Round-optimal Constant-size Blind Signatures

Olivier Blazy[1], Brouilhet Laura[1], Céline Chevalier[2] and Neals Fournaise[1]
[1]*Université de Limoges, XLim, Limoges, France*
[2]*Université Panthéon-Assas, Paris, France*

Keywords: Blind-signature, Round-optimal, Standard Model, e-Voting.

Abstract: Blind signatures schemes allow a user to obtain a signature on messages from a signer, ensuring blindness (the signer should not learn which messages he signed or in which order) and unforgeability (the user should not be able to produce more signatures than the number of times he interacted with the signer). For practical purposes, it is important that such schemes are *round-optimal* (one flow sent by the user and one by the signer) and *constant-size* (the amount of data sent during the interaction should not depend on the length of the message), which are two properties difficult to ensure together. In this paper, we propose the first blind signature scheme both round-optimal, constant-size, in the standard model (without any random oracle) and under a classical assumption (SXDH). Our construction follows the classical framework initially presented by Fischlin. As a side result, we first show how to use a special kind of structure-preserving signatures (where the signatures also are group elements) in order to construct the first constant-size signatures on randomizable ciphertexts, a notion presented a few years ago by Blazy *et al*. Our construction of blind signature then builds upon this primitive and consists of constant-size two-round communication. It can be instantiated under any $k-$MDDH assumption, requires to exchange 9 elements and leads to a final signature with 22 elements when relying on SXDH. .

1 INTRODUCTION

Digital Signature Schemes are well-known cryptographic primitives analogous to manuscript signatures. They are commonly used to allow a recipient to strongly believe that a message or document was created by the supposed sender (authentication) and that it has not been altered (integrity). Such schemes are supposed to be *unforgeable*, in the sense that an adversary should be unable to output a valid signature after having had access to a certain number of valid signatures.

Blind Signature Schemes, introduced in (Chaum, 1982) are a special kind of digital signature schemes, with an additional property, called *blindness*, on top of a variant of the notion of *unforgeability*. The blindness property means that in such schemes, the user does not sign the messages by himself, but rather interacts with a signer, with the guarantee that the signer will learn nothing about the signed message nor the resulting signature. More precisely, the view of the signer should be unlinkable to the (message/signature) pairs resulting from several executions of the protocol (he cannot link a pair to a specific execution).

The second security property for blind signatures is a notion of unforgeability, which intuitively means that after n interactions, a user should not obtain more than n signatures (on different messages). This property has been formalized in (Pointcheval and Stern, 2000), motivated by the use of blind signatures for e-cash: a user should not be able to produce more (message/signature) pairs (coins) than the number of signing executions with the bank (withdrawals). The security model was further revisited in (Schröder and Unruh, 2012) for other contexts.

Blind signature schemes were introduced as a fundamental building block for applications that guarantee user anonymity, e.g. e-cash (Chaum, 1982; Chaum et al., 1990; Okamoto and Ohta, 1992; Camenisch et al., 2005; Fuchsbauer et al., 2009), e-voting solutions (Fujioka et al., 1993; Blazy et al., 2011; Blazy et al., 2012a), and anonymous credentials (Brands, 1994; Camenisch and Lysyanskaya, 2001; Belenkiy et al., 2009; Fuchsbauer, 2011). Due to their practical interest, it is extremely important that such schemes offer low complexity both in terms of number of rounds (not too much interaction between the user and the signer) and amount of data exchanged (amount independent of the length of the signed message, if possible). Furthermore, for security reasons, it is better to design schemes in the standard model (without any random oracle (Bellare and Rogaway,

Blazy, O., Laura, B., Chevalier, C. and Fournaise, N.
Round-optimal Constant-size Blind Signatures.
DOI: 10.5220/0009888702130224
In *Proceedings of the 17th International Joint Conference on e-Business and Telecommunications (ICETE 2020) - SECRYPT*, pages 213-224
ISBN: 978-989-758-446-6

1993)) and proven secure under a classical and well-accepted security assumption.

Related Work. The first constructions for blind signature schemes (such as (Okamoto, 1993; Okamoto, 2006)) were highly interactive, until Fischlin proposed a generic construction of round-optimal blind signature schemes in (Fischlin, 2006), with only two flows of communication between the user and the signer. Several constructions have since then efficiently instantiated this transformation, but at the cost of exchanging information of size depending on the message length (Blazy et al., 2011; Blazy et al., 2012b; Blazy et al., 2012a), or by relying on the Random Oracle Model (Pointcheval and Stern, 2000; Abe, 2001; Baldimtsi and Lysyanskaya, 2013; Hauck et al., 2019).

To the best of our knowledge, the only constant-size instantiations in the standard model (without any random oracle) were given in (Abe et al., 2010; Ghadafi, 2017). The former is given in a pairing-based setting with a final signature consisting of 18 elements in the first group $\mathbb{G}_1$ and 16 in the second one $\mathbb{G}_2$, but relies on a new ad-hoc q-type assumption. In (Ghadafi, 2017), the proposed blind signature is in the standard model but under a non standard q-type assumption: The *Blind Signature One More*, which basically assumes the security of the scheme and could likely only be proven in the generic model.

Contributions. In this paper, we answer an open question mentioned in the presentation of (Hauck et al., 2019) at Eurocrypt'19 by proposing a new blind signature scheme, which is round-optimal, constant-size, in the standard model and with a classical assumption. Our construction follows the framework presented by Fischlin (Fischlin, 2006), and adapts an idea from (Blazy et al., 2011) to decrease some cost.

The main tool used in our construction can be seen as a side contribution: we give the first signature scheme on randomizable ciphertexts (Blazy et al., 2011) based on structure-preserving signatures, which are furthermore constant-size. To this aim, we prove that we can adapt the structure-preserving signatures proposed in (Kiltz et al., 2015) to sign classes of elements, in the spirit of (Hanser and Slamanig, 2014). A class of elements is a group of elements which are ciphertexts of the same message. In other words, these schemes allow to give a signature on a ciphertext C, such that one can derive a signature on any randomization of this ciphertext. Of course, the unforgeability still guarantees that no adversary can generate a signature on an unsigned class.

When comparing our schemes with existing ones (see Figure 1), one can see that we manage to keep all the communication constant-size with respect to the message length ℓ, while relying on a standard assumption.[1].

Organization of the Paper. The paper starts by recalling classical definitions and security experiments in Section 2 and useful building blocks in Section 3. We then proceed by introducing our constant-size signature on randomizable ciphertexts (SRC) in Section 4 and describing our constant-size blind signature scheme in Section 5. We finally give another application of SRC schemes to e-voting in Section 6.

2 DEFINITIONS

2.1 General Notations and Assumptions

In all the remaining of this paper, we will work in a pairing-based setting whose notations are recalled below. We will also use the standard security assumptions in such a setting: we recall them below for completeness. Let $\mathfrak{K}$ be the security parameter.

Pairing Groups. Let GGen be a probabilistic polynomial time (PPT) algorithm that on input $1^{\mathfrak{K}}$ returns a description $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ of asymmetric pairing groups where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are cyclic groups of order p for a $\mathfrak{K}$-bit prime p, g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable (non-degenerated) bilinear map. Define $g_T := e(g_1, g_2)$, which is a generator in $\mathbb{G}_T$.

Definition 1 (Decisional Diffie-Hellman (DDH))**.** *Let $\mathbb{G}$ be a cyclic group of prime order p. The* DDH *assumption states that given $(g, g^a, g^b, g^c) \in \mathbb{G}$, it is hard to determine whether $c = ab$.*

Definition 2 (External Diffie-Hellman (XDH (Boneh et al., 2004)))**.** *This variant of* DDH*, states that while the* DDH *is easy in $\mathbb{G}_2$,* DDH *is hard in $\mathbb{G}_1$.*

Definition 3 (Symmetric External Diffie-Hellman (SXDH (Ateniese et al., 2005)))**.** *This variant of* DDH*, used mostly in bilinear groups in which no computationally efficient homomorphism exists from $\mathbb{G}_2$ to $\mathbb{G}_1$ or $\mathbb{G}_1$ to $\mathbb{G}_2$, states that* DDH *is hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$.*

[1] In this figure, combining (Blazy et al., 2012b) and (Blazy et al., 2012a) could lead to a scheme with sublinear communication cost $O(\log(\ell))$, but at the cost of a slightly weaker definition of blindness (a-posteriori blindness). Furthermore, the user communication cost would not still be constant.

Scheme	User	Server	Signature	# Rounds	Assumptions
(Abe et al., 2010)	$2\mathbb{G}_1$	$3\mathbb{G}_1, 3\mathbb{G}_2$	16 $\mathbb{G}_1$, 14 $\mathbb{G}_2$	2	$q-$ADH$-$SDH
(Blazy et al., 2011)	$O(\ell)\mathbb{G}_1, O(\ell)\mathbb{G}_2$	$2\mathbb{G}_1, 1\mathbb{G}_2$	2 $\mathbb{G}_1, 1\mathbb{G}_2$	2	SXDH
(Blazy et al., 2012b)	$O(\ell/\log(\ell))\mathbb{G}_1$	$2\mathbb{G}_1, 1\mathbb{G}_2$	2 $\mathbb{G}_1, 1\mathbb{G}_2$	2	SXDH
Ours	$2\mathbb{G}_1$	$6\mathbb{G}_1, 1\mathbb{Z}_p$	$12\mathbb{G}_1, 8\mathbb{G}_2$	2	SXDH
Asym	$(k+1)\mathbb{G}_1$	$3(k+1)\mathbb{G}_1, 1\mathbb{Z}_p$	$(k+1)(2k+4)\mathbb{G}_1$, $(k+1)(3k+3)\mathbb{G}_2$	2	$k-$MDDH
Sym	$(k+1)\mathbb{G}$	$3(k+1)\mathbb{G}, 1\mathbb{Z}_p$	$(k+1)(5k+7)\mathbb{G}$	2	$k-$MDDH

Figure 1: Efficiency comparison of our solutions with existing schemes in the standard model.

2.2 Matricial Notations and Assumptions

Our schemes, that we present under the SXDH assumption for the sake of simplicity, can be proven under any $k-$MDDH security assumption. This assumption, presented in (Escala et al., 2013), needs a few notations to be fully understood: this part can be omitted at first read but we recall them below for completeness.

Matricial Notations. If $\mathbf{A} \in \mathbb{Z}_p^{(k+1)\times n}$ is a matrix, then $\overline{\mathbf{A}} \in \mathbb{Z}_p^{k\times n}$ denotes the upper matrix of $\mathbf{A}$ and $\underline{\mathbf{A}} \in \mathbb{Z}_p^{1\times n}$ denotes the last row of $\mathbf{A}$.

We use implicit representation of group elements as introduced in (Escala et al., 2013). For $s \in \{1,2,T\}$ and $a \in \mathbb{Z}_p$ define $[a]_s = g_s^a \in \mathbb{G}_s$ as the *implicit representation* of a in $\mathbb{G}_s$ (we use $[a] = g^a \in \mathbb{G}$ if we consider a unique group). More generally, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_p^{n\times m}$ we define $[\mathbf{A}]_s$ as the implicit representation of $\mathbf{A}$ in $\mathbb{G}_s$:

$$[\mathbf{A}]_s := \begin{pmatrix} g_s^{a_{11}} & \dots & g_s^{a_{1m}} \\ & & \\ g_s^{a_{n1}} & \dots & g_s^{a_{nm}} \end{pmatrix} \in \mathbb{G}_s^{n\times m}$$

We will always use this implicit notation of elements in $\mathbb{G}_s$, i.e., we let $[a]_s \in \mathbb{G}_s$ be an element in $\mathbb{G}_s$. Note that from $[a]_s \in \mathbb{G}_s$ it is generally hard to compute the value a (discrete logarithm problem in $\mathbb{G}_s$). Further, from $[b]_T \in \mathbb{G}_T$ it is hard to compute the value $[b]_1 \in \mathbb{G}_1$ and $[b]_2 \in \mathbb{G}_2$ (pairing inversion problem). Obviously, given $[a]_s \in \mathbb{G}_s$ and a scalar $x \in \mathbb{Z}_p$, one can efficiently compute $[ax]_s \in \mathbb{G}_s$. Further, given $[a]_1, [b]_2$ one can efficiently compute $[ab]_T$ using the pairing e. For $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^k$ define $e([\mathbf{a}]_1, [\mathbf{b}]_2) := [\mathbf{a}^\top \mathbf{b}]_T \in \mathbb{G}_T$.

Assumptions. We recall the definition of the matrix Diffie-Hellman (MDDH) assumption (Escala et al., 2013).

Definition 4 (Matrix Distribution)**.** *Let $k \in \mathbb{N}$. We call $\mathcal{D}_k$ a matrix distribution if it outputs matrices in $\mathbb{Z}_p^{(k+1)\times k}$ of full rank k in polynomial time.*

Without loss of generality, we assume the first k rows of $\mathbf{A} \xleftarrow{\$} \mathcal{D}_k$ form an invertible matrix. The $\mathcal{D}_k$-Matrix Diffie-Hellman problem is to distinguish the two distributions $([\mathbf{A}], [\mathbf{Aw}])$ and $([\mathbf{A}], [\mathbf{u}])$ where $\mathbf{A} \xleftarrow{\$} \mathcal{D}_k$, $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_p^k$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$.

Definition 5 ($\mathcal{D}_k$-Matrix Diffie-Hellman Assumption $\mathcal{D}_k$-MDDH)**.** *Let $\mathcal{D}_k$ be a matrix distribution and $s \in \{1,2,T\}$. We say that the $\mathcal{D}_k$-Matrix Diffie-Hellman ($\mathcal{D}_k$-MDDH) Assumption holds relative to* GGen *in group $\mathbb{G}_s$ if for all PPT adversaries $\mathcal{D}$,*

$$\begin{aligned}\mathbf{Adv}_{\mathcal{D}_k,\mathsf{GGen}}(\mathcal{D}) &\stackrel{\text{def}}{=} |\Pr[\mathcal{D}(\mathcal{G}, [\mathbf{A}]_s, [\mathbf{Aw}]_s) = 1] \\ &\quad - \Pr[\mathcal{D}(\mathcal{G}, [\mathbf{A}]_s, [\mathbf{u}]_s) = 1]| \\ &= \mathsf{negl}(\lambda),\end{aligned}$$

where the probability is taken over $\mathcal{G} \xleftarrow{\$} \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \xleftarrow{\$} \mathcal{D}_k, \mathbf{w} \xleftarrow{\$} \mathbb{Z}_p^k, \mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$.

Definition 6 ($\mathcal{D}_k$-Kernel Diffie-Hellman Assumption $\mathcal{D}_k - \mathsf{KerMDDH}$)**.** *Let $\mathcal{D}_k$ be a matrix distribution and $s \in \{1,2\}$. We define $\mathsf{Adv}^{kmddh}_{\mathcal{D}_k,\mathsf{GGen}}(\mathcal{D})$ by*

$$\Pr[\mathbf{c}^\top \mathbf{A} = \mathbf{0} \wedge \mathbf{c} \neq \mathbf{0} \,|\, [\mathbf{c}]_{3-s} \xleftarrow{\$} \mathcal{D}(\mathcal{G}, [\mathbf{A}]_s)]$$

where the probability is taken over $\mathcal{G} \xleftarrow{\$} \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \xleftarrow{\$} \mathcal{D}_k$. We say that the $\mathcal{D}_k$-Kernel Diffie-Hellman Assumption ($\mathcal{D}_k - \mathsf{KerMDDH}$) assumption holds relative to GGen *in group $\mathbb{G}_s$, if for all PPT adversaries $\mathcal{D}$,*

$$\mathsf{Adv}^{kmddh}_{\mathcal{D}_k,\mathsf{GGen}}(\mathcal{D}) = \mathsf{negl}(\lambda)$$

In the following, we write $k-$MDDH for $\mathcal{D}_k -$ MDDH. It should be noted that $1-$MDDH is the SXDH assumption used in this paper.

2.3 Cryptographic Primitives

We now present the classical cryptographic primitives used in this paper: (randomizable) encryption schemes and (blind) signature schemes.

Definition 7 (Encryption scheme)**.** *An encryption scheme $\mathcal{E}$ is described by four algorithms* $(\mathsf{Setup}_\mathcal{E}, \mathsf{KeyGen}_\mathcal{E}, \mathsf{Enc}, \mathsf{Decrypt})$*:*

- $\mathsf{Setup}_\mathcal{E}(\mathfrak{K})$*, where $\mathfrak{K}$ is the security parameter, generates the global parameters* param *of the scheme;*

- $\mathsf{KeyGen}_{\mathcal{E}}(\mathsf{param})$ *outputs the encryption and decryption key (*ek,dk*);*
- $\mathsf{Enc}(\mathsf{ek},m;\rho)$ *outputs a ciphertext* $\mathbf{c}$*, on the message* m*, under* ek*, with the randomness* ρ*;*
- $\mathsf{Decrypt}(\mathsf{dk},\mathbf{c})$ *outputs the plaintext* m *or* $\perp$.

Such encryption scheme is required to have the following security properties:

- *Correctness*: For every $(\mathsf{ek},\mathsf{dk})$ generated by $\mathsf{KeyGen}_{\mathcal{E}}$, every messages m, every random ρ, we have
  $\mathsf{Decrypt}(\mathsf{dk},\mathsf{Enc}(\mathsf{ek},m;\rho)) = m$.
- *Indistinguishability under Chosen Plaintext Attack (Goldwasser and Micali, 1984)*: This notion (IND-CPA), states that an adversary should not be able to efficiently guess which message has been encrypted even if he chooses the two original plaintexts. The game is described in Figure 2. The advantages are:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\mathfrak{K}) = |\Pr[\mathsf{Exp}^{\mathsf{ind}-1}_{\mathcal{E},\mathcal{A}}(\mathfrak{K}) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ind}-0}_{\mathcal{E},\mathcal{A}}(\mathfrak{K}) = 1]$$

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E}}(\mathfrak{K},t) = \max_{\mathcal{A}\leq t} \mathsf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\mathfrak{K}).$$

$\mathsf{Exp}^{\mathsf{ind}-b}_{\mathcal{E},\mathcal{A}}(\mathfrak{K})$
1. $\mathsf{param} \leftarrow \mathsf{Setup}_{\mathcal{E}}(1^{\mathfrak{K}})$
2. $(\mathsf{ek},\mathsf{dk}) \leftarrow \mathsf{KeyGen}_{\mathcal{E}}(\mathsf{param})$
3. $(m_0,m_1) \leftarrow \mathcal{A}(\texttt{FIND} : \mathsf{ek})$
4. $c^* \leftarrow \mathsf{Enc}(\mathsf{ek},m_b)$
5. $b' \leftarrow \mathcal{A}(\texttt{GUESS} : c^*)$
6. `RETURN` b'

Figure 2: IND-CPA Game for an Encryption Scheme.

Sometimes, one may want to be able to publicly randomize the ciphertext c, using the following algorithm:

- $\mathsf{Random}(\mathsf{ek},c;r')$ outputs a new ciphertext c' equivalent to the ciphertext c, under the public key ek, using the additional random coins $r' \xleftarrow{\$} \mathcal{R}_e$.

An encryption scheme is called randomizable if a randomized ciphertext is indistinguishable from a fresh one.

Definition 8 (Signature scheme)**.** *A signature scheme is composed by four polynomial time algorithms* $\mathcal{S} = (\mathsf{Setup}_{\mathcal{S}},\mathsf{KeyGen}_{\mathcal{S}},\mathsf{Sign},\mathsf{Verify})$.

- $\mathsf{Setup}_{\mathcal{S}}(\mathfrak{K})$ *outputs the global parameters* param *of the scheme.*
- $\mathsf{KeyGen}_{\mathcal{S}}(\mathsf{param})$ *outputs the signature and verification keys:* $(\mathsf{sk},\mathsf{vk})$.
- $\mathsf{Sign}(\mathsf{sk},m;s)$ *outputs a signature* σ *on message* m *using randomness* $s \in \mathcal{R}$.
- $\mathsf{Verify}(\mathsf{vk},\sigma)$ *checks if* σ *is a valid signature. It outputs 1 if the signature is valid, 0 otherwise.*

Such signature scheme is required to have the following security property:

- *Existential Unforgeability under Chosen Message Attacks (Goldwasser et al., 1988)* $(\mathsf{EUF}-\mathsf{CMA})$. Even after querying n valid signatures on chosen messages (m_i), $\mathcal{A}$ should not be able to output a valid signature on a fresh message m. We define a signing oracle:
  $\mathsf{OSign}(\mathsf{vk},m)$: outputs a signature on m valid under the verification key vk. The requested message is added to the signed messages set $\mathcal{SM}$.

The probability of success against the game given in Figure 3 is denoted by

$$\mathsf{Succ}^{\mathsf{euf}}_{\mathcal{S},\mathcal{A}}(\mathfrak{K}) = \Pr[\mathsf{Exp}^{\mathsf{euf}}_{\mathcal{S},\mathcal{A}}(\mathfrak{K}) = 1],$$

$$\mathsf{Succ}^{\mathsf{euf}}_{\mathcal{S}}(\mathfrak{K},t) = \max_{\mathcal{A}\leq t} \mathsf{Succ}^{\mathsf{euf}}_{\mathcal{S},\mathcal{A}}(\mathfrak{K}).$$

$\mathsf{Exp}^{\mathsf{euf}}_{\mathcal{S},\mathcal{A}}(\mathfrak{K})$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^{\mathfrak{K}})$
2. $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{SKeyGen}(\mathsf{param})$
3. $(m^*,\sigma^*) \leftarrow \mathcal{A}(\mathsf{vk},\mathsf{OSign}(\mathsf{vk},\cdot))$
4. $b \leftarrow \mathsf{Verify}(\mathsf{vk},m^*,\sigma^*)$
5. `IF` $m^* \in \mathcal{SM}$ `RETURN` 0
6. `ELSE RETURN` b

Figure 3: $\mathsf{EUF}-\mathsf{CMA}$ Game for a Signature Scheme.

In our work, we use a signature scheme, that of (Kiltz et al., 2015), which is also Structure-Preserving. In such a scheme, both the messages to be signed and the signatures are group elements.

Definition 9 (Blind signature scheme)**.** *A blind signature scheme is defined by three polynomial time algorithms and one interactive polynomial time protocol* $\mathcal{BS} = (\mathsf{BSSetup},\mathsf{BSKeyGen},\mathsf{BSProtocol}\langle\mathcal{S},\mathcal{U}\rangle,\mathsf{Verify})$.

- $\mathsf{BSSetup}(\mathfrak{K})$ *outputs the parameters* param *of the scheme.*
- $\mathsf{BSKeyGen}(\mathsf{param})$ *outputs the signature/verification keys:* $(\mathsf{sk},\mathsf{vk})$.
- $\mathsf{BSProtocol}\langle\mathcal{S}(\mathsf{sk}),\mathcal{U}(\mathsf{vk},m)\rangle$ *is an interactive protocol between user* $\mathcal{U}$ *and signer/server* $\mathcal{S}$*. It issues a signature* σ *on* m *valid under* vk.
- $\mathsf{Verify}(\mathsf{vk},m,\sigma)$ *checks if* σ *is a valid signature. It outputs 1 if* σ *is valid, 0 otherwise.*

The two expected security properties are the *unforgeability*, protecting the signer, and the *blindness*, protecting the user (see Figure 4 and Figure 5).

- The *unforgeability* is the $\mathsf{EUF}-\mathsf{CMA}$ property.
- The *blindness* property says that a malicious signer who signed two messages m_0 and m_1 shouldn't be able to decide which one was signed first.

$\mathsf{Exp}^{\mathsf{uf}}_{\mathcal{BS},\mathcal{U}^*}(\mathfrak{K})$
1. $(\mathsf{param}) \leftarrow \mathsf{BSSetup}(1^{\mathfrak{K}})$
2. $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{BSKeyGen}(\mathsf{param})$
3. For $i = 1,\ldots,q_s$, $\mathsf{BSProtocol}\langle\mathcal{S}(\mathsf{sk}),\mathcal{A}(\texttt{INIT}:\mathsf{vk})\rangle$
4. $\big((m_1,\sigma_1),\ldots,(m_{q_s+1},\sigma_{q_s+1})\big) \leftarrow \mathcal{A}(\texttt{GUESS}:\mathsf{vk})$;
5. IF $\exists i \neq j, m_i = m_j$ OR $\exists i, \mathsf{Verify}(\mathsf{vk},m_i,\sigma_i) = 0$
   RETURN 0
6. ELSE RETURN 1

Figure 4: Unforgeability Game for a $\mathcal{BS}$ Scheme.

$\mathsf{Exp}^{\mathsf{bl}-b}_{\mathcal{BS},\mathcal{S}^*}(\mathfrak{K})$
1. $\mathsf{param} \leftarrow \mathsf{BSSetup}(1^{\mathfrak{K}})$
2. $(\mathsf{vk},m_0,m_1) \leftarrow \mathcal{A}(\texttt{FIND}:\mathsf{param})$
3. $\sigma_b \leftarrow \mathsf{BSProtocol}\langle\mathcal{A},\mathcal{U}(\mathsf{vk},m_b)\rangle$
4. $\sigma_{1-b} \leftarrow \mathsf{BSProtocol}\langle\mathcal{A},\mathcal{U}(\mathsf{vk},m_{1-b})\rangle$
5. $b^* \leftarrow \mathcal{S}^*(\texttt{GUESS}:m_0,m_1)$;
6. RETURN $b^* = b$.

Figure 5: Blindness Game for a $\mathcal{BS}$ Scheme.

3 BUILDING BLOCKS

In this section, we present the standard instantiations of the cryptographic primitives presented in the former section that we will use in the remaining of this paper: ElGamal encryption scheme, Kiltz *et al*'s structure-preserving signature scheme and Groth-Sahai commitments.

3.1 ElGamal Encryption Scheme

The four algorithms of this encryption scheme (ElGamal, 1984) are described as follows:

- $\mathsf{Setup}_{\mathcal{E}}(1^{\mathfrak{K}})$: outputs $\mathsf{param} = (\mathbb{G},p,[1])$.
- $\mathsf{KeyGen}_{\mathcal{E}}(\mathsf{param})$: outputs $(\mathsf{ek},\mathsf{dk}) = ([h],h)$ with $h \xleftarrow{\$} \mathbb{Z}_p$.
- $\mathsf{Enc}(\mathsf{ek},[m];r)$ outputs $c = (c_1,c_2) = ([r,rh+m])$ with $r \xleftarrow{\$} \mathbb{Z}_p$.
- $\mathsf{Decrypt}(\mathsf{dk},c)$ computes $[c_2 - hc_1] = [m]$, outputs $[m]$.

This scheme is semantically secure against chosen-plaintext attacks (IND-CPA) under DDH.

3.2 Structure-preserving Signatures

We will recall here the SXDH version of the structure-preserving signature (SPS) of (Kiltz et al., 2015) in figure 6. As we need to sign only one message we set their parameters $k = n = 1$.

$\underline{\mathsf{Setup}_{\mathcal{S}}(\mathfrak{K})}$:
Return (param)

$\underline{\mathsf{KeyGen}_{\mathcal{S}}(\mathsf{param})}$:
$\mathbf{A},\mathbf{B} \xleftarrow{\$} \mathcal{D}_1; \mathbf{K} \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}$ $\mathbf{K_0},\mathbf{K_1} \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}$
$\mathbf{C} \stackrel{\text{def}}{=} \mathbf{KA} \in \mathbb{Z}_p^{2\times 1}$
$(\mathbf{C_0},\mathbf{C_1}) \stackrel{\text{def}}{=} (\mathbf{K_0A},\mathbf{K_1A}) \in (\mathbb{Z}_p^{2\times 1})^2$
$(\mathbf{P_0},\mathbf{P_1}) \stackrel{\text{def}}{=} (\mathbf{B}^\intercal\mathbf{K_0},\mathbf{B}^\intercal\mathbf{K_1}) \in (\mathbb{Z}_p^{1\times 2})^2$
$\mathbf{sk} \stackrel{\text{def}}{=} (\mathbf{K},[\mathbf{P_0}]_1,[\mathbf{P_1}]_1,[\mathbf{B}]_1)$;
$\mathbf{vk} \stackrel{\text{def}}{=} ([\mathbf{C_0}]_2,[\mathbf{C_1}]_2,[\mathbf{C}]_2,[\mathbf{A}]_2)$

$\underline{\mathsf{Sign}(\mathsf{sk},\mathbf{m})}$:
$\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p; \tau \xleftarrow{\$} \mathbb{Z}_p$
$\sigma_1 \stackrel{\text{def}}{=} [(1,\mathbf{m})\mathbf{K} + \mathbf{r}^\intercal(\mathbf{P_0} + \tau\mathbf{P_1})]_1 \in \mathbb{G}_1^{1\times 2}$,
$\sigma_2 \stackrel{\text{def}}{=} [\mathbf{r}^\intercal\mathbf{B}^\intercal]_1 \in \mathbb{G}_1^{1\times 2}$, $\sigma_3 \stackrel{\text{def}}{=} [\mathbf{r}^\intercal\mathbf{B}^\intercal\tau]_1 \in \mathbb{G}_1^{1\times 2}$,
$\sigma_\tau \stackrel{\text{def}}{=} [\tau]_2 \in \mathbb{G}_2$
Return $(\sigma_1,\sigma_2,\sigma_3,\sigma_\tau)$

$\underline{\mathsf{Verify}(\mathsf{vk},\mathbf{m},\sigma)}$:
Parse $\sigma = (\sigma_1,\sigma_2,\sigma_3,\sigma_4)$
Check $[\sigma_1 \cdot \mathbf{A}]_T = [(1,\mathbf{m}) \cdot \mathbf{C} + \sigma_2 \cdot \mathbf{C_0} + \sigma_3 \cdot \mathbf{C_1}]_T$
and $[\sigma_2 \cdot \sigma_4]_T = [\sigma_3]_T$

Figure 6: the SPS algorithms from (Kiltz et al., 2015).

3.3 Groth-Sahai Commitments

In (Groth and Sahai, 2008), Groth and Sahai proposed non-interactive zero-knowledge proof systems for bilinear groups. It allows a prover to convince a verifier that he possesses group elements or scalars satisfying equations of a particular form. For our work, we mainly use the satisfiability of pairing product equations. Various instantiations were proposed in this seminal paper based on common hardness assumptions such as DLin and SXDH.

Initialization. We work in a bilinear group $(p,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T,e,g_1,g_2)$. The commitment keys for group $\mathbb{G}_1$ is $\mathbf{u} = (\mathbf{u_1},\mathbf{u_2})$. We initialize it with random values $\alpha,\beta \xleftarrow{\$} \mathbb{Z}_p^*$ as $\mathbf{u_1} = [1,\alpha]_1, \mathbf{u_2} = [\beta,\alpha\beta]_1$. Hence, $\mathbf{u}$ is a Diffie-Hellman tuple in $\mathbb{G}_1$. This commitment is binding but can be set to hiding if we define instead $\mathbf{u_2} = \beta\mathbf{u_1} - (1,g_1)$. The commitment key $\mathbf{v}$ can be analogously defined for $\mathbb{G}_2$.

Group Element Commitment. To commit to a group element $\mathcal{X} \in \mathbb{G}_1$, using randomness $r_1,r_2 \in \mathbb{Z}_p$,

$$\mathcal{C}(\mathcal{X}) = [r_1u_{1,1} + r_2u_{2,1}, \mathcal{X} + r_1u_{1,2} + r_2u_{2,2}]_1.$$

Scalar Commitment. To commit to a scalar $x \in \mathbb{Z}_p$, using randomness $r \in \mathbb{Z}_p$,

$$\mathcal{C}'(x) = [ru_{1,1} + xu_{2,1}, ru_{1,2} + x(u_{2,2}+1)]_1$$

Proofs. Under the SXDH assumption, the two initializations of the commitment key (perfectly binding

or perfectly hiding) are indistinguishable. A Groth-Sahai proof is a pair of elements $(\pi,\theta) \in \mathbb{G}_1^{2\times 2} \times \mathbb{G}_2^{2\times 2}$. These elements are constructed to help verifying pairing relations on committed values. Being able to produce a valid pair implies knowing plaintexts verifying the appropriate relations. We will note $\langle X \rangle_1$ for a committed group element in $\mathbb{G}_1$ and $\langle x \rangle_2$ for a committed scalar x in $\mathbb{G}_2$.

Throughout the paper, we are going to encounter two kinds of relations.

- Linear Pairing Product Equations in $\mathbb{G}_1$: $[\sum_i X_i \mathcal{B}_\mathbf{i}]_T = [t]_T$ to prove that the committed group elements $[X_i]_1$ satisfy a pairing product relation with constants vector $[\mathcal{B}_\mathbf{i}]_2$ equal to a target group element in $\mathbb{G}_T$. In this particular case, the proof is composed only of $\theta \in \mathbb{G}_2^2$.
- Multi-scalar multiplications equations: $[y\mathcal{A}]_1 = [\mathcal{X}]_1$ to prove that the committed scalar in $\mathbb{G}_2$ is the discrete log of the committed element $\mathcal{X}$ committed in $\mathbb{G}_1$.

4 SIGNATURE ON RANDOMIZABLE CIPHERTEXT

A signature on randomizable ciphertexts, introduced in (Blazy et al., 2011), is a primitive which allows to sign a ciphertext, as well as all its randomizations (including the plaintext). It has been further generalized in (Hanser and Slamanig, 2014; Fuchsbauer et al., 2019) to structure-preserving signature on equivalence classes.

4.1 Definition and Security Properties

Definition 10. *A Signature on Randomizable Ciphertexts (*SRC*) scheme is composed by the seven following algorithms:*

- $\mathsf{Setup}(1^\mathfrak{K})$*: generates the global parameters* param.
- $\mathsf{KeyGen}_E(\mathsf{param})$ *generates encryption/decryption keys (*ek,dk*).*
- $\mathsf{KeyGen}_S(\mathsf{param})$ *generates verification/signing keys (*vk, sk*).*
- $\mathsf{Enc}(\mathsf{ek},\mathsf{vk},m;r)$ *outputs a ciphertext c on message $m \in \mathcal{M}$ with* ek*, using the random coins $r \in \mathcal{R}$.*
- $\mathsf{Sign}(\mathsf{sk},\mathsf{ek},c;s)$*, with random coins $s \in \mathcal{R}$, outputs a signature σ, or $\perp$ if c is not valid (w.r.t.* ek*, and possibly* vk*).*
- $\mathsf{Decrypt}(\mathsf{dk},\mathsf{vk},c)$ *decrypts c using* dk*. It outputs the plaintext, or $\perp$ if c is invalid (w.r.t.* ek*, and possibly* vk*).*
- $\mathsf{Verify}(\mathsf{vk},\mathsf{ek},c,\sigma)$ *checks whether σ is a valid signature on c, w.r.t. the public key* vk*. It outputs 1 if σ is valid, and 0 otherwise (possibly because of an invalid ciphertext c, with respect to* ek*, and possibly* vk*).*
- $\mathsf{Random}(\mathsf{vk},\mathsf{ek},c,\sigma;r')$ *outputs a ciphertext c' that encrypts the same message as c under* ek*, and a signature σ' on c'.*

A signature on ciphertexts is called ciphertext randomizable *if a randomized signature on a randomized ciphertext is statistically indistinguishable from a fresh one.*

We will denote by 0_e the neutral element in $\mathcal{R}$ that keeps the ciphertexts unchanged after randomization.

4.1.1 Extractable Signatures on Randomizable Ciphertexts

For SRC scheme,we define the following algorithm:

- $\mathsf{SEDecrypt}(\mathsf{dk},\mathsf{vk},\sigma)$, which is given a decryption key, a verification key and a signature, outputs a signature σ'.

Let us assume that there is a signature scheme $\mathcal{S}$ where Setup_S, Setup_E are the respective projections of Setup on the signature and ciphertext components, and that KeyGen_S and KeyGen_E are the associated keygen projections. For $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}_S(\mathsf{param}), m \in \mathcal{M}$, random coins $r \in \mathcal{R}, s \in \mathcal{R}$, $c = \mathsf{Enc}(\mathsf{ek},\mathsf{vk},m;r)$ and $\sigma = \mathsf{Sign}(\mathsf{sk},\mathsf{ek},c;s)$, the output $\sigma' = \mathsf{SEDecrypt}(\mathsf{dk},\mathsf{vk},\sigma)$ is a valid signature on m under vk, that is, $\mathsf{Verify}_S(\mathsf{vk},m,\sigma')$ is true.

An extractable SRC scheme $\mathcal{SC}$ allows the following. First, a user can encrypt a message m and obtain a signature σ on the ciphertext c. From (c,σ), the owner of the decryption key can now not only recover the encrypted message m, but also a signature σ' on the *message* m, using the functionality SEDecrypt. The signature σ on the ciphertext c could thus be seen as an *encryption of a signature* on the message m: for extractable signatures on ciphertexts, encryption and signing can thus be seen as commutative (see Figure 7).

On this figure, one can easily see that $\mathsf{SEDecrypt} \circ \mathsf{Sign} \circ \mathsf{Enc} = \mathsf{Sign}_S$, guarantying therefore some kind of commutativity between the signature and the encryption:

- A message m can be encrypted using random coins r (Enc);
- The signer can sign this ciphertext (Sign) and anyone can randomize the inner cipher (Random);
- A signature on the plaintext can be obtained using either dk (for SEDecrypt) or the coins r (if $\sigma(c)$ has not been randomized); the result is the same

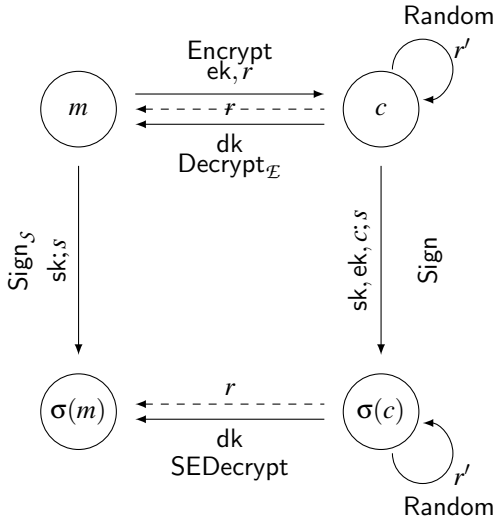

Figure 7: Extractable signatures on randomizable ciphertexts.

as a signature of m by the signer (Sign_S).

Practical Aspect of Signature on Randomizable Ciphertext. In practice, when running Random, one wants to obtain a signature on a ciphertext unlinkable to the execution (besides the tag), the final form may not have to be identical to a fresh signature, nor to be randomizable again. We call this algorithm WRandom, and the natural verification algorithm for the associated output is named WVerify.

- $\mathsf{WRandom}(\mathsf{vk},\mathsf{ek},c,\sigma;r')$ outputs a ciphertext c' that encrypts the same message as c under the public encryption key ek, and a signature σ' on c' valid under vk, with the same tag.

4.2 Instantiation with SXDH

Our construction is presented in Figure 8 and a generalization from any $k-\mathsf{MDDH}$ assumption is presented in Appendix 7. It should be noted, that in this case the WRandom algorithm drops the σ_{ek} component, while this may prevent a further randomization of the signature, this is enough for the user to get a fresh c',σ' under the same tag.

Proof Idea. In order to prove the notion of unforgeability for this instantiation, we need the following lemma from (Kiltz et al., 2015). Compared to the original signature, we add a game in order to randomize σ_{ek}. The complete proof is presented in Appendix 7.

Lemma 1 (Computational core lemma for unbounded CMA-security)**.** *For all adversaries $\mathcal{A}$, there exists a challenger $\mathcal{B}$ with $\mathcal{T}(\mathcal{A}) \approx \mathcal{T}(\mathcal{B})$ and*

$$\Pr\left[\begin{array}{l|l} & \mathbf{A},\mathbf{B} \xleftarrow{\$} \mathcal{D}_k; \\ & \mathbf{K_0},\mathbf{K_1} \xleftarrow{\$} \mathbb{Z}_p^{k+1\times k+1} \\ \tau^* \notin Q_{tag} & (\mathbf{P_0},\mathbf{P_1}) \xleftarrow{\text{def}} (\mathbf{B}^\intercal\mathbf{K_0},\mathbf{B}^\intercal\mathbf{K_1}) \in (\mathbb{Z}_p^{k\times k+1})^2 \\ \wedge b' = b & \mathbf{vk} \xleftarrow{\text{def}} ([\mathbf{P_0}]_1,[\mathbf{P_1}]_1,[\mathbf{B}]_1,\mathbf{K_0A},\mathbf{K_1A},\mathbf{A}) \\ & b \xleftarrow{\$} \{0,1\}; b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_b(.),O^*(.)}(\mathsf{vk}) \end{array}\right]$$

$$\leq \frac{1}{2} + 2Q \cdot \mathsf{Adv}^{mddh}_{\mathcal{D}_k,\mathsf{Setup}}(\mathcal{B}) + Q/q$$

where :

- $O(\tau)$ *returns* $([b\mu\mathbf{a}^\perp + \mathbf{r}^\top(\mathbf{P}_0 + \tau\mathbf{P}_1)]_1, [\mathbf{r}^\top\mathbf{B}]_1) \in (\mathbb{G}_1^{1\times(k+1)})^2$ *with* $\mu \xleftarrow{\$} \mathbb{Z}_p$, $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^k$ *and adds* τ *to* Q_{msg}. *Here,* $\mathbf{a}^\top$ *is a non-zero vector in* $\mathbb{Z}_p^{1\times(k+1)}$ *that satisfies* $\mathbf{a}^\perp\mathbf{A} = 0$.
- $O^*([\tau^*]_2)$ *returns* $[\mathbf{K}_0 + \tau^*\mathbf{K}_1]_2$. *$\mathcal{A}$ only gets a single call τ^* to O^*.*
- *Q is the number of queries $\mathcal{A}$ makes to O_b.*

5 BLIND SIGNATURE

5.1 High-level Idea

Following (Blazy et al., 2011), we can now provide a blind signature scheme using our freshly made signature scheme on randomizable ciphertexts.

The intuition is that after recovering the signature $\sigma_{\mathsf{SRC}} = \mathsf{Sign}(\mathsf{sk},\mathsf{ek},c = \mathsf{Enc}(\mathsf{ek},\mathsf{vk},m;r));s)$, a user can compute $\sigma = \mathsf{WRandom}(\mathsf{vk},\mathsf{ek},c,\sigma_{\mathsf{SRC}},-r)$, which is a valid signature on $(1,m)$. This gives him a signature on m.

The tricky part is now to achieve blindness. While fully randomizing the signature (as in (Blazy et al., 2011)) would be the best solution, this is not possible with the signature from (Kiltz et al., 2015) that we consider in this paper. Instead, following Fischlin's idea in (Fischlin, 2006), we add a complete zero-knowledge proof of the knowledge of σ. Moreover, due to the fact that we cannot extract a scalar τ from a commitment, we need to commit to τ in $\mathbb{G}_2$, and to the original $\sigma_3 = [\tau\sigma_2]_1$.

A high-level idea of the process (pre-blinding) is explained in the figure 9.

5.2 Overview of the Construction

In this section, we instantiate algorithms from the definition 9 of a blind signature $\mathcal{BS} = (\mathsf{BSSetup},\mathsf{BSKeyGen},\langle\mathcal{S},\mathcal{U}\rangle,\mathsf{Verify})$.

- $\mathsf{BSSetup}(\mathfrak{K})$ calls the setup algorithm of the SRC scheme. It outputs param.
- $\mathsf{BSKeyGen}(\mathsf{param})$ calls the KeyGen_S algorithm of the SRC scheme. It is run by the server $\mathcal{S}$.

$\underline{\mathsf{Setup}(1^{\mathfrak{K}})}$:
Return param

$\underline{\mathsf{KeyGen}_E(\mathsf{param})}$:
$\mathsf{dk} = h \xleftarrow{\$} \mathbb{Z}_p, \mathsf{ek} = [1,h]_1 \in \mathbb{G}_1^2$
Return ek, dk

$\underline{\mathsf{KeyGen}_S(\mathsf{param})}$:
$\mathbf{A}, \mathbf{B} \xleftarrow{\$} \mathcal{D}_2, \mathbf{X} \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}$
$\mathbf{K}_0, \mathbf{K}_1 \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}$,
$\mathbf{C} = \mathbf{K}\mathbf{A} \in \mathbb{Z}_p^{k+1}$
$(\mathbf{C}_0, \mathbf{C}_1) = (\mathbf{K}_0\mathbf{A}, \mathbf{K}_1\mathbf{A}) \in \mathbb{Z}_p^2 \times \mathbb{Z}_p^2$
$(\mathbf{P}_0, \mathbf{P}_1) = (\mathbf{B}^\top\mathbf{K}_0, \mathbf{B}^\top\mathbf{K}_1) \in \mathbb{Z}_p^{1\times 2} \times \mathbb{Z}_p^{1\times 2}$
$\mathsf{sk} = (\mathbf{K}, [\mathbf{P}_0]_1, [\mathbf{P}_1]_1, [\mathbf{B}]_1)$
$\mathsf{vk} = ([\mathbf{C}_0]_2, [\mathbf{C}_1]_2, [\mathbf{C}_2]_2, [\mathbf{A}]_2)$
Return vk, sk

$\underline{\mathsf{Enc}(\mathsf{ek}, \perp, m)}$:
$r \xleftarrow{\$} \mathbb{Z}_p$, $c = [r, rh+m]_1$
Return c

$\underline{\mathsf{Sign}(\mathsf{sk}, \mathsf{ek}, c; \mathbf{s})}$:
$\mathbf{s} \xleftarrow{\$} \mathbb{Z}_p, \tau \xleftarrow{\$} \mathbb{Z}_p$
$\sigma_1 = [(1, c^\top)\mathbf{K} + s(\mathbf{P}_0 + \tau\mathbf{P}_1)]_1 \in \mathbb{G}_1^{1\times 2}$
$\sigma_{\mathsf{ek}} = [(0, \mathsf{ek}^\top)\mathbf{K} + s(\mathbf{P}_0 + \tau\mathbf{P}_1)]_1 \in \mathbb{G}_1^{1\times 2}$
$\sigma_2 = [s\mathbf{B}^\top]_1 \in \mathbb{G}_1^{1\times 2}$, $\sigma_\tau = \tau \in \mathbb{Z}_p$
Return $\sigma = (\sigma_1, \sigma_{\mathsf{ek}}, \sigma_2, \sigma_\tau)$

$\underline{\mathsf{Verify}(\mathsf{vk}, \mathsf{ek}, c, \sigma)}$
Check whether:
$[\sigma_1\mathbf{A}^\top]_T = [(1, c^\top)\mathbf{C}^\top + \sigma_2(\mathbf{C}_0^\top + \sigma_\tau\mathbf{C}_1^\top)]_T$
$[\sigma_{\mathsf{ek}}\mathbf{A}^\top]_T = [(0, \mathsf{ek}^\top)\mathbf{C}^\top + \sigma_2(\mathbf{C}_0^\top + \sigma_\tau\mathbf{C}_1^\top)]_T$

$\underline{\mathsf{Decrypt}(\mathsf{dk}, c)}$:
Return $[m]_1 = c_2 / c_1^{\mathsf{dk}} = [rh + m - rh]_1$

$\underline{\mathsf{WRandom}(\mathsf{vk}, \mathsf{ek}, c, \sigma)}$:
$r' \xleftarrow{\$} \mathbb{Z}_p$
Compute $c' = c + [r', r'h]_1$ and update:
$\sigma_1' = \sigma_1 + \mathbf{r}'^\top\sigma_{\mathsf{ek}}$, $\sigma_2' = (1 + \mathbf{r}')\sigma_2$
Return c' and $\sigma' = (\sigma_1', \sigma_2', \sigma_\tau)$

$\underline{\mathsf{WVerify}(\mathsf{vk}, \mathsf{ek}, c', \sigma')}$
Check whether:
$[\sigma_1\mathbf{A}^\top]_T = [(1, c^\top)\mathbf{C}^\top + \sigma_2(\mathbf{C}_0^\top + \sigma_\tau\mathbf{C}_1^\top)]_T$

Figure 8: SXDH Instantiation of Constant Size SRC.

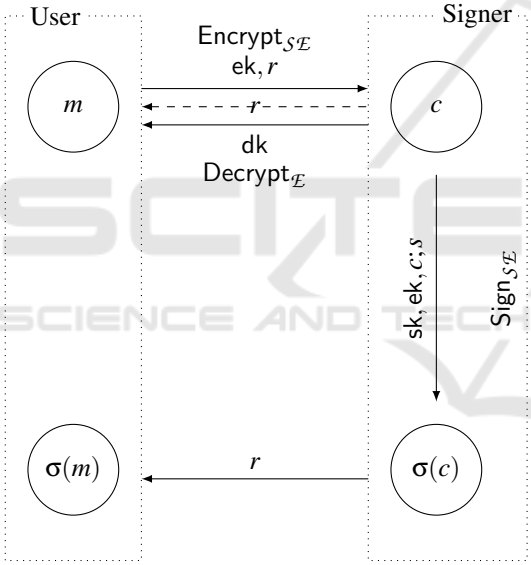

Figure 9: High level Blind Signature.

- Signature Issuing is described by Figure 10. It is an interactive protocol in which the user $\mathcal{U}$ has to encrypt its message and send it to the signer $\mathcal{S}$ which in return computes a SRC signature σ_{SRC}. Having received σ_{SRC}, $\mathcal{U}$ checks its validity using vk. If it is valid, he extracts a signature on m by using the WRandom algorithm.
  Following Fischlin's framework, the blind signature consists in the NIZK proof π guaranteeing $\mathcal{U}$ has the elements satisfying the signature verification equations w.r.t to vk and m.
- $\mathsf{Verify}(\mathsf{vk}, \sigma)$ calls the NIZK proof verification algorithm. If the proof is valid, the signature is valid and it outputs 1, otherwise 0.

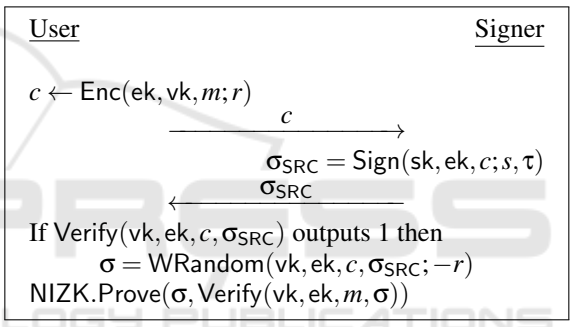

Figure 10: Blind Signature.

5.3 Security Results and Proofs Ideas

Theorem 1. *Our construction achieves the blindness property under the* SXDH *assumption.*

Proof Idea. Every value sent by the user is either encrypted (committed) or a Zero-Knowledge proof. The flow in the SRC hides the target message, while the Zero-Knowledge proof hides both the tag and the randomness used in the signature. Hence, under the IND-CPA property of the encryption and the Zero-Knowledge property of the proof, the scheme achieves blindness.

Theorem 2. *Our construction is unforgeable under the* XDH *in* $\mathbb{G}_1$ *and* $\mathcal{D}_1 - \mathsf{KerMDDH}$ *in* $\mathbb{G}_2$.

Proof Idea. Following the original proof from (Kiltz et al., 2015), we proceed via successive games. First, we guess whether the adversary picks a fresh tag for the forgery or reuse one of the signature he received, and we simulate all the other answers. Then, we show that for a given tag, the signature becomes a valid 1-time signature.

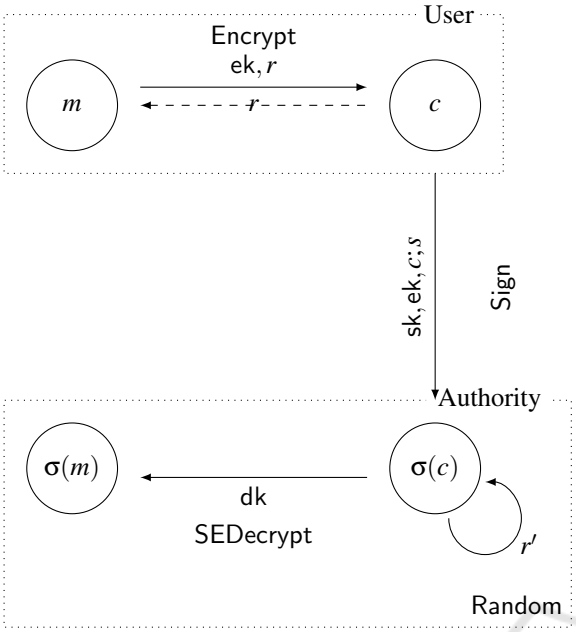

Figure 11: Viewing an SRC as a e-voting solution.

6 APPLICATION TO E-VOTING WITH RECEIPT FREENESS

In voting schemes, anonymity is a crucial property, since nobody should be able to learn the content of someone else's vote. This can be achieved using homomorphic encryption schemes. Basically, each user will compute his vote v_i, commit it into c_i, and then, through the homomorphic property of the encryption scheme, the voting center will compute $f(c_1, \ldots, c_n)$ and open it to solely obtain the global result of the election.

However, this does not address the problem of *vote sellers*: a voter may sell his vote and then reveal/prove the content of his encrypted vote to the buyer. He could do so by simply revealing the randomness used when encrypting the vote, which allows to verify that a claimed message was encrypted. To avoid this, we need to be able to randomize encryption, and adapt the signature accordingly. But before doing so, the voting center randomizes c into c' (which cannot be opened by the voter anymore since he no longer knows the random coins) and then proves (in a non-transferable way) that c and c' contain the same plaintext. The used proof is thus a designated-verifier zero-knowledge proof. Finally, after receiving c' and being convinced by the proof, the voter signs c'.

Our SRC allows to avoid those extra interactions: a voter simply encrypts his vote v as c and makes a signature σ on c. The voting center can now consistently randomize both c and σ as c' and σ', so that the randomness used in c' is unknown to the signer, who is however guaranteed that the vote was not modified by the voting center because of the unforgeability notion for SRC. We have thus constructed a non-interactive *receipt-free* voting scheme.[2]

Since our SRC candidates use both randomizable and homomorphic encryption schemes, classical techniques for voting schemes with homomorphic encryption and threshold decryption can be used (Baudron et al., 2001): there is no risk for the signature on the ciphertext to be converted into a signature on the plaintext if the board of authorities uses the decryption capability on the encrypted tally only.

The size of the ballot is only 6 group elements and a scalar in the instantiation with using ElGamal, independently from the number of check boxes in the vote.

7 CONCLUSION

In this paper, we answer to an open question raised during Eurocrypt'19, by providing the first round-optimal constant-size blind signature in the standard model based on a classical assumption. Towards this goal, we give the first constant-size signature on randomized ciphertext as a side contribution. This signature is based upon a variant of structure-preserving signature.

Our blind signature scheme offers several advantages: in addition to being constant-size in terms of interaction (rather than asking the user to send a first flow linear in the size of the message to be signed) and to being built under well-studied security assumptions, the signature kept on the user side remains very compact (4 group elements and 1 scalar), which is of critical importance for constrained systems.

One way to improve the final size of the signature would be to find a randomizable structure-preserving signature, in order to get rid of the zero-knowledge proof. More generally, finding the lower bound for a blind signature in the standard model remains an open question. Finally, following the study initiated in (Rückert, 2010), future work could include finding a post-quantum version of our scheme (building on a lattice- or code-based assumption).

[2]We want to stress, that as the secret signing key is **secret**, users cannot craft fake encrypted ballot in a way that would allow two of them to be combined in a third one. In addition, as the signer changes the tag with every signing query, this combination would be impossible.

ACKNOWLEDGMENTS

We would like thanks the rewievers for detailed comments. This work was supported in part by the French ANR projects IDFIX (ANR-16-CE39-0004) and CryptiQ (ANR-18-CE39-0015).

REFERENCES

Abe, M. (2001). A secure three-move blind signature scheme for polynomially many signatures. In Pfitzmann, B., editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 136–151. Springer, Heidelberg.

Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., and Ohkubo, M. (2010). Structure-preserving signatures and commitments to group elements. In Rabin, T., editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 209–236. Springer, Heidelberg.

Ateniese, G., Camenisch, J., Hohenberger, S., and de Medeiros, B. (2005). Practical group signatures without random oracles. Cryptology ePrint Archive, Report 2005/385.

Baldimtsi, F. and Lysyanskaya, A. (2013). On the security of one-witness blind signature schemes. In Sako, K. and Sarkar, P., editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 82–99. Springer, Heidelberg.

Baudron, O., Fouque, P.-A., Pointcheval, D., Stern, J., and Poupard, G. (2001). Practical multi-candidate election system. In Kshemkalyani, A. D. and Shavit, N., editors, *20th ACM PODC*, pages 274–283. ACM.

Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., and Shacham, H. (2009). Randomizable proofs and delegatable anonymous credentials. In Halevi, S., editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 108–125. Springer, Heidelberg.

Bellare, M. and Rogaway, P. (1993). Random oracles are practical: A paradigm for designing efficient protocols. In Denning, D. E., Pyle, R., Ganesan, R., Sandhu, R. S., and Ashby, V., editors, *ACM CCS 93*, pages 62–73. ACM Press.

Blazy, O., Fuchsbauer, G., Pointcheval, D., and Vergnaud, D. (2011). Signatures on randomizable ciphertexts. In Catalano, D., Fazio, N., Gennaro, R., and Nicolosi, A., editors, *PKC 2011*, volume 6571 of *LNCS*, pages 403–422. Springer, Heidelberg.

Blazy, O., Pointcheval, D., and Vergnaud, D. (2012a). Compact round-optimal partially-blind signatures. In Visconti, I. and Prisco, R. D., editors, *SCN 12*, volume 7485 of *LNCS*, pages 95–112. Springer, Heidelberg.

Blazy, O., Pointcheval, D., and Vergnaud, D. (2012b). Round-optimal privacy-preserving protocols with smooth projective hash functions. In Cramer, R., editor, *TCC 2012*, volume 7194 of *LNCS*, pages 94–111. Springer, Heidelberg.

Boneh, D., Boyen, X., and Shacham, H. (2004). Short group signatures. In Franklin, M., editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg.

Brands, S. (1994). Untraceable off-line cash in wallets with observers (extended abstract). In Stinson, D. R., editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 302–318. Springer, Heidelberg.

Camenisch, J., Hohenberger, S., and Lysyanskaya, A. (2005). Compact e-cash. In Cramer, R., editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 302–321. Springer, Heidelberg.

Camenisch, J. and Lysyanskaya, A. (2001). An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Pfitzmann, B., editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg.

Chaum, D. (1982). Blind signatures for untraceable payments. In Chaum, D., Rivest, R. L., and Sherman, A. T., editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA.

Chaum, D., Fiat, A., and Naor, M. (1990). Untraceable electronic cash. In Goldwasser, S., editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 319–327. Springer, Heidelberg.

ElGamal, T. (1984). A public key cryptosystem and a signature scheme based on discrete logarithms. In Blakley, G. R. and Chaum, D., editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, Heidelberg.

Escala, A., Herold, G., Kiltz, E., Ràfols, C., and Villar, J. (2013). An algebraic framework for Diffie-Hellman assumptions. In Canetti, R. and Garay, J. A., editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg.

Fischlin, M. (2006). Round-optimal composable blind signatures in the common reference string model. In Dwork, C., editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 60–77. Springer, Heidelberg.

Fuchsbauer, G. (2011). Commuting signatures and verifiable encryption. In Paterson, K. G., editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 224–245. Springer, Heidelberg.

Fuchsbauer, G., Hanser, C., and Slamanig, D. (2019). Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *Journal of Cryptology*, 32(2):498–546.

Fuchsbauer, G., Pointcheval, D., and Vergnaud, D. (2009). Transferable constant-size fair e-cash. In Garay, J. A., Miyaji, A., and Otsuka, A., editors, *CANS 09*, volume 5888 of *LNCS*, pages 226–247. Springer, Heidelberg.

Fujioka, A., Okamoto, T., and Ohta, K. (1993). A practical secret voting scheme for large scale elections. In Seberry, J. and Zheng, Y., editors, *Advances in Cryptology — AUSCRYPT '92*, pages 244–251, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ghadafi, E. (2017). Efficient round-optimal blind signatures in the standard model. In Kiayias, A., editor, *FC 2017*, volume 10322 of *LNCS*, pages 455–473. Springer, Heidelberg.

Goldwasser, S. and Micali, S. (1984). Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299.

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

Groth, J. and Sahai, A. (2008). Efficient non-interactive proof systems for bilinear groups. In Smart, N. P., editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Heidelberg.

Hanser, C. and Slamanig, D. (2014). Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In Sarkar, P. and Iwata, T., editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 491–511. Springer, Heidelberg.

Hauck, E., Kiltz, E., and Loss, J. (2019). A modular treatment of blind signatures from identification schemes. In Ishai, Y. and Rijmen, V., editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 345–375. Springer, Heidelberg.

Kiltz, E., Pan, J., and Wee, H. (2015). Structure-preserving signatures from standard assumptions, revisited. In Gennaro, R. and Robshaw, M. J. B., editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 275–295. Springer, Heidelberg.

Okamoto, T. (1993). Provably secure and practical identification schemes and corresponding signature schemes. In Brickell, E. F., editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 31–53. Springer, Heidelberg.

Okamoto, T. (2006). Efficient blind and partially blind signatures without random oracles. In Halevi, S. and Rabin, T., editors, *TCC 2006*, volume 3876 of *LNCS*, pages 80–99. Springer, Heidelberg.

Okamoto, T. and Ohta, K. (1992). Universal electronic cash. In Feigenbaum, J., editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 324–337. Springer, Heidelberg.

Pointcheval, D. and Stern, J. (2000). Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396.

Rückert, M. (2010). Lattice-based blind signatures. In Abe, M., editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 413–430. Springer, Heidelberg.

Schröder, D. and Unruh, D. (2012). Security of blind signatures revisited. In Fischlin, M., Buchmann, J., and Manulis, M., editors, *PKC 2012*, volume 7293 of *LNCS*, pages 662–679. Springer, Heidelberg.

APPENDIX

PROOFS OF OUR SRC CONSTRUCTIONS

Unforgeability Proof

In this section, we present the proof of unforgeability for our SRC construction. As we use a SPS of Kiltz *et.al*, our proof is an adaptation from their.

Proof. **Game 1.** In this game, we modify the verification algorithm. The pairing equation became : $[\sigma_1 \cdot 1]_T = [(1, \mathbf{m}^\top \mathbf{K}) \cdot 1 + \sigma_2 \cdot (\mathbf{K}_0 + \tau \mathbf{K}_1)]_T$.

Suppose $[\sigma_2 \cdot \sigma_4]_T = [\sigma_3]_T$, we note that:

$$
\begin{aligned}
& [\sigma_1 \cdot \mathbf{A}]_T = [(1, \mathbf{m}^\top) \cdot \mathbf{C} + \sigma_2 \cdot \mathbf{C_0} + \sigma_3 \cdot \mathbf{C_1}]_T \\
\iff & [\sigma_1 \cdot \mathbf{A}]_T = [(1, \mathbf{m}^\top) \cdot \mathbf{KA} + \sigma_2 \cdot \mathbf{K}_0\mathbf{A} + \sigma_3 \cdot \mathbf{K}_1\mathbf{A}]_T \\
\iff & [\sigma_1 \cdot 1]_T = [(1, \mathbf{m}^\top) \cdot \mathbf{K} + \sigma_2 \cdot \mathbf{K}_0 + \sigma_3 \cdot \mathbf{K}_1]_T \\
\iff & [\sigma_1 \cdot 1]_T = [(1, \mathbf{m}^\top) \cdot \mathbf{K} + \sigma_2 \cdot (\mathbf{K}_0 + \tau \mathbf{K}_1)]_T
\end{aligned}
$$

The value $\sigma_1 - ([(1, \mathbf{m}^\top)\mathbf{K}]_1 + \sigma_2 \mathbf{K}_0 + \sigma_3 \mathbf{K}_1) \in \mathbb{G}_1^{1\times(k+1)}$ is a non-zero vector in the kernel of $\mathbf{A}$, which is hard to be computed under the $\mathcal{D}_k$-KerMDDH assumption in $\mathbb{G}_2$. This means that: $|\mathsf{Adv}_0 - \mathsf{Adv}_1| \leq \mathsf{Adv}^{k-\mathsf{MDDH}}_{\mathcal{D}_k,\mathsf{Setup}}(\mathcal{B}_0)$.

Game 2. Let $\tau_1, ..., \tau_Q$ the randomly chosen tags in the Q queries made by the adversary $\mathcal{A}$ to OSign. We abort if $\tau_1, ..., \tau_Q$ are not all distinct. So, we obtain: $\mathsf{Adv}_1 \geq \mathsf{Adv}_1 - Q^2/2q$.

Game 3. We define $\tau_{Q+1} \stackrel{\text{def}}{=} \tau^*$. Now, pick $\imath^* \xleftarrow{\$} [Q+1]$ and abort if i^* is not the smallest index of i^* for which $\tau^* = \tau_i$. In the rest of the proof, we focus on the case we do not abort *i.e* $\tau^* = \tau_{i^*}$ and $\tau_1, ..., \tau_{i^*-1}$ are all different from τ^*. Given τ, OSign can check whether τ^* equals τ. For the rest $i^* - 1$ queries, answer NO, and starting from the i^*'th query, we know τ^*.

Hence, we have: $\mathsf{Adv}_3 \geq \frac{1}{Q+1}\mathsf{Adv}_2$.

Game 4.1. We switch OSign to OSign_1^*:

OSign_1^*:	OSign_2^*
$\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p; \tau \xleftarrow{\$} \mathbb{Z}_p; \mu_1 \xleftarrow{\$} \mathbb{Z}_p$	
$\sigma_1 \stackrel{\text{def}}{=} [(1, C_1, C_2)\mathbf{K} + \mu_1 \mathbf{a}^\perp + \mathbf{r}^\top(\mathbf{P_0} + \tau \mathbf{P_1})]_1 \in \mathbb{G}_1^{1\times 2}$	
$\sigma_{\mathsf{ek}} \stackrel{\text{def}}{=} [(0, 1, \mathsf{ek})\mathbf{K} + \mathbf{r}^\top(\mathbf{P_0} + \tau \mathbf{P_1})]_1 \in \mathbb{G}_1^{1\times 2}$	
$\sigma_{\mathsf{ek}} \stackrel{\text{def}}{=} [(0, 1, \mathsf{ek})\mathbf{K} + \mu_2 \mathbf{a}^\perp + \mathbf{r}^\top(\mathbf{P_0} + \tau \mathbf{P_1})]_1 \in \mathbb{G}_1^{1\times 2}$	
$\sigma_2 \stackrel{\text{def}}{=} [\mathbf{r}^\intercal \mathbf{B}^\intercal]_1 \in \mathbb{G}_1^{1\times 2}$, $\sigma_\tau \stackrel{\text{def}}{=} \tau \in \mathbb{Z}_p$	
$\mathtt{Return}(\sigma_1, \sigma_{\mathsf{ek}}, \sigma_2, \sigma_\tau)$	

Here, $\mathbf{a}^\perp \in^{1\times(k+1)}$ is non-zero vector in the kernel of $\mathbf{A}$ such that: $\mathbf{a}^\perp \mathbf{A} = 0$. We will apply Lemma 1 in order to show: $|\mathsf{Adv}_3 - \mathsf{Adv}_{4.1}| \geq 2Q\mathsf{Adv}^{mddh}_{\mathcal{D}_k,\mathsf{Setup}}(\mathcal{B}_1) + Q/q$. We pick $\mathbf{K}$ and we use O_b to simulate either OSign or OSign_1^* and O^* to simulate Verify as follow:

- For the i'th signing query where $i \neq i^*$, we query O_b at $\tau \xleftarrow{\$} \mathbb{Z}_p$ to obtain: $(\sigma_1', \sigma_2) \stackrel{\text{def}}{=} ([b\mu_1 \mathbf{a}^\perp + \mathbf{r}^\top(\mathbf{P}_0 + \tau \mathbf{P}_1)]_1, [\mathbf{r}^\top \mathbf{B}^\top]_2)$ with $b \xleftarrow{\$} \{0,1\}$ and we return: $(\sigma_1 \stackrel{\text{def}}{=} [(1, \mathbf{m}\top)\mathbf{K}]_1 \cdot \sigma_1', \sigma_{\mathsf{ek}}, \sigma_2, \sigma_3, \sigma_4)$.
- For the i^*'th query, where $i^* \leq Q$, we run Sign honestly.
- For Verify*, we will query O^* on τ^* to get $[\mathbf{K}_0 + \tau^* \mathbf{K}_1]_2$. The latter is sufficient to simulate Verify*

query by computing $[\sigma \cdot (\mathbf{K}_0 + \tau^* \mathbf{K}_1)]_T$.
So, we are allowed to build a distinguisher for Lemma 1.

Game 4.2.

We have to modify σ_{ek} in the same way as before:
In order to apply lemma 1, we simulate oracles:

- For $i \neq i^*$, we have:

$$(\sigma'_1, \sigma'_{\mathsf{ek}}, \sigma_2) \stackrel{\text{def}}{=} ([b\mu_1 \mathbf{a}^\perp + \mathbf{r}^\top (\mathbf{P}_0 + \tau \mathbf{P}_1)]_1, [b\mu_2 \mathbf{a}^\perp + \mathbf{r}^\top (\mathbf{P}_0 + \tau \mathbf{P}_1)]_1, [\mathbf{r}^\top \mathbf{B}^\top]_2).$$

  We return: $(\sigma_1 \stackrel{\text{def}}{=} [(1, \mathbf{m}\top)\mathbf{K}]_1 \cdot \sigma'_1, \sigma_{\mathsf{ek}} \stackrel{\text{def}}{=} [(1, \mathbf{m}\top)\mathbf{K}]_1 \cdot \sigma'_{\mathsf{ek}}, \sigma_2, \sigma_4)$.
- For the challenge, we run Sign honestly.

Game 5. In this game we modify the matrix $\mathbf{K}$ in Setup algorithm. We switch $\mathbf{K}$ by $\mathbf{K} = \mathbf{K}' + \mathbf{u}\mathbf{a}^\perp$ with $\mathbf{K} \xleftarrow{\$} \mathbb{Z}_p^{(n+1)\times(k+1)}$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{n+1}$. $\mathbf{u}\mathbf{a}^\perp$ is masked by a uniform matrix $\mathbf{K}'$, $\mathbf{K}$ is still uniformly random. Thus, Game 5 and Game 4.2 are identical, so we have: $\mathsf{Adv}_5 = \mathsf{Adv}_{4.2}$.

Now, we have to bound Adv_5.

- $C = \mathbf{K}\mathbf{A} = (\mathbf{K}' + \mathbf{u}\mathbf{a}^\perp)\mathbf{A} = \mathbf{K}'\mathbf{A}$. $\mathbf{u}$ is completely hide.
- OSign_2^* on $(C_1, C_2, \mathsf{ek}, \tau)$ for $\tau \neq \tau^*$ returns: $(1, C_1, C_2)(\mathbf{K}' + \mathbf{u}\mathbf{a}^\perp) + \mu_1 \mathbf{a}^\perp$ and $(0, 1, \mathsf{ek}) + (\mathbf{K}' + \mathbf{u}\mathbf{a}^\perp) + \mu_2 \mathbf{a}^\perp$. $\mathbf{u}$ is hide since outputs is identically distributed as: $(1, C_1, C_2)\mathbf{K}' + \mu_1 \mathbf{a}^\perp$ and $(0, 1, \mathsf{ek})\mathbf{K} + \mu_2 \mathbf{a}^\perp$.
- OSign_2^* on τ^* leaks $(1, C_1, C_2)\mathbf{K}' + \mu_1 \mathbf{a}^\perp$ and $(1, \mathsf{ek})\mathbf{K} + \mu_2 \mathbf{a}^\perp$, which is captured by $(1, C_1, C_2)\mathbf{u}$ and by $(0, 1, \mathsf{ek})\mathbf{u}$.

The adversary has to compute correctly: $(1, C_1^*, C_2^*)(\mathbf{K}' + \mathbf{u}\mathbf{a}^\perp)$ and $(0, 1, \mathsf{ek})(\mathbf{K}' + \mathbf{u}\mathbf{a}^\perp)$ to convince Verify^* to accept a signature σ^* on $(1, C_1^*, C_2^*)$ and on $(0, 1, \mathsf{ek}^*)$. As $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{k+1}$, $(1, C_1^*, C_2^*)\mathbf{u}$ and $(0, 1, \mathsf{ek}^*)\mathbf{u}$ are in $\mathbb{Z}_p$.

Given $(1, C_1^*, C_2 *)\mathbf{u}$ and $(0, 1, \mathsf{ek}^*)$for any adaptively chosen $(C_1^*, C_2^*) \neq (C_1, C_2)$ and $\mathsf{ek}^* \neq \mathsf{ek}$, $(1, C_1^*, C_2^*)\mathbf{u}$ and $(0, 1, \mathsf{ek}^*)\mathbf{u}$ are uniformly random over $\mathbb{Z}_p$ from the adversary's view-point. □

Proof of Blindness

Proof. We proceed via a series of games.

Game 0. This is the Blindness experiment from figure 5: $\mathsf{Adv}_0 = \mathsf{Adv}^{Blind}_{\mathcal{BS}}$.

In this game, we don't modify existing algorithms. During the different games, $\mathcal{A}$ proposed two messages m_0 and m_1 to the challenger.

Game 1. Both proofs are replaced by simulated ones thanks to the Zero-Knowledge property of the Groth-Sahai proof system. We replace the NIZK algorithm by a simulated one. Thus, we have:

$$\mathsf{Adv}_0 \leq \mathsf{Adv}_1 + \mathsf{Adv}^{\mathsf{SXDH}}_{\mathcal{GS}}$$

Game 2. Both encryptions of m_0 and m_1 are replaced by random group elements *i.e* the challenger encrypts a random group element noted v instead of m_b. By the indistinguishability property of the ElGamal encryption scheme, we have:

$$\mathsf{Adv}_1 \leq \mathsf{Adv}_2 + \mathsf{Adv}^{\mathsf{SXDH}}_{\mathcal{EG}}$$

In this game, $\mathcal{A}$ is given absolutely no information on m_0 and m_1 therefore $\mathsf{Adv}_2 = 0$. □

GENERIC CONSTRUCTION

We present the generalization of our SXDH constant size SRC. The construction is derived from (Kiltz et al., 2015), and proofs have to be tweaked the same way to achieve unforgeability for equivalence classes.

$\underline{\mathsf{Setup}(1^{\hat{\kappa}})}$:Return param

$\underline{\mathsf{KeyGen}(\mathsf{param})}$:
$\mathsf{ek} = [\mathbf{H}]_1 \xleftarrow{\$} \mathbb{G}_1^{(k+1)\times k}$, $\mathsf{dk} = H \in \mathbb{Z}_p^{(k+1)\times k}$
Return ek, dk

$\underline{\mathsf{KeyGen}(\mathsf{param})}$:
$\mathbf{A}, \mathbf{B} \xleftarrow{\$} \mathcal{D}_{k+1}, \mathbf{X} \xleftarrow{\$} \mathbb{Z}_p^{k+1\times k+1}$, $\mathbf{K}_0, \mathbf{K}_1 \xleftarrow{\$} \mathbb{Z}_p^{k+1\times k+1}$
$\mathbf{C} = \mathbf{K}\mathbf{A} \in \mathbb{Z}_p^{k+1}$
$(\mathbf{C}_0, \mathbf{C}_1) = (\mathbf{K}_0\mathbf{A}, \mathbf{K}_1\mathbf{A}) \in (\mathbb{Z}_p^{k+1})^2$
$(\mathbf{P}_0, \mathbf{P}_1) = (\mathbf{B}^\top \mathbf{K}_0, \mathbf{B}^\top \mathbf{K}_1) \in (\mathbb{Z}_p^{1\times(k+1)})^2$
$\mathsf{sk} = (\mathbf{K}, [\mathbf{P}_0]_1, [\mathbf{P}_1]_1, [\mathbf{B}]_1)$
$\mathsf{vk} = ([\mathbf{C}_0]_2, [\mathbf{C}_1]_2, [\mathbf{C}_2]_2, [\mathbf{a}]_2)$
Return vk, sk

$\underline{\mathsf{Enc}(\mathsf{ek}, \perp, m)}$:
$r \xleftarrow{\$} \mathbb{Z}_p$, Return $c = [r, rh + M]_1$

$\underline{\mathsf{Sign}(\mathsf{sk}, \mathsf{ek}, c; \vec{s})}$:
$\mathbf{s} \xleftarrow{\$} \mathbb{Z}_p^k, \tau \xleftarrow{\$} \mathbb{Z}_p$
$\sigma_1 = [(1, c^\top)\mathbf{K} + \mathbf{s}^\top (\mathbf{P}_0 + \tau \mathbf{P}_1)]_1 \in \mathbb{G}_1^{1\times k+1}$
$\sigma_{\mathsf{ek}} = [(0, \mathbf{H}^\top)\mathbf{K} + \mathbf{s}^\top (\mathbf{P}_0 + \tau \mathbf{P}_1)]_1 \in \mathbb{G}_1^{k\times k+1}$
$\sigma_2 = [\mathbf{s}^\top \mathbf{B}^\top]_1 \in \mathbb{G}_1^{1\times k+1}$, $\sigma_\tau = \tau \in \mathbb{Z}_p$
Return $\sigma = (\sigma_1, \sigma_{\mathsf{ek}}, \sigma_2, \sigma_\tau)$

$\underline{\mathsf{Decrypt}(\mathsf{ek}, c)}$:
Return $[m]_1 = c_2 / c_1^{\mathsf{dk}} = [rh + m - rh]_1$

$\underline{\mathsf{Verify}(\mathsf{vk}, \mathsf{ek}, c, \sigma)}$
Check whether:
$[\sigma_1 \mathbf{a}^\top]_T = [(1, c^\top)\mathbf{C}^\top + \sigma_2(\mathbf{C}_0^\top + \sigma_\tau \mathbf{C}_1^\top)]_T$
$[\sigma_{\mathsf{ek}} \mathbf{a}^\top]_T = [(0, \mathsf{ek}^\top)\mathbf{C}^\top + \sigma_2(\mathbf{C}_0^\top + \sigma_\tau \mathbf{C}_1^\top)]_T$

Figure 12: Generic Construction of Constant Size SRC.