency. Regarding constructions providing a weaker anonymity notion than BSZ, we surprisingly even outperform the popular short BBS group signature scheme (Crypto'04) by some orders of magnitude and thereby obtain shorter signatures.

1 Introduction

Group signatures, initially introduced by Chaum and van Heyst [CvH91], allow a group manager to set up a group so that every member of this group can later anonymously sign messages on behalf of the group. Thereby, a dedicated authority (called opening authority) can open a given group signature to determine the identity of the actual signer. Group signatures were first rigorously formalized for static groups by Bellare et al. in [BMW03]. In this setting, all members are determined at setup and are also given their honestly generated keys at setup. This model was later extended to the dynamic case by Bellare et al. in [BSZ05] (henceforth denoted as BSZ model), where new group members can be dynamically enrolled to the group. Further, it separates the role of the issuer and the opener such that they can operate independently. Moreover, the BSZ model requires a strong anonymity notion, where anonymity of a group signature is preserved even if the adversary can see arbitrary key exposures and arbitrary openings of other group signatures. A slightly weaker model, which is used to prove the security (and in particular anonymity) of the popular BBS group signature scheme was introduced by Boneh et al. [BBS04]. This model is a relaxation of the BSZ model, and in particular weakens anonymity so that the adversary can not request openings for signatures. Henceforth, we refer

The authors have been supported by EU H2020 project Prismacloud, grant agreement n°644962.

to this anonymity notion as CPA-full anonymity, whereas we use CCA2-full anonymity to refer to anonymity in the sense of BSZ.

A widely used paradigm to construct group signatures is the sign-encrypt-proof (SEP) paradigm [CS97]. Here, a signature is essentially an encrypted membership certificate together with a signature of knowledge, where the signer demonstrates knowledge of some signed value in the ciphertext [ACJT00, BBS04, NS04, BSZ05, DP06, BW07, BW06, Gro07, LPY15]. As an alternative to this paradigm, Bichsel et al. in [BCN+10] proposed an elegant design paradigm for group signatures which does not require encryption to produce signatures. Essentially, they use a signature scheme which supports (1) randomization of signatures so that multiple randomized versions of the same signature are unlinkable, and (2) efficiently proving knowledge of a signed value. In their construction, on joining the group, the issuer uses such a signature scheme to sign a commitment to the user's secret key. The user can then produce a group signature for a message by randomizing the signature and computing a signature of knowledge for the message, which demonstrates knowledge of the signed secret key. Bichsel et al. proposed an instantation based on the randomizable pairing-based Camensich-Lysyanskaya (CL) signature scheme [CL04] (whose EUF-CMA security is based on the interactive LRSW assumption). Recently, Pointcheval and Sanders [PS16] proposed another randomizable signature scheme (whose EUF-CMA security is proven in the generic group model), which allows to instantiate the approach due to Bichsel et al. more efficiently. For completeness, we also mention a very recent construction of group signatures from lattice assumptions by Libert et al. [LLM+16] following this paradigm.

However, a drawback of existing constructions following the "without encryption" paradigm is that they rely on a security model that is weaker than the BSZ model [BSZ05]. In particular, anonymity only holds for users whose keys do not leak, which essentially means that once a user key leaks, all previous signatures of this user can potentially be attributed to this user. Furthermore, the model in [BCN+10] assumes that the opening authority and the issuing authority are one entity, meaning that the issuer can identify all signers when seeing group signatures. This can be highly problematic in practical applications of group signatures.

Contribution. In this paper we propose a novel approach to construct group signatures following the paradigm of Bichsel et al., i.e., without including a ciphertext in the group signature. In particular, our approach is a composition of structure preserving signatures on equivalence classes (SPS-EQ) [HS14], conventional digital signatures, public key encryption, non-interactive zero-knowledge proofs, and signatures of knowledge. Although these tools may sound quite heavy, we obtain surprisingly efficient group signatures, which can be proven secure in the strongest model for dynamic group signatures, i.e., the BSZ model. In doing so, we obtain the first construction following the "without encryption" paradigm which achieves this strong security notion (i.e., CCA2-full anonymity). Thus, we can positively answer the question whether such constructions are possible at all. When opting for an instantiation of our construction in the random oracle model, and comparing it with existing constructions having security proofs in the strong BSZ model, we outperform them in terms of computational efficiency. Moreover, when compared to the popular BBS group signature scheme [BBS04] (which achieves CPA-full anonymity), we surprisingly obtain significantly better computational efficiency and even shorter signatures. Finally, when compared to existing instantiations in the vein of Bichsel et al. (who provide a substantially weaker anonymity notion and do not separate the issuer and the opener), we obtain comparable computational efficiency.

2 Preliminaries

In this section, we provide some preliminaries and recall the required primitives.

Notation. Let $x \xleftarrow{R} X$ denote the operation that picks an element x uniformly at random from a finite set X. A function $\epsilon : \mathbb{N} \rightarrow \mathbb{R}^+$ is called negligible if for all $c > 0$ there is a k_0 such that $\epsilon(k) < 1/k^c$ for all $k > k_0$. In the remainder of this paper, we use ϵ to denote such a negligible function. We write $y \leftarrow A(x)$ to denote that the output of algorithm A on input x is assigned to y and write $y \leftarrow A(x;\ r)$ to make the random coins r of A explicit.

Let $\mathbb{G}_1 = \langle P \rangle$, $\mathbb{G}_2 = \langle \hat{P} \rangle$, and $\mathbb{G}_T$ be groups of prime order p. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a map, where it holds for all $(P, \hat{Q}, a, b) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{Z}_p^2$ that $e(aP, b\hat{Q}) = e(P, \hat{Q})^{ab}$, and $e(P, \hat{P}) \neq 1$, and e is efficiently computable. We assume the Type-3 setting, where $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable isomorphism $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$ is known.

Definition 1 (Bilinear Group Generator). *Let* BGGen *be an algorithm which takes a security parameter* κ *and generates a bilinear group* $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P, \hat{P})$ *in the Type-3 setting, where the common group order* p *of the groups* $\mathbb{G}_1, \mathbb{G}_2$ *and* $\mathbb{G}_T$ *is a prime of bitlength* κ*,* e *is a pairing and* P *and* $\hat{P}$ *are generators of* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*, respectively.*

Based on this, we provide the required cryptographic hardness assumptions.

Decisional Diffie-Hellman Assumption (DDH). Let $\mathbb{G} = \langle P \rangle$ be a group of prime order p, such that $\log_2 p = \kappa$. Then, for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that:

$$\Pr\left[\begin{array}{l} b \xleftarrow{R} \{0,1\},\ r, s, t \xleftarrow{R} \mathbb{Z}_p, \\ b^* \leftarrow \mathcal{A}(P, rP, sP, b \cdot (rs) + (1-b) \cdot tP) \end{array} \ :\ b = b^*)\right] \leq 1/2 + \epsilon(\kappa).$$

Symmetric External Diffie-Hellman Assumption (SXDH). Let BG be a bilinear group generated by BGGen. Then, the SXDH assumption states that the DDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$.

Additionally, we introduce a plausible assumption in the Type-3 bilinear group setting.

Computational co-Diffie-Hellman Inversion Assumption (co-CDHI): Let $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa)$. The co-CDHI assumption states that for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that:

$$\Pr\left[a \xleftarrow{R} \mathbb{Z}_p,\ C \leftarrow \mathcal{A}(\mathsf{BG}, aP, 1/a\hat{P})\ :\ C = 1/aP\right] \leq \epsilon(\kappa).$$

Digital Signature Schemes. Subsequently, we recall a definition of digital signature schemes.

Definition 2 (Digital Signatures). *A digital signature scheme* Σ *is a triple* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *of PPT algorithms, which are defined as follows:*

$\mathsf{KeyGen}(1^\kappa):$ *This algorithm takes a security parameter* κ *as input and outputs a secret (signing) key* sk *and a public (verification) key* pk *with associated message space* $\mathcal{M}$ *(we may omit to mention the message space* $\mathcal{M}$*).*

$\mathsf{Sign}(\mathsf{sk}, m):$ *This algorithm takes a secret key* sk *and a message* $m \in \mathcal{M}$ *as input and outputs a signature* σ*.*

$\mathsf{Verify}(\mathsf{pk}, m, \sigma):$ *This algorithm takes a public key* pk*, a message* $m \in \mathcal{M}$ *and a signature* σ *as input and outputs a bit* $b \in \{0, 1\}$*.*

Besides correctness we require existential unforgeability under adaptively chosen message attacks (EUF-CMA) [GMR88]. Subsequently, we recall formal definitions of these properties.

Definition 3 (Correctness). *A digital signature scheme is correct, if for all* κ*, all* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$ *and all* $m \in \mathcal{M}$ *it holds that* $\Pr[\mathsf{Verify}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1] = 1$.

Definition 4 (EUF-CMA). *A digital signature scheme is* EUF-CMA *secure, if for all PPT adversaries $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\left[(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa),\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{Sign}(\mathsf{sk},\cdot)}}(\mathsf{pk}) \ : \ \begin{array}{r} \mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = 1 \ \wedge \\ m^* \notin Q^{\mathsf{Sign}} \end{array} \right] \leq \epsilon(\kappa) \ ,$$

where $\mathcal{A}$ has access to an oracle $\mathcal{O}^{\mathsf{Sign}}$ that allows to execute the Sign *algorithm and the environment keeps track of all message queried to $\mathcal{O}^{\mathsf{Sign}}$ via Q^{Sign}.*

Public Key Encryption. We also require public key encryption, which we recall below.

Definition 5. *A public key encryption scheme Ω is a triple* (KeyGen, Enc, Dec) *of PPT algorithms, which are defined as follows:*

$\mathsf{KeyGen}(1^\kappa)$: This algorithm takes a security parameter κ as input and outputs a keypair $(\mathsf{sk}, \mathsf{pk})$. We assume that the message space $\mathcal{M}$ is implicitly defined by pk.

$\mathsf{Enc}(\mathsf{pk}, m)$: This algorithm takes a public key pk and a message $m \in \mathcal{M}$ as input and outputs a ciphertext c or $\perp$.

$\mathsf{Dec}(\mathsf{sk}, c)$: This algorithm takes a secret key sk and a ciphertext c as input and outputs a message $m \in \mathcal{M}$ or $\perp$.

We require a Ω to be correct and IND-T secure, which are formally defined as follows.

Definition 6 (Correctness). *A public key encryption scheme is correct if it holds for all κ, for all $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, and for all messages $m \in \mathcal{M}$ that $\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m] = 1$.*

Definition 7 (IND-T **Security).** *Let $\mathsf{T} \in \{\mathsf{CPA}, \mathsf{CCA2}\}$. A public key encryption scheme is* IND-T *secure, if for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\left[\begin{array}{l} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa),\ (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}_\mathsf{T}}(\mathsf{pk}), \\ b \xleftarrow{R} \{0,1\},\ c \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b),\ b^* \leftarrow \mathcal{A}^{\mathcal{O}_\mathsf{T}}(c, \mathsf{st}) \end{array} \ : \ \begin{array}{r} b = b^* \ \wedge \\ c \notin Q^{\mathsf{Dec}} \end{array} \right] \leq 1/2 + \epsilon(\kappa),$$

where $\mathcal{O}_\mathsf{T} \leftarrow \emptyset$ and $Q^{\mathsf{Dec}} \leftarrow \emptyset$ if $\mathsf{T} = \mathsf{CPA}$, and $\mathcal{O}_\mathsf{T} \leftarrow \{\mathcal{O}^{\mathsf{Dec}}(\mathsf{sk}, \cdot)\}$ and Q^{Dec} denotes the list of queries to the decryption oracle in the second stage if $\mathsf{T} = \mathsf{CCA2}$.

Non-Interactive Zero-Knowledge Proof Systems. Now, we recall a standard definition of non-interactive zero-knowledge proof systems. Therefore, let L_R be an **NP**-language with witness relation $R : L_R = \{x \mid \exists\ w : R(x, w) = 1\}$.

Definition 8 (Non-Interactive Zero-Knowledge Proof System). *A non-interactive proof system Π is a tuple of algorithms* (Setup, Proof, Verify)*, which are defined as follows:*

$\mathsf{Setup}(1^\kappa)$: This PPT algorithm takes a security parameter κ as input, and outputs a common reference string crs.

$\mathsf{Proof}(\mathsf{crs}, x, w)$: This algorithm takes a common reference string crs, a statement x, and a witness w as input, and outputs a proof π.

$\mathsf{Verify}(\mathsf{crs}, x, \pi)$: This PPT algorithm takes a common reference string crs, a statement x, and a proof π as input, and outputs a bit $b \in \{0, 1\}$.

If, in addition, Proof runs in polynomial time we talk about a non-interactive argument system. We require Π to be complete, sound, and adaptively zero-knowledge. Subsequently, we recall formal definition of those properties (adapted from [BGI14]).

Definition 9 (Completeness). *A non-interactive proof system is* complete*, if for every adversary $\mathcal{A}$ it holds that*

$$\Pr\left[\begin{matrix}\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ (x,w) \leftarrow \mathcal{A}(\mathsf{crs}), \\ \pi \leftarrow \mathsf{Proof}(\mathsf{crs},x,w)\end{matrix} \ :\ \begin{matrix}\mathsf{Verify}(\mathsf{crs},x,\pi)=1 \\ \wedge\ (x,w)\in R\end{matrix}\right] = 1.$$

Definition 10 (Soundness). *A non-interactive proof system is* sound*, if for every PPT adversary $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ (x,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\ :\ \mathsf{Verify}(\mathsf{crs},x,\pi)=1\ \wedge\ x \notin L_R\right] \le \epsilon(\kappa).$$

If $\epsilon = 0$, we have perfect soundness.

Definition 11 (Adaptive Zero-Knowledge). *A non-interactive proof system is* adaptively zero-knowledge*, if there exists a PPT simulator $S = (S_1, S_2)$ such that for every PPT adversary $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\left|\begin{matrix}\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa)\ :\ \mathcal{A}^{\mathcal{P}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs})=1\right]\ - \\ \Pr\left[(\mathsf{crs},\tau) \leftarrow S_1(1^\kappa)\ :\ \mathcal{A}^{\mathcal{S}(\mathsf{crs},\tau,\cdot,\cdot)}(\mathsf{crs})=1\right]\end{matrix}\right| \le \epsilon(\kappa),$$

where, τ denotes a simulation trapdoor. Thereby, $\mathcal{P}$ and $\mathcal{S}$ return $\perp$ if $(x,w) \notin R$ or $\pi \leftarrow \mathsf{Proof}(\mathsf{crs},x,w)$ and $\pi \leftarrow S_2(\mathsf{crs},\tau,x)$, respectively, otherwise.

If $\epsilon = 0$, we have perfect adaptive zero-knowledge.

Signatures of Knowledge. Below we recall signatures of knowledge (SoKs) [CL06], where L_R is as above. For the formal notions we follow [BCC+15] and use a stronger generalization of the original extraction property termed f-extractability. A signature of knowledge (SoK) for L_R is defined as follows.

Definition 12. *A* SoK *is a tuple of PPT algorithms* (Setup, Sign, Verify)*, which are defined as follows:*

$\mathsf{Setup}(1^\kappa)$: This algorithm takes a security parameter κ as input and outputs a common reference string crs. We assume that the message space $\mathcal{M}$ is implicitly defined by crs.

$\mathsf{Sign}(\mathsf{crs},x,w,m)$: This algorithm takes a common reference string crs, a word x, a witness w, and a message m as input and outputs a signature σ.

$\mathsf{Verify}(\mathsf{crs},x,m,\sigma)$: This algorithm takes a common reference string crs, a word x, a message m, and a signature σ as input and outputs a bit $b \in \{0,1\}$.

Definition 13 (Correctness). *A* SoK *with respect to L_R is correct, if there exists a negligible function $\epsilon(\cdot)$ such that for all $x \in L_R$, for all w such that $(x,w) \in R$, and for all $m \in \mathcal{M}$ it holds that*

$$\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ \sigma \leftarrow \mathsf{Sign}(\mathsf{crs},x,w,m) : \mathsf{Verify}(\mathsf{crs},x,m,\sigma)=1\right] \ge 1-\epsilon(\kappa).$$

Definition 14 (Simulatability). *A* SoK *with respect to L_R is simulatable, if there exists a PPT simulator $\mathcal{S} = (\mathsf{SimSetup}, \mathsf{SimSign})$ such that for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that it holds that*

$$\left|\begin{matrix}\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ b \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{crs},\cdot,\cdot,\cdot)}(\mathsf{crs})\ :\ b=1\right]\ - \\ \Pr\left[(\mathsf{crs},\tau) \leftarrow \mathsf{SimSetup}(1^\kappa),\ b \leftarrow \mathcal{A}^{\mathsf{Sim}(\mathsf{crs},\tau,\cdot,\cdot,\cdot)}(\mathsf{crs})\ :\ b=1\right]\end{matrix}\right| \le \epsilon(\kappa),$$

where $\mathsf{Sim}(\mathsf{crs},\tau,x,w,m) := \mathsf{SimSign}(\mathsf{crs},\tau,x,m)$ and Sim *only responds if $(x,w) \in R$.*

Definition 15 (f-Extractability). *A* SoK *with respect to L_R is f-extractable, if in addition to $\mathcal{S}$ there exists a PPT extractor* Extract*, such that for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that it holds that*

$$\Pr\left[\begin{array}{l}(\mathsf{crs},\tau)\leftarrow\mathsf{SimSetup}(1^\kappa),\\ (x,m,\sigma)\leftarrow\mathcal{A}^{\mathsf{Sim}(\mathsf{crs},\tau,\cdot,\cdot,\cdot)}(\mathsf{crs}),\\ y\leftarrow\mathsf{Extract}(\mathsf{crs},\tau,x,m,\sigma)\end{array} : \begin{array}{r}\mathsf{Verify}(\mathsf{crs},x,m,\sigma)=0\ \vee\\ (x,m,\sigma)\in Q^{\mathsf{Sim}}\ \vee\\ (\exists\ w:(x,w)\in R\ \wedge\ y=f(w))\end{array}\right]\geq 1-\epsilon(\kappa),$$

where Q^{Sim} denotes the queries (resp. answers) of Sim.

We note that, as illustrated in [BCC+15], this notion is a strengthening of the original extractability notion from [CL06] which implies the original extractability notion if f is the identity [CL06]. In this case, we simply call the f-extractability property *extractability*. Analogous to [BCC+15], we require the used SoK to be at the same time extractable and *straight-line f-extractable* with respect to some f other than the identity, where straight-line says that the extractor runs without rewinding the adversary [Fis05].

Structure Preserving Signatures on Equivalence Classes. Subsequently, we briefly recall structure-preserving signatures on equivalence classes (SPS-EQ) as presented in [HS14, FHS14]. Therefore, let p be a prime and $\ell > 1$; then $\mathbb{Z}_p^\ell$ is a vector space and one can define a projective equivalence relation on it, which propagates to $\mathbb{G}_i^\ell$ and partitions $\mathbb{G}_i^\ell$ into equivalence classes. Let $\sim_\mathcal{R}$ be this relation, i.e., for $M, N \in \mathbb{G}_i^\ell : M \sim_\mathcal{R} N \Leftrightarrow \exists\, s \in \mathbb{Z}_p^* : M = sN$. An SPS-EQ scheme now signs an equivalence class $[M]_\mathcal{R}$ for $M \in (\mathbb{G}_i^*)^\ell$ by signing a representative M of $[M]_\mathcal{R}$. Let us recall the formal definition of an SPS-EQ scheme subsequently.

Definition 16. *An* SPS-EQ *on $\mathbb{G}_i^*$ (for $i \in \{1,2\}$) consists of the following PPT algorithms:*

$\mathsf{BGGen}_\mathcal{R}(1^\kappa)$: A bilinear-group generation algorithm, which on input of a security parameter κ outputs an asymmetric bilinear group BG.

$\mathsf{KeyGen}_\mathcal{R}(\mathsf{BG},\ell)$: An algorithm, which on input of an asymmetric bilinear group BG and a vector length $\ell > 1$ outputs a key pair $(\mathsf{sk},\mathsf{pk})$.

$\mathsf{Sign}_\mathcal{R}(M,\mathsf{sk})$: An algorithm, which given a representative $M \in (\mathbb{G}_i^*)^\ell$ and a secret key sk outputs a signature σ for the equivalence class $[M]_\mathcal{R}$.

$\mathsf{ChgRep}_\mathcal{R}(M,\sigma,\rho,\mathsf{pk})$: An algorithm, which on input of a representative $M \in (\mathbb{G}_i^*)^\ell$ of class $[M]_\mathcal{R}$, a signature σ for M, a scalar ρ and a public key pk returns an updated message-signature pair (M',σ'), where $M' = \rho \cdot M$ is the new representative and σ' its updated signature.

$\mathsf{Verify}_\mathcal{R}(M,\sigma,\mathsf{pk})$: An algorithm, which on input of a representative $M \in (\mathbb{G}_i^*)^\ell$, a signature σ and a public key pk outputs a bit $b \in \{0,1\}$.

$\mathsf{VKey}_\mathcal{R}(\mathsf{sk},\mathsf{pk})$ is an algorithm, which given a secret key sk and a public key pk outputs a bit $b \in \{0,1\}$.

For security, we require the following properties.

Definition 17 (Correctness). *An* SPS-EQ *scheme on $(\mathbb{G}_i^*)^\ell$ is called* correct *if for all security parameters $\kappa \in \mathbb{N}$, $\ell > 1$, $\mathsf{BG} \leftarrow \mathsf{BGGen}_\mathcal{R}(1^\kappa)$, $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}_\mathcal{R}(\mathsf{BG},\ell)$, $M \in (\mathbb{G}_i^*)^\ell$ and $\rho \in \mathbb{Z}_p^*$: $\mathsf{VKey}_\mathcal{R}(\mathsf{sk},\mathsf{pk}) = 1$ and $\Pr\left[\mathsf{Verify}_\mathcal{R}(M,\mathsf{Sign}_\mathcal{R}(M,\mathsf{sk}),\mathsf{pk}) = 1\right] = 1$ and $\Pr\left[\mathsf{Verify}_\mathcal{R}(\mathsf{ChgRep}_\mathcal{R}(M,\mathsf{Sign}_\mathcal{R}(M,\mathsf{sk}),\rho,\mathsf{pk}),\mathsf{pk}) = 1\right] = 1$.*

For EUF-CMA security, outputting a valid message-signature pair, corresponding to an unqueried equivalence class, is considered to be a forgery.

Definition 18 (EUF-CMA). *An* SPS-EQ *over* $(\mathbb{G}_i^*)^\ell$ *is* existentially unforgeable under adaptively chosen-message attacks*, if for all PPT adversaries* $\mathcal{A}$ *with access to a signing oracle* $\mathcal{O}$*, there is a negligible function* $\epsilon(\cdot)$ *such that:*

$$\Pr\left[\begin{array}{l}\mathsf{BG} \leftarrow \mathsf{BGGen}_{\mathcal{R}}(1^\kappa),\\ (\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}_{\mathcal{R}}(\mathsf{BG},\ell),\\ (M^*,\sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}(\cdot,\mathsf{sk})}(\mathsf{pk})\end{array} : \begin{array}{r}[M^*]_{\mathcal{R}} \neq [M]_{\mathcal{R}} \quad \forall M \in \mathcal{Q} \;\; \wedge\\ \mathsf{Verify}_{\mathcal{R}}(M^*,\sigma^*,\mathsf{pk}) = 1\end{array}\right] \leq \epsilon(\kappa),$$

where $\mathcal{Q}$ *is the set of queries that* $\mathcal{A}$ *has issued to the signing oracle* $\mathcal{O}$*.*

Besides EUF-CMA security, an additional security property for SPS-EQ was introduced in [FHS15].

Definition 19 (Perfect Adaption of Signatures). *An* SPS-EQ *scheme* $(\mathsf{BGGen}_{\mathcal{R}}, \mathsf{KeyGen}_{\mathcal{R}}, \mathsf{Sign}_{\mathcal{R}}, \mathsf{ChgRep}_{\mathcal{R}}, \mathsf{Verify}_{\mathcal{R}}, \mathsf{VKey}_{\mathcal{R}})$ *on* $(\mathbb{G}_i^*)^\ell$ perfectly adapts signatures *if for all tuples* $(\mathsf{sk}, \mathsf{pk}, M, \sigma, \rho)$ *where it holds that* $\mathsf{VKey}_{\mathcal{R}}(\mathsf{sk},\mathsf{pk}) = 1$*,* $\mathsf{Verify}_{\mathcal{R}}(M,\sigma,\mathsf{pk}) = 1$*,* $M \in (\mathbb{G}_i^*)^\ell$*, and* $\rho \in \mathbb{Z}_p^*$*, the distributions* $(\rho M, \mathsf{Sign}_{\mathcal{R}}(\rho M, \mathsf{sk}))$ *and* $\mathsf{ChgRep}_{\mathcal{R}}(M,\sigma,\rho,\mathsf{pk})$ *are identical.*

An instantiation providing all above security properties is provided in [FHS14, FHS15].

3 Dynamic Group Signatures

Subsequently, we recall the established model for dynamic group signatures by Bellare et al. [BSZ05] (BSZ model), where we slightly adapt the notation to ours. We assume that each algorithm outputs a special symbol $\perp$ on error.

$\mathsf{GKeyGen}(1^\kappa)$: This algorithm takes a security parameter κ as input and outputs a triple $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$ containing the group public key gpk, the issuing key ik as well as the opening key ok.

$\mathsf{UKeyGen}(1^\kappa)$: This algorithm takes a security parameter κ as input and outputs a user key pair $(\mathsf{usk}_i, \mathsf{upk}_i)$.

$\mathsf{Join}(\mathsf{gpk}, \mathsf{usk}_i, \mathsf{upk}_i)$: This algorithm takes the group public key gpk and the user's key pair $(\mathsf{usk}_i, \mathsf{upk}_i)$ as input. It interacts with the Issue algorithm and outputs the group signing key gsk_i of user i on success.

$\mathsf{Issue}(\mathsf{gpk}, \mathsf{ik}, i, \mathsf{upk}_i, \mathsf{reg})$: This algorithm takes the group public key gpk, the issuing key ik, the index i of a user, user i's public key upk_i, and the registration table reg as input. It interacts with the Join algorithm and adds an entry for user i in reg on success. In the end, it returns reg.

$\mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m)$: This algorithm takes the group public key gpk, a user's group signing key gsk_i, and a message m as input and outputs a group signature σ.

$\mathsf{Verify}(\mathsf{gpk}, m, \sigma)$: This algorithm takes the group public key gpk, a message m and a signature σ as input and outputs a bit $b \in \{0,1\}$.

$\mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \sigma)$: This algorithm takes the group public key gpk, the opening key ok, the registration table reg, a message m, and a valid signature σ on m under gpk as input. It extracts the identity of the signer and returns a pair (i, τ), where τ is a proof.

$\mathsf{Judge}(\mathsf{gpk}, m, \sigma, i, \mathsf{upk}_i, \tau)$: This algorithm takes the group public key gpk, a message m, a valid signature σ on m under gpk, an index i, user i's public key upk_i, and a proof τ. It returns a bit $b \in \{0,1\}$.

3.1 Oracles

In the following we recall the definitions of the oracles required by the security model. We assume that the keys $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$ created in the experiments are implicitly available to the oracles. Furthermore, the environment maintains the sets $\mathtt{HU}, \mathtt{CU}$ of honest and corrupted users, the set $\mathtt{GS}$ of message-signature tuples returned by the challenge oracle, the lists $\mathsf{upk}, \mathsf{usk}, \mathsf{gsk}$ of user public keys, user private keys, and group signing keys. The list upk is publicly readable and the environment also maintains the registration table reg. Finally, $\mathtt{SI}$ represents a variable that ensures the consistency of subsequent calls to CrptU and SndToI. All sets are initially empty and all list entries are initially set to $\perp$. In the context of lists, we use $\mathsf{upk}_i, \mathsf{usk}_i$, etc. as shorthand for $\mathsf{upk}[i], \mathsf{usk}[i]$, etc.

$\mathsf{AddU}(i)$: This oracle takes an index i as input. If $i \in \mathtt{CU} \cup \mathtt{HU}$ it returns $\perp$. Otherwise it runs $(\mathsf{usk}_i, \mathsf{upk}_i) \leftarrow \mathsf{UKeyGen}(1^\kappa)$ and

$$(\mathsf{reg}, \mathsf{gsk}_i) \leftarrow (\mathsf{Issue}(\mathsf{gpk}, \mathsf{ik}, i, \mathsf{upk}_i, \mathsf{reg}) \leftrightarrow \mathsf{Join}(\mathsf{gpk}, \mathsf{usk}_i, \mathsf{upk}_i)).$$

Finally, it sets $\mathtt{HU} \leftarrow \mathtt{HU} \cup \{i\}$ and returns upk_i.

$\mathsf{CrptU}(i, \mathsf{upk}_j)$: This oracle takes an index i and user public key upk_j as input. If $i \in \mathtt{CU} \cup \mathtt{HU}$ it returns $\perp$. Otherwise it sets $\mathtt{CU} \leftarrow \mathtt{CU} \cup \{i\}$, $\mathtt{SI} \leftarrow i$ and $\mathsf{upk}_i \leftarrow \mathsf{upk}_j$.

$\mathsf{SndToI}(i)$: This oracle takes an index i as input. If $i \neq \mathtt{SI}$ it returns $\perp$. Otherwise, it plays the role of an honest issuer when interacting with the corrupted user i. More precisely, it runs

$$\mathsf{reg} \leftarrow \mathsf{Issue}(\mathsf{gpk}, \mathsf{ik}, i, \mathsf{upk}_i, \mathsf{reg}),$$

thereby interacting with the dishonest user who aims to join the group but does not necessarily follow the Join protocol.

$\mathsf{SndToU}(i)$: This oracle takes an index i as input. If $i \notin \mathtt{HU}$ it sets $\mathtt{HU} \leftarrow \mathtt{HU} \cup \{i\}$, runs $(\mathsf{usk}_i, \mathsf{upk}_i) \leftarrow \mathsf{UKeyGen}(1^\kappa)$. Then it plays the role of the honest user i when interacting with a corrupted issuer. More precisely, it runs

$$\mathsf{gsk}_i \leftarrow \mathsf{Join}(\mathsf{gpk}, \mathsf{usk}_i, \mathsf{upk}_i),$$

thereby interacting with the dishonest issuer who does not necessarily follow the Issue protocol.

$\mathsf{USK}(i)$: This oracle takes an index i as input and returns $(\mathsf{gsk}_i, \mathsf{usk}_i)$.

$\mathsf{RReg}(i)$: This oracle takes an index i as input and returns reg_i.

$\mathsf{WReg}(i, \rho)$: This oracle takes an index i and a registration table entry ρ as input and sets $\mathsf{reg}_i \leftarrow \rho$.

$\mathsf{GSig}(i, m)$: This oracle takes an index i and a message m as input. If $i \notin \mathtt{HU}$ or $\mathsf{gsk}_i = \perp$ it returns $\perp$ and $\sigma \leftarrow \mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m)$ otherwise.

$\mathsf{Ch}(b, i_0, i_1, m)$: This algorithm takes a bit b, two indexes i_0 and i_1, and a message m as input. If $\{i_0, i_1\} \not\subseteq \mathtt{HU} \ \vee\ \mathsf{gsk}_{i_0} = \perp \ \vee\ \mathsf{gsk}_{i_1} = \perp$ it returns $\perp$. Otherwise, it computes $\sigma \leftarrow \mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_{i_b}, m)$, sets $\mathtt{GS} \leftarrow \mathtt{GS} \cup \{(m, \sigma)\}$ and returns σ.

$\mathsf{Open}(m, \sigma)$: This oracle takes a message m and a signature σ as input. If $(m, \sigma) \in \mathtt{GS}$ or $\mathsf{Verify}(\mathsf{gpk}, m, \sigma) = 0$ it returns $\perp$. Otherwise, it returns $(i, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \sigma)$.

3.2 Security Notions

For security, dynamic group signatures are required to be correct, anonymous, traceable, and non-frameable. We recall the formal definitions below, where, in addition to CCA2-full anonymity, we also include the weaker anonymity notion CPA-full anonymity from [BBS04].

Definition 20 (Correctness). *A* GSS *is correct, if for all adversaries* $\mathcal{A}$ *it holds that*

$$\Pr\left[\begin{array}{l}(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKeyGen}(1^\kappa),\\ \mathcal{O} \leftarrow \{\mathsf{AddU}(\cdot), \mathsf{RReg}(\cdot)\},\\ (i, m) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{gpk}),\\ \sigma \leftarrow \mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m),\\ (j, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \sigma)\end{array} : \begin{array}{r}\mathsf{Verify}(\mathsf{gpk}, m, \sigma) = 1 \ \wedge\\ i \in \mathtt{HU} \ \wedge\ \mathsf{gsk}_i \neq \perp \ \wedge\ i = j \ \wedge\\ \mathsf{Judge}(\mathsf{gpk}, m, \sigma, i, \mathsf{upk}_i, \tau) = 1\end{array}\right] = 1.$$

Definition 21 (T-Full Anonymity). *Let* $\mathsf{T} \in \{\mathsf{CPA}, \mathsf{CCA2}\}$. *A* GSS *is* T*-full anonymous, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\epsilon(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKeyGen}(1^\kappa),\ b \xleftarrow{R} \{0,1\}, \\ b^* \leftarrow \mathcal{A}^{\mathcal{O}_\mathsf{T}}(\mathsf{gpk}, \mathsf{ik})\end{array} \ :\ b = b^*\right] \leq 1/2 + \epsilon(\kappa),$$

where $\mathcal{O}_\mathsf{T} \leftarrow \{\mathsf{Ch}(b,\cdot,\cdot,\cdot),\ \mathsf{SndToU}(\cdot), \mathsf{WReg}(\cdot,\cdot),\ \mathsf{USK}(\cdot),\ \mathsf{CrptU}(\cdot,\cdot)\}$, *if* $\mathsf{T} = \mathsf{CPA}$ *and* $\mathcal{O}_\mathsf{T} \leftarrow \{\mathsf{Ch}(b,\cdot,\cdot,\cdot),\ \mathsf{Open}(\cdot,\cdot),\ \mathsf{SndToU}(\cdot), \mathsf{WReg}(\cdot,\cdot),\ \mathsf{USK}(\cdot),\ \mathsf{CrptU}(\cdot,\cdot)\}$ *if* $\mathsf{T} = \mathsf{CCA2}$.

Definition 22 (Traceability). *A* GSS *is traceable, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\epsilon(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKeyGen}(1^\kappa), \\ \mathcal{O} \leftarrow \{\mathsf{SndToI}(\cdot),\ \mathsf{AddU}(\cdot), \\ \mathsf{RReg}(\cdot),\ \mathsf{USK}(\cdot),\ \mathsf{CrptU}(\cdot)\}, \\ (m, \sigma) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{gpk}, \mathsf{ok}), \\ (i, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \sigma)\end{array} \ :\ \begin{array}{r}\mathsf{Verify}(\mathsf{gpk}, m, \sigma) = 1\ \wedge \\ (i = \perp\ \vee \\ \mathsf{Judge}(\mathsf{gpk}, m, \sigma, i, \mathsf{upk}_i, \tau) = 0)\end{array}\right] \leq \epsilon(\kappa).$$

Definition 23 (Non-Frameability). *A* GSS *is non-frameable, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\epsilon(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKeyGen}(1^\kappa), \\ \mathcal{O} \leftarrow \{\mathsf{SndToU}(\cdot),\ \mathsf{WReg}(\cdot,\cdot), \\ \mathsf{GSig}(\cdot,\cdot),\ \mathsf{USK}(\cdot),\ \mathsf{CrptU}(\cdot)\}, \\ (m, \sigma, i, \tau) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{gpk}, \mathsf{ok}, \mathsf{ik})\end{array} \ :\ \begin{array}{r}\mathsf{Verify}(\mathsf{gpk}, m, \sigma) = 1\ \wedge \\ i \in \mathtt{HU}\ \wedge\ \mathsf{gsk}_i \neq \perp\ \wedge \\ i \notin \mathtt{USK}\ \wedge\ (i, m) \notin \mathtt{SIG}\ \wedge \\ \mathsf{Judge}(\mathsf{gpk}, m, \sigma, i, \mathsf{upk}_i, \tau) = 1\end{array}\right] \leq \epsilon(\kappa),$$

where USK *and* SIG *denote the queries to the oracles* USK *and* Sign, *respectively.*

4 Construction

Before we present our construction in Scheme 1, we briefly revisit our basic idea. In our scheme, each group member chooses a secret vector $(R, P) \in (\mathbb{G}_1^*)^2$. When joining the group, a blinded version $q \cdot (R, P)$ with $q \xleftarrow{R} \mathbb{Z}_p^*$ of this vector is signed by the issuer using an SPS-EQ, and, by the re-randomization property of SPS-EQ, the user thus obtains a signature on the unblinded key (R, P) using $\mathsf{ChgRep}_\mathcal{R}$ with q^{-1}. To provide a means to open signatures, a user additionally has to provide an encryption of a value $\hat{R} \in \mathbb{G}_2$ such that $e(R, \hat{P}) = e(P, \hat{R})$ on joining (and signs the ciphertext to prove his identity). The group signing key of the user is then the pair consisting of the vector (R, P) and the SPS-EQ signature on this vector. A group member can sign a message m on behalf of the group by randomizing it's group signing key and computing a signature of knowledge (SoK) to the message m proving knowledge of the used randomizer.[1] The group signature is then the randomized group signing key and the SoK.

Very roughly, a signer then remains anonymous since it is infeasible to distinguish two randomized user secret keys under DDH in $\mathbb{G}_1$. The unforgeability of SPS-EQ ensures that each valid signature can be opened. Furthermore, it is hard to forge signatures of honest group members since it is hard to unblind a user secret key under co-CDHI and the signature of knowledge essentially ensures that we can extract such an unblinded user secret key from a successful adversary.

In our scheme, we require zero-knowledge proofs upon Join and Open. The **NP** relation corresponding to the proof carried out in Join is defined as

$$((U_i, Q, \hat{C}_{\mathsf{J}_i}, \mathsf{pk}_\mathsf{O}), (r, \omega)) \in R_\mathsf{J} \iff \hat{C}_{\mathsf{J}_i} = \Omega.\mathsf{Enc}(\mathsf{pk}_\mathsf{O}, r\hat{P};\ \omega)\ \wedge\ U_i = r \cdot Q.$$

[1] For technical reasons and in particular extractability, we actually require a signature of knowledge for message $m' = \sigma_1 || m$, where σ_1 contains the re-randomized user secret key and SPS-EQ signature.

The **NP** relation corresponding to the proof carried out upon Open is defined as

$$((\hat{C}_{\mathsf{J}_i}, \mathsf{pk}_{\mathsf{O}}, \mathsf{upk}_i, \sigma), (\mathsf{sk}_{\mathsf{O}}, \hat{R})) \in R_{\mathsf{O}} \iff \hat{R} = \Omega.\mathsf{Dec}(\mathsf{sk}_{\mathsf{O}}, \hat{C}_{\mathsf{J}_i}) \ \wedge \ \mathsf{pk}_{\mathsf{O}} \equiv \mathsf{sk}_{\mathsf{O}} \ \wedge \ e(\sigma_1[1][1], \hat{P}) = e(\sigma_1[1][2], \hat{R}).$$

Thereby, $\mathsf{pk} \equiv \mathsf{sk}$ denotes the consistence of pk and sk. Furthermore, upon Sign we require a signatures of knowledge which is with respect to the following **NP** relation.

$$((P, Q), \rho) \in R_{\mathsf{S}} \iff Q = \rho \cdot P \ .$$

For the sake of compact presentation, we assume that the languages defined by R_{J}, R_{O}, R_{S} are implicit in the CRSs $\mathsf{crs}_{\mathsf{J}}$, $\mathsf{crs}_{\mathsf{O}}$, and $\mathsf{crs}_{\mathsf{S}}$, respectively. Correctness of Scheme 1 is straight

$\mathsf{GKeyGen}(1^\kappa)$: Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa)$, $(\mathsf{sk}_{\mathcal{R}}, \mathsf{pk}_{\mathcal{R}}) \leftarrow \mathsf{KeyGen}_{\mathcal{R}}(\mathsf{BG}, 2)$, $(\mathsf{sk}_{\mathsf{O}}, \mathsf{pk}_{\mathsf{O}}) \leftarrow \Omega.\mathsf{KeyGen}(1^\kappa)$, $\mathsf{crs}_{\mathsf{J}} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$, $\mathsf{crs}_{\mathsf{O}} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$, $\mathsf{crs}_{\mathsf{S}} \leftarrow \mathsf{SoK}.\mathsf{Setup}(1^\kappa)$, set $\mathsf{gpk} \leftarrow (\mathsf{pk}_{\mathcal{R}}, \mathsf{pk}_{\mathsf{O}}, \mathsf{crs}_{\mathsf{J}}, \mathsf{crs}_{\mathsf{O}}, \mathsf{crs}_{\mathsf{S}})$, $\mathsf{ik} \leftarrow \mathsf{sk}_{\mathcal{R}}$, $\mathsf{ok} \leftarrow \mathsf{sk}_{\mathsf{O}}$ and return $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$.

$\mathsf{UKeyGen}(1^\kappa)$: Return $(\mathsf{usk}_i, \mathsf{upk}_i) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$.

$\mathsf{Join}^{(1)}(\mathsf{gpk}, \mathsf{usk}_i, \mathsf{upk}_i)$: Choose $q, r \xleftarrow{R} \mathbb{Z}_p^*$, set $(U_i, Q) \leftarrow (r \cdot qP, qP)$, and output $M_{\mathsf{J}} \leftarrow ((U_i, Q), \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i}, \pi_{\mathsf{J}_i})$ and $\mathsf{st} \leftarrow (\mathsf{gpk}, \mathsf{usk}_i, \mathsf{upk}_i, q, U_i, Q)$, where

$$\hat{C}_{\mathsf{J}_i} \leftarrow \Omega.\mathsf{Enc}(\mathsf{pk}_{\mathsf{O}}, r\hat{P};\ \omega),\ \ \sigma_{\mathsf{J}_i} \leftarrow \Sigma.\mathsf{Sign}(\mathsf{usk}_i, \hat{C}_{\mathsf{J}_i}),$$
$$\pi_{\mathsf{J}_i} \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}_{\mathsf{J}}, (U_i, Q, \hat{C}_{\mathsf{J}_i}, \mathsf{pk}_{\mathsf{O}}), (r, \omega)).$$

$\mathsf{Issue}(\mathsf{gpk}, \mathsf{ik}, i, \mathsf{upk}_i, \mathsf{reg})$: Receive $M_{\mathsf{J}} = ((U_i, Q), \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i}, \pi_{\mathsf{J}_i})$, return reg and send σ' to user i, where

$$\mathsf{reg} \leftarrow \mathsf{reg} \cup \{(i, \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i})\},\ \ \sigma' \leftarrow \mathsf{Sign}_{\mathcal{R}}((U_i, Q), \mathsf{sk}_{\mathcal{R}}),$$

if $\Pi.\mathsf{Verify}(\mathsf{crs}_{\mathsf{J}}, (U_i, Q, \hat{C}_{\mathsf{J}_i}, \mathsf{pk}_{\mathsf{O}}), \pi_{\mathsf{J}_i}) = 1 \ \wedge\ \Sigma.\mathsf{Verify}(\mathsf{upk}_i, \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i}) = 1$, and return $\perp$ otherwise.

$\mathsf{Join}^{(2)}(\mathsf{st}, \sigma')$: Parse st as $(\mathsf{gpk}, \mathsf{usk}_i, \mathsf{upk}_i, q, U_i, Q)$ and return gsk_i, where

$$\mathsf{gsk}_i = ((rP, P), \sigma) \leftarrow \mathsf{ChgRep}_{\mathcal{R}}((U_i, Q), \sigma', q^{-1}, \mathsf{pk}_{\mathcal{R}}),$$

if $\mathsf{Verify}_{\mathcal{R}}((U_i, Q), \sigma', \mathsf{pk}_{\mathcal{R}}) = 1$, and return $\perp$ otherwise.

$\mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m)$: Choose $\rho \xleftarrow{R} \mathbb{Z}_p^*$, and return $\sigma \leftarrow (\sigma_1, \sigma_2)$, where

$$\sigma_1 \leftarrow \mathsf{ChgRep}_{\mathcal{R}}(\mathsf{gsk}_i, \rho, \mathsf{pk}_{\mathcal{R}}),\ \ \sigma_2 \leftarrow \mathsf{SoK}.\mathsf{Sign}(\mathsf{crs}_{\mathsf{S}}, (P, \sigma_1[1][2]), \rho, \sigma_1||m).$$

$\mathsf{Verify}(\mathsf{gpk}, m, \sigma)$: Return 1 if the following holds, and 0 otherwise:

$$\mathsf{Verify}_{\mathcal{R}}(\sigma_1, \mathsf{pk}_{\mathcal{R}}) = 1 \quad \wedge \quad \mathsf{SoK}.\mathsf{Verify}(\mathsf{crs}_{\mathsf{S}}, (P, \sigma_1[1][2]), \sigma_1||m, \sigma_2) = 1.$$

$\mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \sigma)$: Obtain $(i, \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i})$ and $\hat{R} \leftarrow \Omega.\mathsf{Dec}(\mathsf{sk}_{\mathsf{O}}, \hat{C}_{\mathsf{J}_i})$ from reg such that $e(\sigma_1[1][1], \hat{P}) = e(\sigma_1[1][2], \hat{R})$. Return $\tau \leftarrow (\pi_{\mathsf{O}}, \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i})$ and $\perp$ if no such upk_i exists, where

$$\pi_{\mathsf{O}} \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}_{\mathsf{O}}, (\hat{C}_{\mathsf{J}_i}, \mathsf{pk}_{\mathsf{O}}, \mathsf{upk}_i, \sigma), (\mathsf{sk}_{\mathsf{O}}, \hat{R}))$$

$\mathsf{Judge}(\mathsf{gpk}, m, \sigma, i, \mathsf{upk}_i, \tau)$: Parse τ as $(\pi_{\mathsf{O}}, \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i})$, and return 1 if the following holds and 0 otherwise:

$$\Sigma.\mathsf{Verify}(\mathsf{upk}_i, \hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i}) = 1 \quad \wedge \quad \Pi.\mathsf{Verify}(\mathsf{crs}_{\mathsf{O}}, (\hat{C}_{\mathsf{J}_i}, \mathsf{pk}_{\mathsf{O}}, \mathsf{upk}_i, \sigma), \pi_{\mathsf{O}}) = 1.$$

Scheme 1: Fully Secure Dynamic Group Signature Scheme

forward and can be verified by inspection. The remaining properties are proven subsequently.

Theorem 1. *If* $\mathsf{\Pi}$ *is adaptively zero knowledge,* SoK *is simulatable,* Ω *is* $\mathsf{IND\text{-}CPA}$ *secure,* $\mathsf{SPS\text{-}EQ}$ *perfectly adapts signatures, and the DDH assumption holds in* $\mathbb{G}_1$*, then Scheme 1 is* CPA*-full anonymous.*

Theorem 2. *If* $\mathsf{\Pi}$ *is adaptively zero knowledge,* SoK *is simulatable and straight-line* f*-extractable, where* $f : \mathbb{Z}_p \to \mathbb{G}_2$ *is defined as* $r \mapsto r \cdot \hat{P}$*,* Ω *is* $\mathsf{IND\text{-}CCA2}$ *secure,* $\mathsf{SPS\text{-}EQ}$ *perfectly adapts signatures, and the DDH assumption holds in* $\mathbb{G}_1$*, then Scheme 1 is* $\mathsf{CCA2}$*-full anonymous.*

Proof (Anonymity). We prove Theorem 1 and 2 by showing that the output distributions of the Ch oracle are (computationally) independent of the bit b, where we highlight the parts of the proof which are specific to Theorem 2 and can be omitted to prove Theorem 1. Therefore, let $q_{\mathsf{Ch}} \leq \mathsf{poly}(\kappa)$ be the number of queries to Ch, $q_{\mathsf{O}} \leq \mathsf{poly}(\kappa)$ be the number of queries to Open, and $q_{\mathsf{SndToU}} \leq \mathsf{poly}(\kappa)$ be the number of queries to SndToU.

Game 0: The original anonymity game.

Game 1: As Game 0, but we run $(\mathsf{crs_J}, \tau_{\mathsf{J}}) \leftarrow \mathsf{\Pi}.S_1(1^\kappa)$ instead of $\mathsf{crs_J} \leftarrow \mathsf{\Pi.Setup}(1^\kappa)$ upon running $\mathsf{GKeyGen}$ and store the trapdoor τ_{J}. Then, we simulate all calls to $\mathsf{\Pi.Proof}$ executed in Join using the simulator, i.e., without a witness.

Transition - Game 0 → Game 1: A distinguisher between Game 0 and Game 1 is an adversary against adaptive zero knowledge of $\mathsf{\Pi}$, and, therefore, the probability to distinguish Game 0 and Game 1 is negligible, i.e., $|\Pr[S_1] - \Pr[S_0]| \leq \epsilon_{\mathsf{ZK_J}}(\kappa)$.

Game 2: As Game 1, but we run $(\mathsf{crs_O}, \tau_{\mathsf{O}}) \leftarrow \mathsf{\Pi}.S_1(1^\kappa)$ instead of $\mathsf{crs_O} \leftarrow \mathsf{\Pi.Setup}(1^\kappa)$ upon running $\mathsf{GKeyGen}$ and store the trapdoor τ_{O}. Then, we simulate all calls to $\mathsf{\Pi.Proof}$ in Open using the simulator, i.e., without a witness.

Transition - Game 1 → Game 2: A distinguisher between Game 0 and Game 1 is an adversary against adaptive zero knowledge of $\mathsf{\Pi}$, and, therefore, the probability to distinguish Game 1 and Game 2 is negligible, i.e., $|\Pr[S_2] - \Pr[S_1]| \leq \epsilon_{\mathsf{ZK_O}}(\kappa)$.

Game 3: As Game 2, but we run $(\mathsf{crs_S}, \tau_{\mathsf{S}}) \leftarrow \mathsf{SoK.SimSetup}(1^\kappa)$ instead of $\mathsf{crs_S} \leftarrow \mathsf{SoK.Setup}(1^\kappa)$ upon running $\mathsf{GKeyGen}$ and store the trapdoor τ_{S}.

Transition - Game 2 → Game 3: A distinguisher between Game 2 and Game 3 is an adversary against simulatability of SoK. Therefore, the distinguishing probability is negligible, i.e., $|\Pr[S_3] - \Pr[S_2]| \leq \epsilon_{\mathsf{SIM}}(\kappa)$.

Game 4: As Game 3, but instead of computing $(\mathsf{sk_O}, \mathsf{pk_O}) \leftarrow \Omega.\mathsf{KeyGen}(1^\kappa)$ in $\mathsf{GKeyGen}$, we obtain $\mathsf{pk_O}$ from an $\mathsf{IND\text{-}CPA}$ (resp. $\mathsf{IND\text{-}CCA2}$) challenger and set $\mathsf{sk_O} \leftarrow \perp$.

> In the $\mathsf{CCA2}$ case, the environment additionally maintains a secret list $\mathtt{GSK}$ and upon each call to the SndToU oracle it sets $\mathtt{GSK}[i] \leftarrow \mathsf{gsk}_i$. Furthermore, we simulate the Open algorithm executed within the Open oracle as follows.
>
> $\mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathsf{reg}, m, \sigma)$: Obtain $\hat{R}$ using the straight-line f-extractor, and obtain index i such that $e(\mathtt{GSK}[i][1][1], \hat{R}) = e(\sigma_1[1][1], \hat{P})$. If $i \neq \perp$ and the entry for i in the registration table was not overwritten by the adversary, compute a simulated proof τ and return (i, τ). Otherwise we submit $\hat{C}_{\mathsf{J}_i}$ to the decryption oracle provided by the $\mathsf{IND\text{-}CCA2}$ challenger and return whatever the original open oracle would return (but with a simulated proof).
>
> Note that since we never need to simulate SoKs in the anonymity game, we know that the conditions for the straight-line f-extractor to work with overwhelming probability are satisfied for every query. If the extractor fails at some point, we choose $b \xleftarrow{R} \{0, 1\}$ and return b.

Transition - Game 3 → Game 4 (CPA*):* This change is only conceptual, i.e., $\Pr[S_3] = \Pr[S_4]$.

*Transition - Game 3 → Game 4 (*CCA2*):* By the straight-line f-extractability of the SoK, one can extract a witness ρ in every call to Open with overwhelming probability $1 - \epsilon_{\mathsf{EXT}}(\kappa)$. Thus, we have that $\Pr[S_4] = 1/2 + (\Pr[S_3] - 1/2) \cdot (1 - \epsilon_{\mathsf{EXT}}(\kappa))^{q_{\mathsf{O}}}$.

Game 5: As Game 4, but we compute the ciphertext $\hat{C}_{\mathsf{J}_i}$ in the Join algorithm (executed within the SndToU oracle) as $\hat{C}_{\mathsf{J}_i} \leftarrow \Omega.\mathsf{Enc}(\mathsf{pk}, \hat{P})$, i.e., with a constant message that is independent of the user.

Transition - Game 4 → Game 5: A distinguisher between Game 4 and Game 5 is a distinguisher for the IND-CPA (resp. IND-CCA2) game of Ω, i.e., $|\Pr[S_5] - \Pr[S_4]| \leq q_{\mathsf{SndToU}} \cdot \epsilon_{\mathsf{CPA}}(\kappa)$ (resp. $|\Pr[S_5] - \Pr[S_4]| \leq q_{\mathsf{SndToU}} \cdot \epsilon_{\mathsf{CCA2}}(\kappa)$).[2]

Game 6: As Game 5, but the environment obtains and stores a DDH instance (aP, bP, cP) in $\mathbb{G}_1$. Additionally, we further modify the Join algorithm (executed within SndToU) as follows. Instead of choosing $r \xleftarrow{R} \mathbb{Z}_p$, we use the random self reducibility of DDH to obtain an independent DDH instance $(uP, vP, wP) \overset{RSR}{\leftarrow} (aP, bP, cP)$, choose $q \xleftarrow{R} \mathbb{Z}_p^*$, compute $(U_i, Q_i) \leftarrow (q \cdot uP, q \cdot P)$. Furthermore, the environment maintains a secret list $\mathtt{DDH}$ and upon each Issue it sets $\mathtt{DDH}[i] \leftarrow (uP, vP, wP)$.

Transition - Game 5 → Game 6: The output distributions in Game 5 and Game 6 are identical, i.e., $\Pr[S_6] = \Pr[S_5]$.

Game 7: As Game 6, but all calls to $\mathsf{ChgRep}_{\mathcal{R}}(M, \rho, \mathsf{pk}_{\mathcal{R}})$ are replaced by $\mathsf{Sign}_{\mathcal{R}}(\rho \cdot M, \mathsf{sk}_{\mathcal{R}})$.

Transition - Game 6 → Game 7: Under perfect adaption of signatures, the output distributions in Game 6 and Game 7 are identical, i.e., $\Pr[S_7] = \Pr[S_6]$.

Game 8_j ($1 \leq j \leq q_{\mathsf{Ch}}$): As Game 7, but we modify the Ch oracle as follows. For the first j queries, instead of running $\sigma_1 \leftarrow \mathsf{Sign}_{\mathcal{R}}(\rho \cdot \mathsf{gsk}_{i_b}[1], \mathsf{sk}_{\mathcal{R}})$, we choose $R \xleftarrow{R} \mathbb{G}_1$, and compute $\sigma_1 \leftarrow \mathsf{Sign}_{\mathcal{R}}(\rho \cdot (R, P), \mathsf{sk}_{\mathcal{R}})$.

Transition - Game 7 → Game 8_1: A distinguisher between Game 7 and Game 8_1 is a DDH distinguisher. To show this, we present an implementation of the Ch oracle, that—depending on the validity of the DDH instance (aP, bP, cP)—interpolates between Game 7 and Game 8_1. That is, in the first query we obtain the tuple $(uP, vP, wP) \leftarrow \mathtt{DDH}[i_b]$ and compute σ_1 as $\sigma_1 \leftarrow \mathsf{Sign}_{\mathcal{R}}((wP, vP), \mathsf{sk}_{\mathcal{R}})$. Then, if the initial DDH instance (aP, bP, cP) is valid, we have a distribution as in Game 7, whereas we have a distribution as in Game 8_1 otherwise. The success probability of a distinguisher between Game 7 and Game 8_1 is thus negligible, i.e., $|\Pr[S_{8_1}] - \Pr[S_7]| \leq \epsilon_{\mathsf{DDH}}(\kappa)$.

Transitions - Game 8_j → Game 8_{j+1} ($1 \leq j \leq q_{\mathsf{Ch}}$): The answers of the Ch oracle for the first j queries are already random in Game 8_j. Then, it is easy to show that a distinguisher for Game 8_j and game 8_{j+1} is a DDH distinguisher, i.e., by embedding $(uP, vP, wP) \leftarrow \mathtt{DDH}[i_b]$ in the answer of the Ch query $j+1$ using the same strategy as above. Summing up, we have $|\Pr[S_{8_q}] - \Pr[S_{8_1}]| \leq (q_{\mathsf{Ch}} - 1) \cdot \epsilon_{\mathsf{DDH}}(\kappa)$.

In Game $8_{q_{\mathsf{Ch}}}$, the simulation is independent of the bit b, i.e., $\Pr[S_{8_q}] = 1/2$; what remains is to obtain a bound on the success probability in Game 0. In the CPA case, we have that $\Pr[S_0] \leq 1/2 + q_{\mathsf{SndToU}} \cdot \epsilon_{\mathsf{CPA}}(\kappa) + q_{\mathsf{Ch}} \cdot \epsilon_{\mathsf{DDH}}(\kappa) + \epsilon_{\mathsf{ZK_J}}(\kappa) + \epsilon_{\mathsf{ZK_O}}(\kappa) + \epsilon_{\mathsf{SIM}}(\kappa)$, which proves Theorem 1. In the CCA2 case, we have that $\Pr[S_0] \leq \Pr[S_3] + \epsilon_{\mathsf{ZK_J}}(\kappa) + \epsilon_{\mathsf{ZK_O}} + \epsilon_{\mathsf{SIM}}(\kappa)$, that $\Pr[S_4] \leq 1/2 + q_{\mathsf{SndToU}} \cdot \epsilon_{\mathsf{CCA2}}(\kappa) + q_{\mathsf{Ch}} \cdot \epsilon_{\mathsf{DDH}}(\kappa)$, and that $\frac{\Pr[S_4] - 1/2}{(1-\epsilon_{\mathsf{EXT}}(\kappa))^{q_{\mathsf{O}}}} + 1/2 = \Pr[S_3]$. This yields $\Pr[S_0] \leq \frac{\Pr[S_4] - 1/2}{(1-\epsilon_{\mathsf{EXT}}(\kappa))^{q_{\mathsf{O}}}} + 1/2 + \epsilon_{\mathsf{ZK_J}}(\kappa) + \epsilon_{\mathsf{ZK_O}} + \epsilon_{\mathsf{SIM}}(\kappa) = \frac{q_{\mathsf{SndToU}} \cdot \epsilon_{\mathsf{CCA2}}(\kappa) + q_{\mathsf{Ch}} \cdot \epsilon_{\mathsf{DDH}}(\kappa)}{(1-\epsilon_{\mathsf{EXT}}(\kappa))^{q_{\mathsf{O}}}} + 1/2 + \epsilon_{\mathsf{ZK_J}}(\kappa) + \epsilon_{\mathsf{ZK_O}}(\kappa) + \epsilon_{\mathsf{SIM}}(\kappa)$, which proves Theorem 2. □

We note that one can avoid the factor q_{Ch} in the proof using the fact that one can obtain q_{Ch} DDH instances $(uP, v_iP, w_iP)_{i \in [q_{\mathsf{Ch}}]}$ from a single DDH instance (uP, vP, wP) so that (in-)validity of the original instance carries over to each $(uP, v_iP, w_iP)_{i \in [q_{\mathsf{Ch}}]}$. To do so, one chooses $x_j, y_j \xleftarrow{R} \mathbb{Z}_p$ and computes $v_iP \leftarrow x_j \cdot vP + y_jP$ and $w_jP \leftarrow x_j \cdot wP + y_j \cdot uP$.

[2] For compactness, we collapsed the q_{SndToU} game changes into a single game change and note that one can straight forwardly unroll this to q_{SndToU} game changes where a single ciphertext is exchanged in each game.

Theorem 3. *If* $\mathsf{SPS\text{-}EQ}$ *is* $\mathsf{EUF\text{-}CMA}$ *secure, and* Π *is sound, then Scheme 1 is traceable.*

Proof (Traceability). We show that traceability holds using a sequence of games, where we let $q \leq \mathsf{poly}(\kappa)$ be the number of queries to the SndToI oracle.

Game 0: The original traceability game.
Game 1: As the original game, but we obtain $\mathsf{crs_J}$ from a soundness challenger of Π.
Transition - Game 0 → Game 1: This change is only conceptual, i.e., $\Pr[S_0] = \Pr[S_1]$.
Game 2: As Game 1, but after every successful execution of SndToI, we obtain $\hat{R} \leftarrow \Omega.\mathsf{Dec}(\mathsf{sk_O}, C_{\mathsf{J}_i})$ and abort if $e(U_i, \hat{P}) \neq e(Q, \hat{R})$.
Transition - Game 0 → Game 1: If we abort we have a valid proof π_{J_i} attesting that $(U_i, Q, \hat{C}_{\mathsf{J}_i}, \mathsf{pk_O}) \in L_{\mathsf{R_J}}$, but by the perfect correctness of Ω there exists no ω such that $C_{\mathsf{J}_i} = \Omega.\mathsf{Enc}(\mathsf{pk_O}, r \cdot \hat{P};\ \omega)\ \wedge\ U_i = r \cdot Q$, i.e., we have that $(U_i, Q, \hat{C}_{\mathsf{J}_i}, \mathsf{pk_O})$ is actually not in L_{R_J}. Thus, we only abort if the adversary breaks the soundness of Π in one of the oracle queries, i.e., $\Pr[S_2] = \Pr[S_1] \cdot (1 - \epsilon_{\mathsf{S}}(\kappa))^q$.
Game 3: As Game 2, but we obtain BG and a public key $\mathsf{pk}_{\mathcal{R}}$ from an $\mathsf{EUF\text{-}CMA}$ challenger of the $\mathsf{SPS\text{-}EQ}$. Whenever an $\mathsf{SPS\text{-}EQ}$ signature is required, $\mathcal{R}^{\mathsf{f}}$ forwards the message to be signed to the signing oracle provided by the $\mathsf{EUF\text{-}CMA}$ challenger.
Transition - Game 2 → Game 3: This change is only conceptual, i.e., $\Pr[S_2] = \Pr[S_3]$.

If the adversary eventually outputs a valid forgery (m, σ), we know that σ contains an $\mathsf{SPS\text{-}EQ}$ signature σ_1 for some (rP, P) such that we have never seen a corresponding $r\hat{P}$, i.e., there is no entry i in the registration table where $\hat{C}_{\mathsf{J}_i}$ contains $r\hat{P}$ s.t. $e(\sigma_1[1][1], \hat{P}) = e(\sigma_1[1][2], r\hat{P})$ holds. Consequently, σ_1 is a valid $\mathsf{SPS\text{-}EQ}$ signature for an unqueried equivalence class and we have that $\Pr[S_3] \leq \epsilon_{\mathsf{F}}(\kappa)$. All in all, this yields that $\Pr[S_0] \leq \frac{\epsilon_{\mathsf{F}}(\kappa)}{(1-\epsilon_{\mathsf{S}}(\kappa))^q}$, which proves the theorem. □

Theorem 4. *If* Π *is sound and adaptively zero knowledge,* SoK *is simulatable and extractable,* Σ *is* $\mathsf{EUF\text{-}CMA}$ *secure,* Ω *is perfectly correct, and the co-CDHI assumption holds, then Scheme 1 is non-frameable.*

Proof (Non-frameability). We prove non-frameability using a sequence of games. Thereby we let the number of users in the system be $q \leq \mathsf{poly}(\kappa)$.

Game 0: The original non-frameability game.
Game 1: As Game 0, but we guess the index i that will be attacked by the adversary. If the adversary attacks another index, we abort.
Transition - Game 0 → Game 1: The winning probability in Game 1 is the same as in Game 0, unless an abort event happens. We thus have $\Pr[S_1] = \Pr[S_0] \cdot {}^1/_q$.
Game 2: As Game 1, but we run $(\mathsf{crs_J}, \tau_\mathsf{J}) \leftarrow \Pi.S_1(1^\kappa)$ instead of $\mathsf{crs_J} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$ upon running $\mathsf{GKeyGen}$ and store the trapdoor τ_J. Then, we simulate all calls to $\Pi.\mathsf{Proof}$ in Join using the simulator, i.e., without a witness.
Transition - Game 1 → Game 2: A distinguisher between Game 1 and Game 2 is an adversary against adaptive zero knowledge of Π, and, therefore, the probability to distinguish Game 1 and Game 2 is negligible, i.e., $|\Pr[S_2] - \Pr[S_1]| \leq \epsilon_{\mathsf{ZK_J}}(\kappa)$.
Game 3: As Game 2, but we obtain $\mathsf{crs_O}$ from a soundness challenger upon running $\mathsf{GKeyGen}$.
Transition - Game 2 → Game 3: This change is only conceptual, i.e., $\Pr[S_3] = \Pr[S_2]$.
Game 4: As Game 3, but we setup the SoK in simulation mode, i.e., we run $(\mathsf{crs_S}, \tau_\mathsf{S}) \leftarrow \mathsf{SoK.SimSetup}(1^\kappa)$ instead of $\mathsf{crs_S} \leftarrow \mathsf{SoK.Setup}(1^\kappa)$ upon running $\mathsf{GKeyGen}$ and store the trapdoor τ_S. Then, we simulate all calls to $\mathsf{SoK.Sign}$ using the simulator, i.e., without a witness.
Transition - Game 3 → Game 4: A distinguisher between Game 3 and Game 4 is an adversary against simulatability of SoK. Therefore, the distinguishing probability is negligible, i.e., $|\Pr[S_4] - \Pr[S_3]| \leq \epsilon_{\mathsf{SIM}}(\kappa)$.

Game 5: As Game 4, but we modify the Join algorithm (executed within the SndToU oracle) when queried for user with index i as follows. We obtain a co-CDHI instance $(aP, {}^{1}/_{a}\hat{P})$ for BG, choose $r \xleftarrow{R} \mathbb{Z}_p$, set $(U_i, Q) \leftarrow (r \cdot P, aP)$, and compute $\hat{C}_{\mathsf{J}_i} \leftarrow \Omega.\mathsf{Enc}(\mathsf{pk}_{\mathsf{O}}, r \cdot {}^{1}/_{a}\hat{P})$ and store r. On successful execution we set $\mathsf{gsk}_i \leftarrow ((U_i, Q), \sigma')$ (note that π_{J_i} as well as the signatures in the GSig oracle are already simulated, i.e., the discrete log of Q is not required to be known to the environment).

Transition - Game 4 → Game 5: Since r is uniformly random, we can write it as $r = r'a$ for some $r' \in \mathbb{Z}_p$. Then it is easy to see that the game change is only conceptual, i.e., $\Pr[S_5] = \Pr[S_4]$.

Game 6: As Game 5, but for every forgery output by the adversary, we extract $\rho \leftarrow \mathsf{SoK.Extract}(\mathsf{crs}_{\mathsf{S}}, \tau_{\mathsf{S}}, (P, \sigma_1[1][2]), \sigma_1||m, \sigma_2)$ and abort if the extraction fails.

Transition - Game 5 → Game 6: By the extractability of the SoK, one can extract a witness ρ with overwhelming probability $1 - \epsilon_{\mathsf{EXT}}(\kappa)$.[3] Thus, we abort with probability $\epsilon_{\mathsf{EXT}}(\kappa)$ and $\Pr[S_6] = \Pr[S_5] \cdot (1 - \epsilon_{\mathsf{EXT}}(\kappa))$.

Game 7: As Game 6, but we further modify the Join algorithm when queried for user with index i (executed within the SndToU oracle) as follows. Instead of choosing $(\mathsf{usk}_i, \mathsf{upk}_i) \leftarrow \mathsf{UKeyGen}(1^\kappa)$, we engage with an $\mathsf{EUF\text{-}CMA}$ challenger, obtain upk_i and set $\mathsf{usk}_i \leftarrow \emptyset$. If any signature is required, we obtain it using the oracle provided by the $\mathsf{EUF\text{-}CMA}$ challenger.

Transition Game 6 → Game 7: This change is only conceptual, i.e., $\Pr[S_7] = \Pr[S_6]$.

At this point we have three possibilities if $\mathcal{A}$ outputs a valid forgery.

1. If a signature for $\hat{C}_{\mathsf{J}_i}$ was never requested, $\mathcal{A}$ is an $\mathsf{EUF\text{-}CMA}$ forger for Σ and the forgery is $(\hat{C}_{\mathsf{J}_i}, \sigma_{\mathsf{J}_i})$. The probability for this to happen is upper bounded by $\epsilon_{\mathsf{f}}(\kappa)$.
2. Otherwise, we know that $\hat{C}_{\mathsf{J}_i}$ is honestly computed by the environment and—by the perfect correctness of Ω—thus contains ${}^{r}/_{a}\hat{P}$, which leaves us two possibilities:
   (a) If $e(\sigma[1][1], \hat{P}) = e(\sigma[1][2], {}^{r}/_{a}\hat{P})$, $\mathcal{A}$ is an adversary against co-CDHI, since we can obtain $(((r \cdot {}^{1}/_{a}P, P), \sigma')) \leftarrow \mathsf{ChgRep}_{\mathcal{R}}(\sigma_1, \rho^{-1}, \mathsf{pk}_{\mathcal{R}})$ and use r to output $r^{-1} \cdot (r \cdot {}^{1}/_{a}P) = {}^{1}/_{a}P$. The probability for this to happen is upper bounded by $\epsilon_{\mathsf{co\text{-}CDHI}}(\kappa)$.
   (b) If $e(\sigma[1][1], \hat{P}) \neq e(\sigma[1][2], {}^{r}/_{a}\hat{P})$, $\mathcal{A}$ has produced an opening proof for a statement which is actually not in $L_{R_{\mathsf{O}}}$. The probability for this to happen is upper bounded by $\epsilon_{\mathsf{S}}(\kappa)$.

Taking the union bound we obtain $\epsilon_{\mathsf{nf7}}(\kappa) \leq \epsilon_{\mathsf{f}}(\kappa) + \epsilon_{\mathsf{co\text{-}CDHI}}(\kappa) + \epsilon_{\mathsf{S}}(\kappa)$, which yields the following bound for the success probability in Game 1: $\Pr[S_1] \leq \Pr[S_5] + \epsilon_{\mathsf{ZK_J}}(\kappa) + \epsilon_{\mathsf{SIM}}(\kappa)$. Furthermore, we know that $\Pr[S_5] = \frac{\epsilon_{\mathsf{nf7}}(\kappa)}{1-\epsilon_{\mathsf{EXT}}(\kappa)}$ and $\Pr[S_0] = \Pr[S_1] \cdot q$. Taking all together we have that $\Pr[S_0] \leq q \cdot (\frac{\epsilon_{\mathsf{nf7}}(\kappa)}{1-\epsilon_{\mathsf{EXT}}(\kappa)} + \epsilon_{\mathsf{ZK_J}}(\kappa) + \epsilon_{\mathsf{SIM}}(\kappa))$, which is negligible. □

5 Instantiation in the ROM

To compare our approach to existing approaches regarding signature size and computational effort upon signature generation and verification, we present the sign and verification algorithms for an instantiation of our scheme with the $\mathsf{SPS\text{-}EQ}$ from [FHS14,FHS15], whose security holds in the generic group model. For the instantiation of signatures of knowledge (SoKs) in the ROM, we apply the Fiat-Shamir (FS) [FS87] heuristic to Σ-protocols and further apply the transformation from [FKMV12] to obtain simulation soundness.

Before we introduce the approaches to obtain CPA-fully (resp. $\mathsf{CCA2}$-fully) anonymous instantiations, we recall that the group signing key gsk_i consists of a vector of two group elements $(R, P) \in (\mathbb{G}_1^*)^2$ and an $\mathsf{SPS\text{-}EQ}$ signature $\sigma \in \mathbb{G}_1 \times \mathbb{G}_1^* \times \mathbb{G}_2^*$ on this vector. Randomization

[3] Note that using $\sigma_1||m$ as message in the SoK ensures that the conditions for the extractor to work with overwhelming probability are satisfied for every forgery output by the adversary.

of a gsk_i with a random value $\rho \in \mathbb{Z}_p^*$, i.e., $\mathsf{ChgRep}_\mathcal{R}$, requires 4 multiplications in $\mathbb{G}_1$ and 1 multiplication in $\mathbb{G}_2$. Verification of an SPS-EQ signature on gsk_i requires 5 pairings.

Finally, we note that the proofs performed using Π within Join and Open can straightforwardly be instantiated using standard techniques. Therefore, and since they are neither required within Sign nor Verify, we do not discuss their instantiation here.

5.1 CPA-Full Anonymity

Subsequently, we show how Sign and Verify are instantiated in the CPA-full anonymity setting. Therefore, let $H : \{0,1\}^* \to \mathbb{Z}_p$ be a random oracle and let x be the proven statement (which is implicitly defined by the scheme):

$\mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m):$ Parse gsk_i as $((R, P), \sigma)$, choose $\rho \xleftarrow{R} \mathbb{Z}_p$, compute $\sigma_1 = ((R', P'), \sigma') \leftarrow \mathsf{ChgRep}_\mathcal{R}(\mathsf{gsk}_i, \rho, \mathsf{pk}_\mathcal{R})$. Choose $\nu \xleftarrow{R} \mathbb{Z}_p$, compute $N \leftarrow \nu P$, $c \leftarrow H(N||\sigma_1||m||x)$, $z \leftarrow \nu + c \cdot \rho$, set $\sigma_2 \leftarrow (c, z)$, and return $\sigma \leftarrow (\sigma_1, \sigma_2)$.

$\mathsf{Verify}(\mathsf{gpk}, m, \sigma):$ Parse σ as $(\sigma_1, \sigma_2) = (((R', P'), \sigma), (c, z))$, return 0 if $\mathsf{Verify}_\mathcal{R}(\sigma_1, \mathsf{pk}_\mathcal{R}) = 0$. Otherwise compute $N \leftarrow zP - cP'$ and check whether $c = H(N||\sigma_1||m||x)$ holds. If so return 1 and 0 otherwise.

Since the used Σ-protocol is a standard proof of knowledge of the discrete logarithm $\log_P P'$, it is easy to see that applying the transformations from [FKMV12] yields a SoK in the ROM with the properties we require. All in all, group signatures contain 4 elements in $\mathbb{G}_1$, 1 element in $\mathbb{G}_2$ and 2 elements in $\mathbb{Z}_p$. Counting only the expensive operations, signing costs 5 multiplications in $\mathbb{G}_1$ and 1 multiplication in $\mathbb{G}_2$, and verification costs 2 multiplications in $\mathbb{G}_1$ and 5 pairings.

5.2 CCA2-Full Anonymity

When we want to achieve CCA2-full anonymity, we require our signatures of knowledge to be straight-line extractable, since standard rewinding techniques would lead to an exponential blowup in the reduction (cf. [BFW15]). One posssibility would be to rely on the rather inefficient approach to straight-line extraction due to Fischlin [Fis05]. However, as we do not need to straight-line extract the full witness w, but it is sufficient for us to straight-line extract an image of w under a one-way function $f : \rho \mapsto \rho \cdot \hat{P}$, we can fortunately use the notion of straight-line f-extractable SoKs as recently proposed by Cerulli et al. [BCC+15]. This allows us to still use the FS paradigm with good efficiency. The construction builds upon the generic conversion in [FKMV12, BPW12] and the generic trick in [BCC+15] to obtain straight-line f-extractability is by encrypting the image of the witness w under a function f with respect to a public key in the CRS and proving consistency with the witness.[4]

For straight-line extractability, we let $\hat{Y}$ be a public key for the ElGamal variant in $\mathbb{G}_2$ from [BCC+15], which is generated upon SoK.Setup and represents the CRS of SoK. SoK.SimSetup additionally returns τ such that $\hat{Y} = \tau \cdot \hat{P}$. Furthermore, let x be the proven statement (implicitly defined by the scheme and the generic compiler). Subsequently, we show how Sign and Verify are instantiated in this setting, where $H : \{0,1\}^* \to \mathbb{Z}_p$ is modelled as a random oracle:

$\mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, m):$ Parse gsk_i as $((R, P), \sigma)$, choose $\rho \xleftarrow{R} \mathbb{Z}_p$, compute $\sigma_1 = ((R', P'), \sigma') \leftarrow \mathsf{ChgRep}_\mathcal{R}(\mathsf{gsk}_i, \rho, \mathsf{pk}_\mathcal{R})$. Choose $u, \nu, \eta \xleftarrow{R} \mathbb{Z}_p$, compute $(\hat{C}_1, \hat{C}_2) = (u\hat{Y}, \rho\hat{P} + u\hat{P})$, $N \leftarrow \nu P$, $\hat{M}_1 \leftarrow \eta\hat{Y}$, $M_2 \leftarrow \nu\hat{P} + \eta\hat{P}$, $c \leftarrow H(N||M_1||M_2||\sigma_1||m||x)$, $z_1 \leftarrow \nu + c \cdot \rho$, $z_2 \leftarrow \eta + c \cdot u$, set $\sigma_2 \leftarrow (\hat{C}_1, \hat{C}_2, c, z_1, z_2)$, and return $\sigma \leftarrow (\sigma_1, \sigma_2)$.

$\mathsf{Verify}(\mathsf{gpk}, m, \sigma):$ Parse σ as $(\sigma_1, \sigma_2) = (((R', P'), \sigma), (c, z_1, z_2))$, return 0 if $\mathsf{Verify}_\mathcal{R}(\sigma_1, \mathsf{pk}_\mathcal{R}) = 0$. Otherwise compute $N \leftarrow z_1 P - cP'$, $M_1 \leftarrow z_2 \cdot \hat{Y} - c \cdot \hat{C}_1$, $M_2 \leftarrow (z_1 + z_2) \cdot \hat{P} - c \cdot \hat{C}_2$, and check whether $c = H(N||M_1||M_2||\sigma_1||m||x)$ holds. If so return 1 and 0 otherwise.

[4] Note that one can still obtain the full witness w using the rewinding extractor.

We now show that the above instantiation provides the properties we require. That is, we show that the Σ-protocol provides perfect completeness, special honest-verifier zero-knowledge (SHVZK) and provides special soundness. Moreover, we additionally require the Σ-protocol to provide quasi-unique responses [Fis05], i.e., given an accepting proof it should be infeasible to find a new valid response for that proof, in order for the compiler in [BCC+15] to apply.

Lemma 1. *The above Σ-protocol is perfectly complete, SHVZK, special-sound and has quasi-unique responses.*

Proof. We investigate all the properties below.

Perfect completeness. We omit perfect completeness since it is straight-forward to verify by inspection.

SHVZK. We describe a simulator which outputs transcripts being indistinguishable from real transcripts. First, it chooses $P' \xleftarrow{R} \mathbb{G}_1, \hat{C}_1 \xleftarrow{R} \mathbb{G}_2, \hat{C}_2 \xleftarrow{R} \mathbb{G}_2$. While P' and $\hat{C}_1$ are identically distributed as in a real transcript, the random choice of $\hat{C}_2$ is not detectable under DDH in $\mathbb{G}_2$ which holds in the SXDH setting (more generally under $\mathsf{IND\text{-}CPA}$ of the used encryption scheme). Then, the simulator chooses $z_1, z_2, c \xleftarrow{R} \mathbb{Z}_p$ and computes $N \leftarrow z_1 \cdot P - c \cdot P'$, $\hat{M}_1 \leftarrow z_2 \cdot \hat{Y} - c \cdot \hat{C}_1$, $\hat{M}_2 \leftarrow (z_1 + z_2) \cdot \hat{P} - c \cdot \hat{C}_2$. It is easy to see that the transcript $(P', \hat{C}_1, \hat{C}_2, N, \hat{M}_1, \hat{M}_2, z_1, z_2, c)$ represents a valid transcript and its distribution is computationally indistinguishable from a real transcript.

Special soundness. Let us consider that we have two accepting answers (z_1, z_2, c) and (z'_1, z'_2, c') from the prover for distinct challenges $c \neq c'$. Then we have that

$$z_1 - c \cdot \rho = z'_1 - c' \cdot \rho \text{ and } z_2 - c \cdot u = z'_2 - c' \cdot u,$$

and extract a witness as $\rho \leftarrow \frac{z_1 - z'_1}{c + c'}$, $u \leftarrow \frac{z_2 - z'_2}{c + c'}$.

Quasi-unique responses. The answers z_1 and z_2 are uniquely determined by the word $\hat{Y}$, P', $\hat{C}_1$, $\hat{C}_2$, the commitments N, $\hat{M}_1$, $\hat{M}_2$ as well as the challenge c (and thus the verification equation). ☐

Lemma 2. *Applying the generic conversions from [FKMV12] to the Fiat-Shamir transformed version of the above Σ-protocol with the setup* $\mathsf{SoK.Setup}$ *as described in Section 5.2 produces a signature of knowledge in the random oracle model, that is extractable and straight-line f-extractable.*

The proof is analogous to [BCC+15], but we re-state it for completeness.

Proof. For simulatability, we observe that the CRS output by $\mathsf{SoK.SimSetup}$ is identical to the CRS output by $\mathsf{SoK.Setup}$ and $\mathsf{SoK.SimSign}$ programs the random oracle to simulate proofs. Simulatability then follows from SHVZK. For extractability we rely on rewinding, special soundness and quasi-unique responses, using the results from [FKMV12]. For straight-line f-extractability, we use the trapdoor τ to decrypt $(\hat{C}_1, \hat{C}_2)$ in the proof transcript and obtain $\rho\hat{P} = f(\rho)$. ☐

Switching Groups. All in all, this instantiation yields signatures containing 4 elements in $\mathbb{G}_1$, 3 elements in $\mathbb{G}_2$, and 3 $\mathbb{Z}_p$ elements. Counting only the expensive operations, signing costs 5 multiplications in $\mathbb{G}_1$ and 7 multiplications in $\mathbb{G}_2$, and verification costs 2 multiplications in $\mathbb{G}_1$, 4 multiplications in $\mathbb{G}_2$, and 5 pairings. We observe that the protocol presented above now requires more operations in the more expensive group $\mathbb{G}_2$ than in $\mathbb{G}_1$. However, as we work in the SXDH setting, we can simply switch the roles of $\mathbb{G}_1$ and $\mathbb{G}_2$ and thus all elements in $\mathbb{G}_1$ to $\mathbb{G}_2$ and vice versa. This gives us improved computational performance at the expense of slightly larger signatures.

6 Evaluation and Discussion

To underline the practical efficiency of our approach, we provide a comparison of our ROM instantiation with other related schemes in the ROM. In particular we use two schemes who follow the approach of Bichsel et al., i.e., [BCN+10, PS16], which are proven secure in a weaker model (denoted BCN+), and the well known BBS scheme [BBS04] (with and without precomputations) providing the stronger notion of CPA-full anonymity. We note that we use the plain BBS scheme for comparison, which does not even provide non-frameability. The non-frameable version would be even more expensive. Moreover, we use the group signature scheme with the shortest known signatures [DP06] (with and without precomputations) being secure in the strong BSZ model and thus providing CCA2-full anonymity. In Table 1, we provide a comparison regarding signature size and computational costs. In Table 2 we provide a comparison

Scheme	Anon.	Signature Size	Signature Cost	Verification Cost
[BCN+10]	BCN+	$3\mathbb{G}_1 + 2\mathbb{Z}_p$	$1\mathbb{G}_T + 3\mathbb{G}_1$	$5\text{P} + 1\mathbb{G}_T + 1\mathbb{G}_1$
[PS16]	BCN+	$2\mathbb{G}_1 + 2\mathbb{Z}_p$	$1\mathbb{G}_T + 2\mathbb{G}_1$	$3\text{P} + 1\mathbb{G}_T + 1\mathbb{G}_1$
[BBS04]	CPA	$3\mathbb{G}_1 + 6\mathbb{Z}_p$	$3\text{P} + 3\mathbb{G}_T + 9\mathbb{G}_1$	$5\text{P} + 4\mathbb{G}_T + 8\mathbb{G}_1$
[BBS04] (prec.)	CPA	$3\mathbb{G}_1 + 6\mathbb{Z}_p$	$3\mathbb{G}_T + 9\mathbb{G}_1$	$4\mathbb{G}_T + 8\mathbb{G}_1$
This paper	CPA	$1\mathbb{G}_2 + 4\mathbb{G}_1 + 2\mathbb{Z}_p$	$1\mathbb{G}_2 + 5\mathbb{G}_1$	$5\text{P} + 2\mathbb{G}_1$
This paper	CCA2	$3\mathbb{G}_2 + 4\mathbb{G}_1 + 3\mathbb{Z}_p$	$7\mathbb{G}_2 + 5\mathbb{G}_1$	$5\text{P} + 4\mathbb{G}_2 + 2\mathbb{G}_1$
This paper (switch)	CCA2	$4\mathbb{G}_2 + 3\mathbb{G}_1 + 3\mathbb{Z}_p$	$5\mathbb{G}_2 + 7\mathbb{G}_1$	$5\text{P} + 2\mathbb{G}_2 + 4\mathbb{G}_1$
[DP06]	CCA2	$4\mathbb{G}_1 + 5\mathbb{Z}_p$	$3\text{P} + 3\mathbb{G}_T + 4\mathbb{G}_1$	$5\text{P} + 4\mathbb{G}_T + 7\,\mathbb{G}_1$
[DP06] (prec.)	CCA2	$4\mathbb{G}_1 + 5\mathbb{Z}_p$	$3\mathbb{G}_T + 8\mathbb{G}_1$	$1\text{P} + 3\mathbb{G}_T + 2\mathbb{G}_2 + 7\mathbb{G}_1$

Table 1. Comparison of related group signature schemes regarding signature size, signing and verification cost, where, in terms of computational costs, we only count the expensive operations in $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ as well as the pairings. The values for [BCN+10] and [PS16] are taken from [PS16]. We use 'BCN+' to denote anonymity in the sense of [BCN+10] and note that precomputation in [BBS04, DP06] requires to store extra elements in $\mathbb{G}_T$.

of the estimated efficiency in a 254bit BN-pairing setting, based on performance values on an ARM-Cortex-M0+ with drop-in hardware accelerator [UW14]. This processor is small enough to be suitable to be employed in smart cards or wireless sensor nodes [UW14].

Compared to [BBS04], our CPA-fully anonymous instantiation is more efficient regarding signature generation and verification by some orders of magnitude and we even obtain shorter signatures. Similarly, when comparing our CCA2-fully anonymous instantiation to [DP06] (which is to the best of our knowledge the most efficient CCA2-fully anonymous scheme), we are more efficient with respect to signature generation and verification but obtain larger signatures. Furthermore, for the schemes which do not achieve full anonymity and are proven secure in the weaker BCN+ model while following a similar paradigm as we do, the efficiency of our CPA-fully anonymous instantiation is in between [BCN+10] and [PS16] for verification. It seems that the stronger security notion in our scheme comes at the cost of larger signatures when compared to [BCN+10, PS16].

Regarding signature generation, we want to emphasize that our CPA-fully anonymous instantiation is the fastest among the schemes used for comparison, and, to the best of our knowledge the fastest among all existing scheme. This is of particular importance since signature generation is most likely to be executed on a constrained device.

Finally, we mention two interesting open points and leave a rigorous investigation open as future work. Firstly, it would be interesting to investigate whether our scheme provides the

Scheme	Anon.	Signature Size	Signature Cost	Verification Cost
[BCN+10]	BCN+	1273bit	351ms	1105ms
[PS16]	BCN+	1018bit	318ms	777ms
[BBS04]	CPA	2289bit	1545ms	2092ms
[BBS04] (prec.)	CPA	2289bit	1053ms	1600ms
This paper	CPA	2037bit	266ms	886ms
This paper	CCA2	3309bit	872ms	1290ms
This paper (switch)	CCA2	3563bit	736ms	1154ms
[DP06]	CCA2	2290bit	1380ms	2059ms
[DP06] (prec.)	CCA2	2290bit	1020ms	1353ms

Table 2. Estimated efficiency based on a BN-pairing implementation on an ARM-Cortex-M0+ with a drop-in hardware accelerator, operating at 48MHz [UW14]. At the 112bit security level, (254-bit curves), this implementation delivers the performance values 33ms-101ms-252ms-164ms ($\mathbb{G}_1$-$\mathbb{G}_2$-$\mathbb{G}_T$-pairing). For the estimation of signature sizes, we use 255bit for elements in $\mathbb{G}_1$, 509bit for elements in $\mathbb{G}_2$ and 254bit for elements in $\mathbb{Z}_p$. The semantics of 'BCN+' is the same as in Table 1. We note that [BBS04] is defined for a Type-2 pairing setting, which means that our performance estimation for this scheme is rather optimistic and likely to be worse in practice.

notion of opening soundness introduced by Sakai et al. [SSE+12]. Furthermore, our construction paradigm also seems to be interesting when it comes to efficient standard model instantiations.

References

[ACJT00] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference*, volume 1880, pages 255–270, 2000.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short Group Signatures. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference*, volume 3152 of *LNCS*, pages 41–55. Springer, 2004.

[BCC+15] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. Short Accountable Ring Signatures Based on DDH. In *Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I*, volume 9326 of *LNCS*, pages 243–265. Springer, 2015.

[BCN+10] Patrik Bichsel, Jan Camenisch, Gregory Neven, Nigel P. Smart, and Bogdan Warinschi. Get Shorty via Group Signatures without Encryption. In *Security and Cryptography for Networks, 7th International Conference, SCN 2010*, volume 6280 of *LNCS*, pages 381–398. Springer, 2010.

[BFW15] David Bernhard, Marc Fischlin, and Bogdan Warinschi. Adaptive proofs of knowledge in the random oracle model. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, volume 9020 of *LNCS*, pages 629–649. Springer, 2015.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional Signatures and Pseudorandom Functions. In *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography*, volume 8383, pages 501–519. Springer, 2014.

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions. In *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques*, volume 2656 of *LNCS*, pages 614–629. Springer, 2003.

[BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *LNCS*, pages 626–643. Springer, 2012.

[BSZ05] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of Group Signatures: The Case of Dynamic Groups. In *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, 2005.

[BW06] Xavier Boyen and Brent Waters. Compact Group Signatures Without Random Oracles. In *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, volume 4004 of *LNCS*, pages 427–444. Springer, 2006.

[BW07] Xavier Boyen and Brent Waters. Full-Domain Subgroup Hiding and Constant-Size Group Signatures. In *Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography*, volume 4450 of *LNCS*, pages 1–15. Springer, 2007.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference*, volume 3152 of *LNCS*, pages 56–72. Springer, 2004.

[CL06] Melissa Chase and Anna Lysyanskaya. On Signatures of Knowledge. In *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference*, volume 4117 of *LNCS*, pages 78–96. Springer, 2006.

[CS97] Jan Camenisch and Markus Stadler. Efficient Group Signature Schemes for Large Groups (Extended Abstract). In *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference*, volume 1294 of *LNCS*, pages 410–424. Springer, 1997.

[CvH91] David Chaum and Eugène van Heyst. Group Signatures. In *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques*, volume 547 of *LNCS*, pages 257–265. Springer, 1991.

[DP06] Cécile Delerablée and David Pointcheval. Dynamic Fully Anonymous Short Group Signatures. In *Progressin Cryptology - VIETCRYPT 2006, First International Conferenceon Cryptology in Vietnam*, volume 4341 of *LNCS*, pages 193–210. Springer, 2006.

[FHS14] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-Preserving Signatures on Equivalence Classes and Constant-Size Anonymous Credentials. IACR Cryptology ePrint Archive, 2014.

[FHS15] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical Round-Optimal Blind Signatures in the Standard Model. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference*, volume 9216 of *LNCS*, pages 233–253. Springer, 2015.

[Fis05] Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *LNCS*, pages 152–168. Springer, 2005.

[FKMV12] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the fiat-shamir transform. In *Progress in Cryptology - INDOCRYPT 2012, 13th International Conference on Cryptology in India, Kolkata, India, December 9-12, 2012. Proceedings*, volume 7668 of *LNCS*, pages 60–79. Springer, 2012.

[FS87] Amos Fiat and Adi Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *CRYPTO'87*, volume 263 of *LNCS*, pages 186–194. Springer, 1987.

[GMR88] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM JoC*, 17(2):281–308, 1988.

[Gro07] Jens Groth. Fully Anonymous Group Signatures Without Random Oracles. In *Advances in Cryptology - ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security*, volume 4833 of *LNCS*, pages 164–180. Springer, 2007.

[HS14] Christian Hanser and Daniel Slamanig. Structure-Preserving Signatures on Equivalence Classes and Their Application to Anonymous Credentials. In *Advances in Cryptology -*

ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, volume 8873 of *LNCS*, pages 491–511. Springer, 2014.

[LLM+16] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Signature Schemes with Efficient Protocols and Dynamic Group Signatures from Lattice Assumptions. Cryptology ePrint Archive, Report 2016/101, 2016. http://eprint.iacr.org/.

[LPY15] Benoît Libert, Thomas Peters, and Moti Yung. Short Group Signatures via Structure-Preserving Signatures: Standard Model Security from Simple Assumptions. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference*, pages 296–316, 2015.

[NS04] Lan Nguyen and Reihaneh Safavi-Naini. Efficient and provably secure trapdoor-free group signature schemes from bilinear pairings. In *Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security*, pages 372–386, 2004.

[PS16] David Pointcheval and Olivier Sanders. Short Randomizable Signatures. In *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016*, volume 9610 of *LNCS*, pages 111–126. Springer, 2016.

[SSE+12] Yusuke Sakai, Jacob C. N. Schuldt, Keita Emura, Goichiro Hanaoka, and Kazuo Ohta. On the Security of Dynamic Group Signatures: Preventing Signature Hijacking. In *Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography*, volume 7293 of *LNCS*, pages 715–732. Springer, 2012.

[UW14] Thomas Unterluggauer and Erich Wenger. Efficient Pairings and ECC for Embedded Systems. In *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop*, volume 8731 of *LNCS*, pages 298–315. Springer, 2014.