This is the Full Version of the Extended Abstract that appears
in the Proceedings of 8th Conference on Security and Cryptography for Networks (SCN 2012)
(5 – 7 september 2012, Amalfi, Italy)
I. Visconti and R. De Prisco Eds., Springer-Verlag, LNCS ????, pages ???–???.

Compact Round-Optimal Partially-Blind Signatures

Olivier Blazy, David Pointcheval, and Damien Vergnaud
{olivier.blazy, david.pointcheval, damien.vergnaud}@ens.fr

ENS, Paris, France *

Abstract Partially-blind signatures find many applications in the area of anonymity, such as in e-cash or e-voting systems. They extend classical blind signatures, with a signed message composed of two parts: a public one (common to the user and the signer) and a private one (chosen by the user, and blindly signed). The signer cannot link later the message-signature to the initial interaction with the user, among other signatures on messages with the same public part.

This paper presents a one-round partially-blind signature which achieves perfect blindness in the standard model using a Common Reference String, under classical assumptions: CDH and DLin assumptions in symmetric groups, and similar ones in asymmetric groups. This scheme is more efficient than the previous ones: reduced round complexity and communication complexity, but still weaker complexity assumptions. A great advantage is also to end up with a standard Waters signature, which is quite short.

In addition, in all the previous schemes, the public part required a prior agreement between the parties on the public part of the message before running the blind signature protocol. Our protocol does not require such pre-processing: the public part can be chosen by the signer only.

Our scheme even allows multiple messages provided from independent sources to be blindly signed. These messages can either be concatenated or aggregated by the signer, without learning any information about them, before returning the blind signature to the recipient. For the aggregation (addition of the messages), we provide a new result, of independent interest, about the Waters hash function over non binary-alphabets.

1 Introduction

Blind signatures were proposed by Chaum in 1982 [8]: they are an interactive signature scheme between a user and a signer, in a way that the signed message, and even the resulting signature, are unknown to the signer, this is the *blindness* property. More precisely, if the signer runs several executions of the protocol that led to several message-signature pairs, he cannot link back any pair to a specific execution: the view of the signer is unlinkable to the resulting message-signature pair. This unlinkability can either be computational, we then talk about *computational blindness*, or perfect, we then talk about *perfect blindness*. In addition, they guarantee some kind of unforgeability for the signer, which has been formalized in [18] to cope with e-cash properties: the user cannot produce more message-signature pairs (coins) than the number of interactions (withdrawals).

There have been several highly interactive schemes (like [17]), but Fischlin [10] gave a generic construction of *round-optimal* blind signatures. Recent schemes have instantiated this construction, the user obtains an actual signature on the message, of which he proves knowledge [1, 11] or can simply randomize it to make it unlinkable [4, 5]. In the latter case, the blind signature has the same format as the underlying signatures and, in addition to being round-optimal, is thus short. Our construction, like this last one produced a simple (randomized) Waters signature on the message m so two group elements and a scalar m under basic assumptions DLin, where [1] uses less standard assumption SXDH and ADH-CDH, and around 38 elements in $\mathbb{G}_1$ and 34 in $\mathbb{G}_2$ for the final signature because of the required proofs of knowledge. [12] presented a round-optimal blind signature without CRS but less efficient than the construction relying on the Common Reference String.

A loophole in standard blind signatures was detailed by Abe and Okamoto [3]: the signer has no control over the signed messages (except in some sense the unforgeability which limits their number). In e-cash schemes, we want the bank to sign a coin (a random, and thus unknown, serial number), but with a specific expiration date. Partially-Blind Signatures proposed by Abe and Fujisaki [2] solve this problem, by allowing

* CNRS – UMR 8548 and INRIA – EPI Cascade, Université Paris Diderot

the user and the signer to agree on a predetermined piece of information which must be included in the final signed message.

Recently, in [19], Seo and Cheon presented a construction leading to (Partially) Blind-Signatures in the standard model. However their construction relies on a trick consisting in starting from prime order groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3$ and considering group elements in $\mathcal{G} = \mathbb{G}_1 \oplus \mathbb{G}_2 \oplus \mathbb{G}_3$. While their approach provides nice theoretical tools, the resulting signatures lies in $\mathcal{G}^2$ and are therefore three times longer than our proposal.

Our contributions. In this paper, we go one step further, improving [4] in several directions. We first present a blind signature scheme with perfect blindness, using the perfectly hiding instantiation of Groth-Sahai commitments [13]. We also widen the model of partially-blind signatures to supplement the predetermined communication with an on-the-fly public information generated by the signer: the signer can simply include it during the signing process, even if the user does not want this extra information. In the latter case, the user can simply discard the signature and start anew. We call this new primitive *signer-friendly partially-blind signatures*. This new notion allows to skip the prior agreement and allow the public information to be set on-the fly. Of course this new notion does not forbid any kind of prior agreement on the public part, it just strengthens the existing notion.

It is now possible to get rid of the prior agreement on the common piece of information in the signed message and our instantiation allows the signer to do so in a round-optimal way. These two constructions being compatible, we can present a round-optimal partially-blind signature with perfect blindness. Our protocol does not need any pre-processing for the public part of the message. Basically both the user and the signer can choose a piece of the public part, but instead of having a computational overhead for the agreement both can simply choose during the 2 flows interaction what they want. The signer can always refuse to sign something where the user's public information doesn't suit him and the user can always choose not to exploit an uninteresting signature, so a protocol should avoid to waste communication costs when one can manage without any security loss to stay in a two-flows protocol.

Eventually, discarding the perfect blindness, we take advantage of this asynchronous property (the user and the signer can independently choose their inputs) and we consider the new context where the message to be signed comes from several independent sources that cannot communicate together. We first present a way to obtain a signature on the concatenation of the input messages. We also present a shorter instantiation which gives a signature on the sum of the input messages. Such a sum can be useful when working on ballots, sensor information, etc. Since we still apply the Waters signature, this led us to consider the Waters function programmability over a non-binary alphabet, in a similar way as it was done in [14] for the binary alphabet. We prove a negative result on the $(2, 1)$-programmability, but a nice positive one on the $(1, poly)$-programmability, which is of independent interest.

Instantiations. We give several instantiations of our different blind signatures, all of which are based on weak assumptions. Our constructions mainly use the two following building blocks, from which they inherit their security: Groth-Sahai proofs for languages over pairing-friendly groups [13] and Waters signatures derived from the scheme in [20] and used in [7]. Since verification of the revisited Waters signatures [4] is a statement of the language for Groth-Sahai proofs, these two building blocks combine smoothly. The first instantiations are in symmetric pairing-friendly elliptic curves and additionally use linear commitments [6]. Both unforgeability and semantic security of these constructions rely solely on the decision linear assumption (DLin). The blindness property is easily achieved granted the homomorphic property of the Waters signature. An instantiation with improved efficiency, in *asymmetric* bilinear groups, using the SXDH variant of Groth-Sahai proofs and commitments is drafted in the Appendix E. This setting requires an asymmetric Waters signature scheme secure under a slightly stronger assumption, called CDH^+, where some additional elements in the second group are given to the adversary.

Applications. Our blind signature schemes find various kinds of applications:

E-voting. The security of several e-voting protocols relies on the fact that each ballot is certified by an election authority. Since this authority should not learn the voter's choice, a blind signature scheme (or even partially-blind, if the authority wants to specify the election in the ballot) is usually used to achieve this property. In order to achieve privacy of the ballot in an information-theoretic sense, it is necessary to use a signature scheme that achieves perfect blindness. Our scheme is the first to achieve this property in the standard model and under classical complexity assumptions.

E-cash. As mentioned above, partially-blind signatures played an important role in many electronic commerce applications. In e-cash systems, for instance, the bank issuing coins must ensure that the message contains accurate information such as the face value of the e-cash without seeing it and moreover in order to prevent double-spending, the bank's database has to record all spent coins. Partially-blind signatures can cope with these problems, since the bank can explicitly include some information such as the expiration date and the face value in the coin. Thanks to our proposal, the coin issuing protocol can be done without prior agreement between the bank and the client.

Data aggregation in networks. A *wireless (ad hoc) sensor network* (WSN) consists of many sensor nodes that are deployed for sensing the environment and collecting data from it. Since transmitting and receiving data are the most energy consuming operations, data aggregation has been put forward as an essential paradigm in these networks. The idea is to combine the data coming from different sources – minimizing the number of transmissions and thus saving energy. In this setting, a WSN consists usually of three types of nodes:

- *sensor nodes* that are small devices equipped with one or more sensors, a processor and a radio transceiver for wireless communication.
- *aggregation nodes* (or aggregators) performing the data aggregation (*e.g.* average, sum, minimum or maximum of data).
- *base stations* responsible for querying the nodes and gathering the data collected by them.

WSNs are at high security risk and two important security goals when doing in-network data aggregation are *data confidentiality* and *data integrity*. When homomorphic encryption is used for data aggregation, end-to-end encryption allows aggregation of the encrypted data so that the aggregators do not need to decrypt and get access to the data and thus provides end-to-end data confidentiality. Achieving data integrity is a harder problem and usually we do not consider the attack where a sensor node reports a false reading value (the impact of such an attack being usually limited). The main security flaw is a *data pollution attack* in which an attacker tampers with the intermediate aggregation result at an aggregation node. The purpose of the attack is to make the base station receive the wrong aggregation result, and thus make the improper or wrong decisions.

While in most conventional data aggregation protocols, data integrity and privacy are not preserved at the same time, our multi-source blind signature primitive permits to achieve data confidentiality and to prevent data pollution attacks simultaneously by using the following simple protocol:

1. Data aggregation is initiated by a base station, which broadcasts a query to the whole network.
2. Upon receiving the query, sensor nodes report encrypted values of their readings (for the base station public key) to their aggregators
3. The aggregators check the validity of the received values, perform data aggregation via the homomorphic properties of the encryption scheme, (blindly) sign the result and route the aggregated results back to the base station.
4. The base station decrypts the aggregated data and the signature which proves the validity of the gathered information to the base station (but also to any other third party).

2 Definition

This section presents the global framework and the security model for partially-blind signature schemes. A reminder of standard definition and security notions on Blind Signature can be found in the Appendix A.

Blind signatures introduced a nice feature, however it may be undesirable that requesters can ask the signer to blindly sign any message. For example, in an e-cash scheme, some expiration date information should be embedded in the e-coin, to avoid the bank's database an uncontrolled growth when storing information for double-spending checking. Partially-blind signatures are thus a natural extension of blind signatures: instead of signing an unknown message, the signer signs a message which contains a shared piece of information in addition to the hidden part. This piece is called info and, in the standard definition, is expected to have been defined before the execution of the protocol. But since our schemes will not require the public part to be agreed on by the two players before the protocol execution (as opposed to all the previous schemes from the literature), we extend the usual partially-blind signature scheme with two public parts in the message, in addition to the hidden part: $\mathsf{info} = \mathsf{info}_c \| \mathsf{info}_s$, where info_c is the common public part with prior agreement, and info_s is set on-the-fly by the signer. This provides a more flexible scheme, and this definition generalizes all the above ones. If $\mathsf{info}_s = \perp$, we are in the regular case of partially blind signature, whereas in case of regular blind signature both parts are empty $\perp$.

Definition 1 (Partially-Blind Signature Scheme). A $\mathcal{PBS}$ scheme is defined by 4 algorithms or protocols $(\mathsf{Setup}_{\mathcal{PBS}}, \mathsf{KeyGen}_{\mathcal{PBS}}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Verif}_{\mathcal{PBS}})$ where

- $\mathsf{Setup}_{\mathcal{PBS}}(1^\lambda)$ generates the global parameters param_{pbs} of the system;
- $\mathsf{KeyGen}_{\mathcal{PBS}}(\mathsf{param}_{pbs})$ generates a pair of keys $(\mathsf{pk}_{\mathcal{PBS}}, \mathsf{sk}_{\mathcal{PBS}})$;
- Signature Issuing: this is an interactive protocol between $\mathcal{S}(\mathsf{sk}_{\mathcal{PBS}}, \mathsf{info} = \mathsf{info}_c \| \mathsf{info}_s)$ and $\mathcal{U}(\mathsf{pk}_{\mathcal{PBS}}, m, \mathsf{info})$, for a message $m \in \{0,1\}^n$ and shared information info. It generates an output σ for the user: $\sigma \leftarrow \langle \mathcal{S}(\mathsf{sk}_{\mathcal{PBS}}, \mathsf{info}), \mathcal{U}(\mathsf{pk}_{\mathcal{PBS}}, m, \mathsf{info}) \rangle$.
- $\mathsf{Verif}_{\mathcal{PBS}}(\mathsf{pk}_{\mathcal{PBS}}, m, \mathsf{info}, \sigma)$ outputs 1 if the signature σ is valid with respect to the message $m \| \mathsf{info}$ and $\mathsf{pk}_{\mathcal{PBS}}$, 0 otherwise.

Quick note on security: The security requirements are a direct extension of the classical ones: for unforgeability, we consider $m \| \mathsf{info}$ instead of m, and for the blindness, we condition the unlinkability between signatures with the same public part info. Without the latter restriction, anyone can simply distinguish which message was signed by comparing the public information. The unforgeability is strengthened by considering also the public information so that the signer can be sure that the user won't be able to exploit his signature in another context.

Definition 2 (Signer-Friendly Partially-Blind Signature Scheme). A signer-friendly partially-blind signature scheme $\mathcal{PBS}$ is defined by 4 algorithms or protocols $(\mathsf{Setup}_{\mathcal{PBS}}, \mathsf{KeyGen}_{\mathcal{PBS}}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Verif}_{\mathcal{PBS}})$ where

- $\mathsf{Setup}(1^\lambda)$ generates the global parameters param_{pbs} of the system;
- $\mathsf{KeyGen}(\mathsf{param}_{pbs})$ generates a pair of keys $(\mathsf{pk}_{\mathcal{PBS}}, \mathsf{sk}_{\mathcal{PBS}})$;
- Signature Issuing: this is an interactive protocol between $\mathcal{S}(\mathsf{sk}_{\mathcal{PBS}}, \mathsf{info}_c, \mathsf{info}_s)$ and $\mathcal{U}(\mathsf{pk}_{\mathcal{PBS}}, m, \mathsf{info}_c)$, for a message $m \in \{0,1\}^n$, signer information info_s and common information info_c. It generates an output σ for the user: $\sigma \leftarrow \langle \mathcal{S}(\mathsf{sk}_{\mathcal{PBS}}, \mathsf{info}_c, \mathsf{info}_s), \mathcal{U}(\mathsf{pk}_{\mathcal{PBS}}, m, \mathsf{info}_c) \rangle$.
- $\mathsf{Verif}(\mathsf{pk}_{\mathcal{PBS}}, m, \mathsf{info}_c, \mathsf{info}_s, \sigma)$ outputs 1 if the signature σ is valid with respect to the message $m \| \mathsf{info}_c \| \mathsf{info}_s$ and $\mathsf{pk}_{\mathcal{PBS}}$, 0 otherwise.

One notes that $\mathsf{info}_c = \mathsf{info}$ and $\mathsf{info}_s = \perp$ lead to a standard partially-blind signature; whereas the case $\mathsf{info}_c = \mathsf{info}_s = \perp$ is the standard blind signature.

The signer always has the last word in the process, and so if he does not want to sign a specific info, he will simply abort the protocol several times until the shared part suits his will. So, in the following, we

decided that it was wiser to let him choose this input. If the user wants a specific word in the final message he can always add it to the blinded message. Intuitively this strengthens the unforgeability notion as the adversary (the user in this case) won't be able to chose the whole message to be signed because of info_s. This is ensured in the security game, because the adversary should outputs valid signatures, therefore they should be done with the chosen info_s. For the blindness property, the adversary should guess on signatures with the same public $\mathsf{info}_c||\mathsf{info}_s$ component, if it is not the case we answer with a blind-signature $\perp$.

The complete security games can be found in the Appendix B.

3 Partially-Blind Signature

Our constructions will combine Groth-Sahai Linear Commitments [13] and the Waters signature [20] as follows: given a commitment on the "Waters hash" $\mathcal{F}(M)$ (and some additional values proving we know the message M and the randomness used), a pre-agreed shared information info_c, the signer can make a partially-blind signature on M, info_c and an extra piece of public information info_s. This construction makes use of a symmetric pairing, but we extend it to asymmetric pairings in the Appendix E.

3.1 Assumptions

We rely on classical assumptions only: CDH for the unforgeability of signatures and DLin for the blindness property (when not perfect), and also for soundness of the proofs:

Definition 3 (The Computational Diffie-Hellman problem (CDH)). The CDH assumption, in a cyclic group $\mathbb{G}$ of prime order p, states that for a generator $g \in \mathbb{G}$ and random $a, b \in \mathbb{Z}_p$, given (g, g^a, g^b) it is hard to compute g^{ab}.

Definition 4 (Decision Linear Assumption (DLin)). The DLin assumption, in a cyclic group $\mathbb{G}$ of prime order p, states that given $(g, g^x, g^y, g^{xa}, g^{yb}, g^c)$ for random $a, b, x, y \in \mathbb{Z}_p$, it is hard to determine whether $c = a + b$ or a random value. When $(g, u = g^x, v = g^y)$ is fixed, a tuple (u^a, v^b, g^{a+b}) is called a linear tuple *w.r.t.* (u, v, g), whereas a tuple (u^a, v^b, g^c) for a random and independent c is called a random tuple.

One can easily see that if an adversary is able to solve a CDH challenge, then he can easily solve a DLin one. So the DLin assumption implies the CDH assumption. Some reminders on Groth-Sahai Commitments and Waters function can be found in the Appendix C as those are the main building blocks of our construction.

3.2 Partially-Blind Signature with Perfect Blindness

With those building blocks, we design a partially-blind signature scheme, which basically consists in committing the message to be signed. And granted the random coins of the commitment, the user can unblind the signature sent by the signer. Eventually, using the randomizability of the Waters signature, the user breaks all the links that could remain between the message-signature pair and the transaction. Our protocol proceeds as follows, on a commitment of $F = \mathcal{F}(M)$, a public common message info_c, and a public message info_s chosen by the signer. It is split into five steps, that correspond to an optimal 2-flow protocol: $\mathsf{Blind}_{\mathcal{BS}}$, which is first run by the user, $\mathsf{Sign}_{\mathcal{BS}}$, which is thereafter run by the signer, and $\mathsf{Verif}_{\mathcal{BS}}$, $\mathsf{Unblind}_{\mathcal{BS}}$, $\mathsf{Random}_{\mathcal{BS}}$ that are eventually successively run by the user to generate the final signature. We thus have $\mathcal{U} = (\mathsf{Blind}_{\mathcal{BS}}; \mathsf{Verif}_{\mathcal{BS}}, \mathsf{Unblind}_{\mathcal{BS}}, \mathsf{Random}_{\mathcal{BS}})$ and $\mathcal{S} = \mathsf{Sign}_{\mathcal{BS}}$:

- $\mathsf{Setup}_{\mathcal{BS}}(1^\lambda)$ first chooses a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, e, g)$. We need an additional vector $\boldsymbol{u} = (u_0, \ldots, u_k) \xleftarrow{\$} \mathbb{G}^{k+1}$ which defines the Waters function $\mathcal{F}$ (where k is the global length of $M||\mathsf{info}_c||\mathsf{info}_s$), a generator $h \xleftarrow{\$} \mathbb{G}$, and a tuple of Groth-Sahai parameters $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ in the perfectly hiding setting: $\mathsf{param}_{bs} = (p, \mathbb{G}, \mathbb{G}_T, e, g, h, \mathcal{F}, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$;

– $\mathsf{KeyGen}_{\mathcal{BS}}(\mathsf{param}_{bs})$ chooses a random scalar $x \xleftarrow{\$} \mathbb{Z}_p$, which defines the public key as $\mathsf{pk}_{\mathcal{BS}} = Y = g^x$, and the secret key as $\mathsf{sk}_{\mathcal{BS}} = Z = h^x$;
– Signature Issuing $(\mathcal{S}(\mathsf{sk}_{\mathcal{BS}}, \mathsf{info}_c, \mathsf{info}_s), \mathcal{U}(\mathsf{pk}_{\mathcal{BS}}, M, \mathsf{info}_c))$, which is split in several steps:
  - $\mathsf{Blind}_{\mathcal{BS}}(M, \mathsf{pk}_{\mathcal{BS}}; (r_1, r_2, r_3))$: For a message $M \in \{0,1\}^\ell$ and random scalars $(r_1, r_2, r_3) \xleftarrow{\$} \mathbb{Z}_p$, define the commitment as $c = \big(c_1 = u_{1,1}^{r_1} u_{3,1}^{r_3}, c_2 = u_{2,2}^{r_2} u_{3,2}^{r_3}, c_3 = g^{r_1+r_2} u_{3,3}^{r_3} \cdot \mathcal{F}(M)\big)$ and compute $Y_{1,2} = Y^{r_1+r_2}, Y_3 = Y^{r_3}$. One also generates additional proofs of validity of the commitment:
    * A proof Π_M of knowledge of M in c, the encrypted $\mathcal{F}(M)$, which consists of a bit-by-bit commitment $C_M = (\mathcal{C}'(M_1), \ldots, \mathcal{C}'(M_\ell))$ and proofs that each committed value is a bit, and a proof that c_3 is well-formed. Π_M is therefore composed of $9\ell + 3$ group elements.
    * A proof Π_r containing the commitments $C_r = (\mathcal{C}(Y_{1,2}), \mathcal{C}(Y_3))$ and proofs asserting that they are correctly generated. It requires 9 additional group elements.

    Π thus consists of $9\ell + 12$ group elements, where ℓ is the bit-length of the message M
  - $\mathsf{Sign}_{\mathcal{BS}}(\mathsf{sk}_{\mathcal{BS}}, (c, \Pi), \mathsf{info}_c, \mathsf{info}_s; s)$: To sign the commitment c, one first checks if the proof Π is valid. It then appends the public message $\mathsf{info} = \mathsf{info}_c \| \mathsf{info}_s$ to c_3 to create $c'_3 = c_3 \cdot \prod u_{i+\ell}^{\mathsf{info}_i}$, which thus becomes a commitment of the Waters function evaluation on $M \| \mathsf{info}_c \| \mathsf{info}_s$ of global length k. It eventually outputs $\sigma = (Z \cdot {c'_3}^s, u_{3,3}^s, g^s)$ together with the additional public information info_s, for a random scalar $s \in \mathbb{Z}_p$.
  - $\mathsf{Verif}(\mathsf{pk}_{\mathcal{BS}}, (c, \mathsf{info}_c, \mathsf{info}_s), \sigma = (\sigma_1, \sigma_2, \sigma_3))$: In order to check the validity of the signature, one first computes c'_3 as above, and then checks whether the following pairing equations are verified: $e(\sigma_1, g) = e(h, \mathsf{pk}_{\mathcal{BS}}) \cdot e(c'_3, \sigma_3)$ and $e(\sigma_2, g) = e(u_{3,3}, \sigma_3)$. If it is not the case, then this is not a valid signature on the original ciphertext, and the blind signature is set as $\Sigma = \perp$.
  - $\mathsf{Unblind}_{\mathcal{BS}}((r_1, r_2, r_3), \mathsf{pk}_{\mathcal{BS}}, (c, \mathsf{info}_c, \mathsf{info}_s), \sigma)$: If the previous tests are positive, one can use the random coins r_1, r_2, r_3 to get back a valid signature on $M \| \mathsf{info}_c \| \mathsf{info}_s$: $\sigma' = (\sigma'_1 = \sigma_1 / (\sigma_3^{r_1+r_2} \sigma_2^{r_3}), \sigma'_2 = \sigma_3)$, which is a valid Waters signature.
  - $\mathsf{Random}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, (c, \mathsf{info}_c, \mathsf{info}_s), \sigma'; s')$: The latter can eventually be rerandomized to get $\Sigma = (\sigma'_1 \cdot \mathcal{F}(M || \mathsf{info}_c || \mathsf{info}_s)^{s'}, \sigma'_2 \cdot g^{s'})$.

  One can note that Σ is a random Waters signature on $M || \mathsf{info}_c || \mathsf{info}_s$, where we denote $F = \mathcal{F}(M || \mathsf{info}_c || \mathsf{info}_s)$:

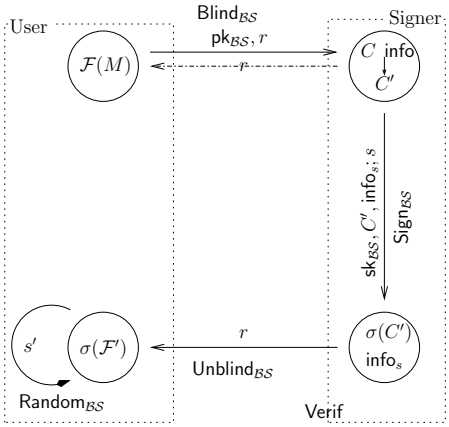
$$\begin{aligned}\Sigma &= (\sigma'_1 \cdot F^{s'}, \sigma'_2 \cdot g^{s'}) = (F^{s'} \cdot \sigma_1 / (\sigma_3^{r_1+r_2} \sigma_2^{r_3}), g^{s'} \cdot \sigma_3) \\ &= (F^{s'} \cdot Z \cdot {c'_3}^s / (g^{s(r_1+r_2)} u_{3,3}^{sr_3}), g^{s+s'}) \\ &= (F^{s'} \cdot Z \cdot g^{s(r_1+r_2)} u_{3,3}^{sr_3} \cdot F^s / (g^{s(r_1+r_2)} u_{3,3}^{sr_3}), g^{s+s'}) = (M^{s+s'} \cdot Z, g^{s+s'})\end{aligned}$$

– $\mathsf{Verif}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, (M, \mathsf{info}_c, \mathsf{info}_s), \Sigma = (\Sigma_1, \Sigma_2))$: One checks whether the following pairing equations holds (Waters signature): $e(\Sigma_1, g) = e(h, \mathsf{pk}_{\mathcal{BS}}) \cdot e(\mathcal{F}(M || \mathsf{info}_c || \mathsf{info}_s), \Sigma_2)$.

Theorem 5. *This signer-friendly partially-blind signature scheme is unforgeable under the* CDH *assumption in* $\mathbb{G}$.

Proof. Let us denote $\mathcal{PBS}$ our above partially-blind signature (but omit it in the subscripts for clarity). Let us assume there is an adversary $\mathcal{A}$ against the unforgeability that succeeds within probability ϵ, we will build an adversary $\mathcal{B}$ against the CDH problem.

DLin *Assumption.* The unforgeability means that after q_s interactions with the signer, the adversary manages to output $q_s + 1$ valid message-signature pairs on distinct messages. If the adversary $\mathcal{A}$ can do that with probability ϵ with the above commitment scheme using a perfectly hiding setting, under the DLin assumption, $\mathcal{A}$ can also generate $q_s + 1$ valid message-signature pairs in a perfectly binding setting, with not too small probability ϵ'.



A message M can be hidden using random coins r ($\mathsf{Blind}_{\mathcal{BS}}$).
The signer can adapt this commitment and concatenate a public message info_s into the original commitment, with also the common public information info_s, creating a commitment C' on $F = \mathcal{F}(M||\mathsf{info}_c||\mathsf{info}_s)$.
A signature on the plaintext can be obtained using the randomness r (for $\mathsf{Unblind}_{\mathcal{BS}}$); the result is the same as a direct signature on $M||\mathsf{info}_c||\mathsf{info}_s$ by the signer.
Randomizing this signature is easy, and prevents the signer to actually know which ciphertext was the one involved.

Figure 1. Partially-Blind Signatures with Perfect Blindness

Signer Simulation. Let us thus now consider the above blind signature scheme with a commitment scheme using a perfectly binding setting (named $\mathcal{PBS}'$), and our simulator $\mathcal{B}$ can extract values from the commitments since it knows ν and μ. We thus now assume that $\mathcal{A}$ is able to break the unforgeability of $\mathcal{PBS}'$ with probability ϵ' after q_s interactions with the signer. And we build an adversary $\mathcal{B}$ against the CDH problem: Let $(A = g^a, B = g^b)$ be a CDH-instance in a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, e, g)$.

We now generate the global parameters using this instance: for simulating $\mathsf{Setup}_{\mathcal{BS}}/\mathsf{KeyGen}_{\mathcal{BS}}$, $\mathcal{B}$ picks a random position $j \xleftarrow{\$} \{0, \ldots, k\}$, chooses random indexes $y_0, y_1, \ldots, y_k \xleftarrow{\$} \{0, \ldots, 2q_s - 1\}$, and random scalars $z_0, z_1, \ldots, z_k \xleftarrow{\$} \mathbb{Z}_p$. One defines $Y = A = g^a$, $h = B = g^b$, $u_0 = h^{y_0 - 2jq_s} g^{z_0}$, and $u_i = h^{y_i} g^{z_i}$ for $i = 1, \ldots, k$. $\mathcal{B}$ also picks two random scalars ν, μ,and generates the Groth-Sahai parameters $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ in the perfectly binding setting, and thus with $(\mathbf{u}_1 = (u_{1,1} = g^{x_1}, 1, g), \mathbf{u}_2 = (1, u_{2,2} = g^{x_2}, g), \mathbf{u}_3 = \mathbf{u}_1^{\nu} \odot \mathbf{u}_2^{\mu})$, for two random scalars x_1, x_2. Note that $u_{3,3} = g^{\nu+\mu}$. It outputs $\mathsf{param}_{bs} = (p, \mathbb{G}, \mathbb{G}_T, e, g, h, \mathcal{F}, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$; one can note that the signing key is implicitly defined as $Z = h^a = B^a = g^{ab}$, and is thus the expected Diffie-Hellman value.

To answer a signing query on ciphertext $c = (c_1, c_2, c_3)$, with the additional proofs, one first checks the proof Π. From the proof Π and the commitment secret parameters x_1, x_2, $\mathcal{B}$ can extract M from the bit-by-bit commitments in Π_M, and $Y_{1,2} = Y^{r_1+r_2}$, $Y_3 = Y^{r_3}$, from Π_r, where $c_1 = u_{1,1}^{r_1} u_{3,1}^{r_3}$ and $c_2 = u_{2,2}^{r_2} u_{3,2}^{r_3}$. Furthermore, we can compute $c_3' = g^{r_1+r_2} u_{3,3}^{r_3} \cdot F$, where we denote $M' = M||\mathsf{info}_c||\mathsf{info}_s$ and $F = \mathcal{F}(M||\mathsf{info}_c||\mathsf{info}_s)$. $\mathcal{B}$ defines

$$H = -2jq_s + y_0 + \sum_i y_i M_i', \quad J = z_0 + \sum_i z_i M_i' \quad : \quad F = h^H g^J.$$

If $H \equiv 0 \pmod p$ then $\mathcal{B}$ aborts, otherwise it sets

$$\sigma = (Y^{-J/H}(Y_{1,2} Y_3^{\nu+\mu})^{-1/H}(F(c_1^{1/x_1} c_2^{1/x_2}))^s, (Y^{-1/H} g^s)^{\nu+\mu}, Y^{-1/H} g^s).$$

Defining $\tilde{s} = s - a/H$, we have

$$\begin{aligned}
\sigma_1 &= Y^{-J/H}(Y_{1,2} Y_3^{\nu+\mu})^{-1/H}(h^H g^J (c_1^{1/x_1} c_2^{1/x_2}))^s = Z \cdot (c_3')^{\tilde{s}} \\
\sigma_3 &= Y^{-1/H} g^s = Y^{-1/H} g^{\tilde{s}+a/H} = g^{\tilde{s}} \\
\sigma_2 &= (\sigma_3)^{\nu+\mu} = g^{(\nu+\mu)\tilde{s}} = u_{3,3}^{\tilde{s}}
\end{aligned}$$

It thus exactly looks like a real signature sent by the signer.

Diffie-Hellman Extraction. After at most q_s signing queries $\mathcal{A}$ outputs $q_s + 1$ valid Waters signatures. Since there are more than the number of signing queries, there is a least one message M^* that is different from all the messages $M||\mathsf{info}_c||\mathsf{info}_s$ involved in the signing queries. We define

$$H^* = -2jq_s + y_0 + \sum_i y_i M_i^*, \quad J^* = z_0 + \sum_i z_i M_i^* \quad : \quad \mathcal{F}(M^*) = h^{H^*} g^{J^*}.$$

If $H^* \not\equiv 0 \pmod p$ then $\mathcal{B}$ abort, otherwise, for some s^*, $\sigma^* = (h^a \mathcal{F}(M^*)^{s^*}, g^{s^*}) = (h^a g^{s^* J^*}, g^{s^*})$. Then, $\sigma_1^*/(\sigma_2^*)^{J^*} = h^a = g^{ab}$: one has solved the CDH problem.

Success Probability. (Based on [14]) The Waters hash function is $(1, q_s)$-programmable (*i.e.*, we can find with non negligible probability a case where q_s intermediate hashes are not null, and the last one is), therefore the previous simulation succeeds with non negligible probability ($\Theta(\epsilon/q_s\sqrt{k})$), and so $\mathcal{B}$ breaks CDH. □

Theorem 6. *This signer-friendly partially-blind signature scheme achieves perfect blindness.*

Proof. The transcript sent to the signer contains a commitment on the message to be signed, but in a perfectly hiding setting: no information leaks about M. The additional proofs are perfectly witness-indistinguishable and thus do not provide any additional information about M. This is due to the fact that in the Groth Sahai framework in the perfectly hiding setting, for any message M, committed with randomness r and a message M', one can find random r' such that $c(M, r) = c(M', r')$. Granted the randomizability of the Waters signature, the final output signature is a random signature on $M||\mathsf{info}_c||\mathsf{info}_s$, on which no information leaked, and so the resulting signature is perfectly independent from the transcript seen by the signer, and any adversary. □

4 Multi-Source Blind Signature

4.1 Concatenation

The previous constructions lead to a good way to allow a user to obtain a signature on a plaintext without revealing it to the signer. But what happens when the original message is in fact coming from various users? We now present a new way to obtain a blind signature without requiring multiple users to combine their messages, providing once again a round-optimal way to achieve our goal.

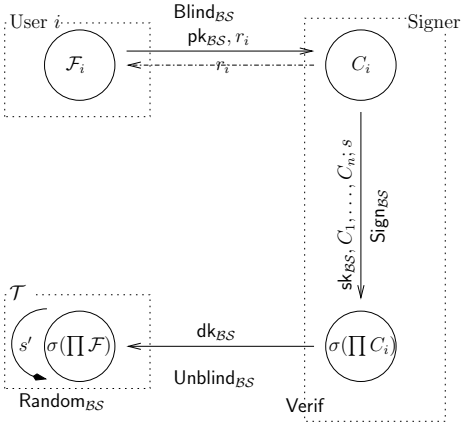
We thus consider a variation of our blind signature scheme. In the Setup phase we no longer create perfectly hiding Groth-Sahai generators, but perfectly binding parameters, so we do not need to compute $u_{3,3}^s$ to run $\mathsf{Unblind}$, since it will be performed with the decryption key and not the random coins. In addition, in this scenario, we do not consider a unique user providing a ciphertext, but several users. As a consequence, the signer will have to produce a signature on a multi-source message, provided as ciphertexts. The signature and the messages will actually be encrypted under a third-party key. The third-party only will be able to extract the message and the signature.

Basically the instantiation is similar to the previous ones in the perfectly binding setting. For the sake of clarity, we remove the partially-blind part, but of course it could be adapted in the same way.

A full instantiation of such protocol and its security analysis can be found in the Appendix D. One can see that it can be efficiently instantiated under DLin assumption.

4.2 Addition

The previous scheme presents a way to combine multiple blind messages into one in order to sign it. However it requires a huge number of generators and the final unblinded signature gives a lot of information on the repartition of the original messages, since they are simply concatenated. We now want to improve the previous



Several messages M_i can be hidden using random coins r_i ($\mathsf{Blind}_{\mathcal{BS}}$) by different users.
The signer can adapt these commitments and concatenate the messages inside them, creating a commitment on $\mathcal{F} = \prod \mathcal{F}_i$.
A signature on the plaintext can be obtained by the tallier using the decryption key $\mathsf{dk}_{\mathcal{BS}}$ (for $\mathsf{Unblind}_{\mathcal{BS}}$); the result is the same as a direct signature on $||M_i$ by the signer.
Randomizing this signature is easy, and prevent the signer from knowing which ciphertexts were involved.

Figure 2. Multi-Source Blind Signature on Concatenation

scheme to drastically reduce the public key size, and the information leaked about the individual messages when one would like a signature on some computation on these messages, such as the addition or the mean. Instead of signing the concatenation of the messages, we now allow the users to use the same generators, and thus the messages will add together instead of concatenating.

The resulting algorithm is the same as before except during the Setup phase where $\boldsymbol{u} = (u_0, \ldots, u_k) \xleftarrow{\$} \mathbb{G}^{k+1}$. We then proceed as before considering $\mathcal{F}(M_i) = \prod_\ell u_\ell^{m_{i,\ell}}$. The $\mathsf{Unblind}$ algorithm now returns a valid signature on the sum of the messages. The various Groth-Sahai proofs help to ensure that the messages given to the Waters hash function are of reasonable size.

With this construction, the exponents in the Waters hash function are not longer bits but belong to a larger alphabet (*e.g.* $\{0, \ldots, t\}$ if t users sign only bit strings). Following the work done in [14], we will show in the next section that over a non-binary alphabet the Waters function remains $(1, poly)$-programmable as long as the size of the alphabet a polynomial in the security parameter. This result readily implies the security of the multi-source blind signature scheme for addition:

Theorem 7. *This multi-source blind signature scheme for addition is blind and unforgeable under the* DLin *assumption as long the alphabet size and the number of sources are polynomial in the security parameter.*

5 Non-Binary Waters Function Programmability

In this section, we prove that for a polynomial-size alphabet, the Waters function remains programmable. We recall some notations introduced in [14] and show our result which can be seen as an improvement over the result presented by Naccache [16] where he considered a variant of Waters identity-based encryption [20] with shorter public parameters.

5.1 Definitions

Let us recall some basic definitions. A family of cyclic groups $G = (\mathbb{G}_\lambda)_{\lambda \in \mathbb{N}}$, indexed by a security parameter λ, is called a *group family*. A *group hash function* H for G, an alphabet $\Sigma = \Sigma(\lambda)$ and an input length $\ell = \ell(\lambda)$ is a pair of probabilistic polynomial-time algorithms $(\mathsf{PHF.Gen}, \mathsf{PHF.Eval})$ such that:

- $\mathsf{PHF.Gen}$ takes as input a security parameter λ and outputs a key κ
- $\mathsf{PHF.Eval}$ takes as input a key κ output by $\mathsf{PHF.Gen}$ and a string $X \in \Sigma^\ell$ and outputs an element of $\mathbb{G}_\lambda$.

Definition 8 (*cf.* [14]). A group hash function $(\mathsf{PHF.Gen}, \mathsf{PHF.Eval})$ is (m, n, δ)-programmable, if there exist two PPT algorithms $(\mathsf{PHF.TrapGen}, \mathsf{PHF.TrapEval})$ such that

- **Syntactics:** For $g, h \in \mathbb{G}$, $\mathsf{PHF.TrapGen}(1^\lambda, g, h)$ generates a key κ' and a trapdoor t such that $\mathsf{PHF.TrapEval}(t, X)$ produces integers a_X, b_X for any $X \in \Sigma^\ell$
- **Correctness:** For all generators $g, h \in \mathbb{G}$, all $(\kappa', t) \leftarrow \mathsf{PHF.TrapGen}(1^\lambda, g, h)$ and all $X \in \Sigma^\ell$, $H_{\kappa'}(X) := \mathsf{PHF.Eval}(\kappa', X)$ satisfies $H_{\kappa'}(X) = g^{a_X} h^{b_X}$ where $(a_X, b_X) := \mathsf{PHF.TrapEval}(t, X)$.
- **Statistically close trapdoor keys:** For all generators $g, h \in \mathbb{G}^2$, the functions $\mathsf{PHF.Gen}(1^\lambda)$ and $\mathsf{PHF.TrapGen}(1^\lambda, g, h)$ output keys κ and κ' statistically close.
- **Well-distributed logarithms:** For all generators $g, h \in \mathbb{G}$, all (κ', t) output by $\mathsf{PHF.TrapGen}(1^\lambda, g, h)$ and all bit-strings $(X_i)_{1,\ldots,m}, (Z_i)_{1,\ldots,n} \in \Sigma^\ell$ such that $\forall i, j, X_i \neq Z_j$, we have $\Pr[a_{X_1} = \ldots, a_{X_m} = 0 \wedge a_{Z_1} \cdot \ldots \cdot a_{Z_n} \neq 0] \geq \delta$, where the probability is taken over the random coins used by $\mathsf{PHF.TrapGen}$ and $(a_{X_i}, b_{X_i}) := \mathsf{PHF.TrapEval}(t, X_i)$ and $(a_{Z_i}, b_{Z_i}) := \mathsf{PHF.TrapEval}(t, Z_i)$.

5.2 Instantiation with Waters function

Let us consider the Waters function presented in [20].

Definition 9 (Multi-Generator PHF). Let $G = (\mathbb{G}_\lambda)$ be a group family, and $\ell = \ell(\lambda)$ a polynomial. We define $\mathcal{F} = (\mathsf{PHF.Gen}, \mathsf{PHF.Eval})$ as the following group hash function:

- $\mathsf{PHF.Gen}(1^\lambda)$ outputs $\kappa = (h_0, \ldots, h_\ell) \xleftarrow{\$} \mathbb{G}^{\ell+1}$;
- $\mathsf{PHF.Eval}(\kappa, X)$ parses κ and $X = (x_1, \ldots, x_\ell) \in \{0,1\}^\ell$ and outputs $\mathcal{F}_\kappa(X) = h_0 \prod_{i=1}^{\ell} h_i^{x_i}$.

This function was shown to be $(1, q, \delta)$-programmable with a $\delta = O(1/(q\sqrt{\ell}))$ and $(2, 1, \delta)$-programmable with a $\delta = O(1/\ell)$ (*cf.* [14]). However this definition requires to generate and store $n + 1$ group generators where n is the bit-length of the messages one wants to hash. We consider a more general case where instead of hashing bit-per-bit we decide to hash blocks of bits.

Definition 10 (Improved Multi-Generator PHF). Let $G = (\mathbb{G}_\lambda)$ be a group family, $\Sigma = \{0, \ldots, \tau\}$ a finite alphabet and $\ell = \ell(\lambda)$ a polynomial. We define $\mathcal{F} = (\mathsf{PHF.Gen}, \mathsf{PHF.Eval})$ as the following group hash function:

- $\mathsf{PHF.Gen}(1^\lambda)$ returns $\kappa = (h_0, \ldots, h_\ell) \xleftarrow{\$} \mathbb{G}^{\ell+1}$;
- $\mathsf{PHF.Eval}(\kappa, X)$ parses κ and $X = (x_1, \ldots, x_\ell) \in \Sigma^\ell$ and returns ${\mathcal{F}^+}_\kappa(X) = h_0 \prod_{i=1}^{\ell} h_i^{x_i}$.

Using a larger alphabet allows to hash from a larger domain with a smaller hash key, but it comes at a price since one can easily prove that the function is no longer $(2, 1)$-programmable (*i.e.*, no longer $(2, 1, \delta)$ programmable for a non-negligible δ):

Theorem 11 ((2,1)-Programmability). *For any group family G with known order and $\tau > 1$, the function $\mathcal{F}^+$ is not a (2,1)-programmable hash function if the discrete logarithm problem is hard in G.*

Proof. Consider a discrete logarithm challenge (g, h) in a group $\mathbb{G}_\lambda$ and suppose by contradiction that the function $\mathcal{F}^+$ is $(2, 1)$-programmable with $\tau \geq 2$ (*i.e.*, we suppose that there exist two probabilistic polynomial-time algorithms $(\mathsf{PHF.TrapGen}, \mathsf{PHF.TrapEval})$ satisfying the definition 8 for a non-negligible δ).

For any hash key κ' and trapdoor t generated by $\mathsf{PHF.TrapGen}(1^\lambda, g, h)$, we can consider the messages $X_1 = (2, 0), X_2 = (1, 1), Z = (0, 2)$ and with non-negligible probability over the random coins used by $\mathsf{PHF.TrapGen}$ we have $a_{X_1} = a_{X_2} = 0$ and $a_Z \neq 0$ where $(a_{X_1}, b_{X_1}) := \mathsf{PHF.TrapEval}(t, X_1)$, $(a_{X_2}, b_{X_2}) := \mathsf{PHF.TrapEval}(t, X_2)$ and $(a_Z, b_Z) := \mathsf{PHF.TrapEval}(t, Z)$. By the correctness property, we have $g^{a_Z} h^{b_Z} = h_0 h_2^2 = h^{2b_{X_2}} / h^{b_{X_1}}$ and we can extract the discrete logarithm of g in base h as follows:

$$\log_h(g) = \frac{2b_{X_2} - b_{X_1} - b_Z}{a_Z} \mod |\mathbb{G}_\lambda|.$$

□

However we still have the interesting property:

Theorem 12 ((1,poly)-Programmability). *For any polynomial q and a group family G with groups of known order, the function $\mathcal{F}^+$ is a $(1, q, \delta)$-programmable hash function with a $\delta = \Omega(1/\tau q\sqrt{\ell})$.*

Remark 13. This theorem improves the result presented by Naccache in [16] where the lower bound on the $(1, q, \delta)$-programmability was only $\delta = \Omega(1/\tau q\ell)$.

Remark 14. In order to be able to sign all messages in a set $\mathcal{M}$, we have to consider parameters τ and ℓ such that $\tau^\ell \geq \#\mathcal{M}$, but the security is proved only if the value δ is non-negligible (*i.e.* if $\ell = \lambda^{O(1)}$ and $\tau = \lambda^{O(1)}$). In particular if $\mathcal{M}$ is of polynomial size in λ (which is the case in our WSN application with data aggregation), one can use $\tau = \#\mathcal{M}$ and $\ell = 1$ (namely, the Boneh-Boyen hash function), and therefore get data confidentiality.

Proof. Let us first introduce some notations. Let $n \in \mathbb{N}^*$, let A_j be independent and uniform random variables in $\{-1, 0, 1\}$ (for $j \in \{1, \ldots, n\}$). If we denote $2\sigma_j^2$ their quadratic moment, we have $2\sigma_j^2 = 2/3$ and $\sigma_j = \sqrt{1/3}$. We note $s_n^2 = \sum_{j=1}^n \sigma_j^2 = n/3$.

The Local Central Limit Theorem. Our analysis relies on a classical result on random walks, called the *Local Central Limit Theorem*. It basically provides an approximation of $\Pr[\sum A_j = a]$ for independent random variables A_j. This is a version of the Central Limit Theorem in which the conclusion is strengthened from convergence of the law to locally uniform pointwise convergence of the densities. It is worded as follows in [9, *Theorem 1.1*], where ϕ and Φ are the standard normal density and distribution functions:

Theorem 15. *Let A_j be independent, integer-valued random variables where A_j has probability mass function f_j (for $j \in \mathbb{N}^*$). For each $j, n \in \mathbb{N}^*$, let $q(f_j) = \sum_k \min(f_j(k), f_j(k+1))$ and $Q_n = \sum_{j=1}^n q(f_j)$. Denote $S_n = A_1 + \cdots + A_n$. Suppose that there are sequences of numbers $(\alpha_n), (\beta_n)$ such that*

1. *$\lim_{n\to\infty} \Pr[(S_n - \alpha_n)/\beta_n) < t] = \Phi(t), -\infty < t < \infty$,*
2. *$\beta_n \to \infty$,*
3. *and $\limsup \beta_n^2/Q_n < \infty$,*

then $\sup_k |\beta_n \Pr[S_n = k] - \phi((k - \alpha_n)/\beta_n)| \to 0$ as $n \to \infty$[1].

While those notations may seem a little overwhelming, this can be easily explained in our case. With $A_j \in \{-1, 0, 1\}$ with probability $1/3$ for each value.

1. It requires the variables to verify the Lindeberg-Feller theorem. However as long as the variables verify the Lindeberg's condition[2], this is true for $\beta_n = s_n$ and $\alpha_n = 0$.
2. In our application, $\beta_n = s_n = \sqrt{n/3}$, so again we comply with the condition.
3. Since $f_j(k)$ is simply the probability that A_j equals k, then $q(f_j) = 2/3$. This leads to $Q_n = 2n/3$. As a consequence, $\beta_n^2/Q_n = 1/2$.

So we have: $\sup_k |\beta_n \Pr[S_n = k] - \phi((k - \alpha_n)/\beta_n)| \to 0$, that is, in our case

$$\sup_k |\sqrt{n/3} \Pr[S_n = k] - \phi(k/\sqrt{n/3})| \to 0.$$

We solely focus on the case $k = 0$: since $\phi(0) = 1/\sqrt{2\pi}$, $\Pr[S_n = 0] = \Theta(1/\sqrt{n})$. In addition, it is clear that $\Pr[S_n = k] \leq \Pr[S_n = 0]$ for any $k \neq 0$ (*c.f.* [14]).

[1] The so-called Berry-Esseen theorem gives the rate of convergence of this supremum.

[2] Lindeberg's condition is a sufficient criteria of the Lindeberg-Feller theorem, for variables with a null expected value it requires that $\forall \epsilon > 0, \lim_{n\to\infty} 1/s_n^2 \sum_{j=1}^n E[A_j^2 \cdot 1_{\{|A_j| > \epsilon s_n\}}] \to 0$. In our case, as soon as $n > 3/\epsilon^2$, we have $|A_j| \leq 1 \leq \epsilon\sqrt{n/3} \leq \epsilon s_n$, so the sum is null. ($1_{\{|A_j| > \epsilon s_n\}}$ is the indicator function of variables greater that ϵs_n)

Lemma 16. *Let $(A_{ij})_{[\![1,n]\!]\times[\![1,J]\!]}$ be independent, integer-valued random variables in $\{-1,0,1\}$, then $\forall X \in [\![1,\tau]\!]^n$, $\Pr[\sum_{i=1}^n \sum_{j=1}^J X_i A_{ij} = 0] = \Omega(1/\tau\sqrt{nJ})$, where the probability distribution is over the A_{ij}.*

This lemma will be useful to prove the lower bound in the following, we only consider word with no null coefficient X_i, if a X_i is null, we simply work with a shorter random walk of length $J \cdot (n-1)$ instead of Jn.

Proof. Let us denote d_{ij}, the random variable defined as $X_i A_{ij}$: they are independent, integer-valued random variables. As above, $s_n^2 = \sum_{i=1}^n \sum_{j=1}^J \sigma_j^2 = \sum_{i=1}^n JX_i^2/3$. So $nJ/3 \leq s_n^2 \leq n\tau^2 J/3$.

1. The Lindeberg's condition is verified. As soon as $n > 3\tau/J\epsilon^2$ we have $\epsilon s_n > \tau$ and so $|d_{ij}| < s_n$, and so once again the sum is null.
2. $s_n \to \infty$.
3. Each $d_{ij} \in \{-X_i, 0, X_i\}$ with probability $1/3$ for each value, so $q(f_{ij}) = 2/3$ and $Q_n = \sum_{i,j} q(f_{ij}) = 2nJ/3$. So $\beta_n^2/Q_n \leq (n\tau J/3)/(2nJ/3) \leq \tau/2 < \infty$.

Then we can apply the Local Central Limit Theorem to the d_{ij}'s, and conclude: $\Pr[\sum_{i=1}^n \sum_{j=1}^J X_i A_{ij} = 0] = \Theta(1/s_n) = \Theta(1/\tau\sqrt{(nJ)})$. □

In the following, we will denote $a(X) = \sum_{i=1}^n a_i X_i$, where $X \in \{0,\ldots,\tau\}^n$. The probabilities will be over the a_{ij}'s variables while X and Y are assumed to be chosen by the adversary. Our goal is to show that even for bad choices of X and Y, a random draw of a_{ij}'s provides enough freedom.

Let $J = J(\lambda)$ be a positive function. We define the following two probabilistic polynomial-time algorithms ($\mathsf{PHF.TrapGen}, \mathsf{PHF.TrapEval}$):

- $\mathsf{PHF.TrapGen}(1^\lambda, g, h)$**:** which chooses some independent and uniform elements $(a_{ij})_{(0,\ldots,\ell),(1,\ldots,J)}$ in $\{-1,0,1\}$, and random group exponents $(b_i)_{(0,\ldots,\ell)}$. It sets $a_i = \sum_{j=1}^J a_{ij}$ and $h_i = g^{a_i} h^{b_i}$ for $i \in \{0,\ldots,\ell\}$. It then outputs the hash key $\kappa = (h_0,\ldots,h_\ell)$ and the trapdoor $t = (a_0, b_0, \ldots, a_\ell, b_\ell)$.
- $\mathsf{PHF.TrapEval}(t, X)$**:** which parses $X = (X_1,\ldots,X_\ell) \in \Sigma^\ell = \{0,\ldots,\tau\}^\ell$ and outputs $a_X = a_0 + \sum a_i X_i$ and $b_X = b_0 + \sum b_i X_i$.

As this definition verifies readily the syntactic and correctness requirements, we only have to prove the two other ones. We stress the importance of the hardwired 1 in front of a_0 this allows us to consider multisets $X' = 1 :: X$ and $Y' = 1 :: Y$, and so there is no k such that $X' = kY'$. And we also stress that $a_i = \sum_{j=1}^J a_{ij}$ is already a random walk of length J (described by the a_{ij}), on which we can apply the Local Central Limit Theorem and so $\Pr[a_i = 0] = \Theta(1/\sqrt{J})$. By noticing that summing independent random walks is equivalent to a longer one and applying the Local Central Limit Theorem, we have:

$$\Theta(1/\tau\sqrt{(\ell+1)J}) \leq \Pr[a(X') = 0] \leq \Theta(1/\sqrt{J}).$$

To explain further the two bounds:

- For the upper bound: we consider X fixed, and note $t = \sum_{i=1}^\ell a_i X_i$, by construction a_i are independent, so a_0 is independent from t then
$$\Pr[a(X') = 0] = \Pr[a_0 = -t] \leq \Pr[a_0 = 0] \leq \Theta(1/\sqrt{J})$$
using the above remark that a random walk is more likely to reach 0 than any other value, and a_0 is a random walk of length J.
- For the lower bound, we proceed by recurrence on ℓ, to show
$$H_\ell : \Theta(1/\tau\sqrt{(\ell+1)J}) \leq \Pr[a(X') = 0] \quad (\text{where } X' \in 1 :: [\![0,\tau]\!]^\ell).$$
For $\ell = 0$, we consider $X' = 1$, we have a random walk of length J, so $\Theta(1/\tau\sqrt{J}) \leq \Theta(1/\sqrt{J}) \leq \Pr[a(X') = 0]$. We note $X_0 = 1$ for the hardwired 1 in X'. Let us suppose the property true at rank k, let us prove it at rank $k+1$:

- If $\exists i_0, X_{i_0} = 0$ then we can consider a random walk of length k and apply the previous step, and conclude because $\Theta(1/\tau\sqrt{(k+1)J}) \leq \Theta(1/\tau\sqrt{kJ})$
- Else, one can apply Lemma 16 to conclude.

Therefore, $\forall \ell, \forall X' \in 1 :: [\![0,\tau]\!]^\ell, \Theta(1/\tau\sqrt{(\ell+1)J}) \leq \Pr[a(X') = 0]$.

We can now deduce that $\forall X, Y \in [\![0,\tau]\!]^\ell$ with $X \neq Y$: $\Pr[a(Y') = 0 | a(X') = 0] \leq \Theta(1/\sqrt{J})$. This can easily be seen by noting i_0 the first index where $Y_i \neq X_i$. We will note $\bar{X}' = X' - X_{i_0}$, in the following we will use the fact that $a(X') = 0 \Leftrightarrow a(\bar{X}') = -a_{i_0} X_{i_0}$.[3]

$$\begin{aligned}
\Pr[a(Y') = 0 | a(X') = 0] &\leq \Pr[a(Y') = a(X') | a(X') = 0] \\
&\leq \Pr[Y_{i_0} a_{i_0} + a(\bar{Y}') = X_{i_0} a_{i_0} + a(\bar{X}') | a(X') = 0] \\
&\leq \max_t \Pr[(Y_{i_0} - X_{i_0}) a_{i_0} = t | a(\bar{X}') = -X_{i_0} a_{i_0}] \quad (1) \\
&\leq \max_{s,t'} \Pr[a_{i_0} = t' | a(\bar{X}') = s] \quad (2) \\
&\leq \max_{t'} \Pr[a_{i_0} = t'] \quad (3) \\
&\leq \Pr[a_{i_0} = 0] \leq \Theta(1/\sqrt{J})
\end{aligned}$$

(1) we start with $(Y_{i_0} - X_{i_0}) a_{i_0} = a(\bar{X}') - a(\bar{Y}')$, and then consider the maximum probability for all values $a(\bar{X}') - a(\bar{Y}')$.
(2) We consider the maximum probability for all values of $-X_{i_0} a_{i_0}$.
(3) a_{i_0} and $a(\bar{X}')$ are independent.

Hence, for all $X_1, Y_1, \ldots, Y_q$, we have

$$\begin{aligned}
\Pr[a_{X_1} = 0 \wedge a_{Y_1}, \ldots, a_{Y_q} \neq 0] &= \Pr[a_{X_1} = 0] \Pr[a_{Y_1}, \ldots, a_{Y_q} \neq 0 | a_{X_1} = 0] \\
&\geq \Theta(1/\tau\sqrt{\ell J}) \left(1 - \sum_{i=1}^{q} \Pr[a_{Y_i} = 0 | a_{X_1} = 0]\right) \\
&\geq \Theta(1/\tau\sqrt{\ell + 1J})(1 - q\Theta(1/\sqrt{J})).
\end{aligned}$$

Now we set $J = q^2$, to obtain the result. In that case the experiment success is lower-bounded by something linear in $1/(q\tau\sqrt{\ell+1})$. □

Acknowledgments

This work was supported in part by the European Commission through the ICT Program under Contract ICT-2007-216676 ECRYPT II.

References

1. Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In CRYPTO 2010, LNCS, pages 209–236. Springer, August 2010.
2. Masayuki Abe and Eiichiro Fujisaki. How to date blind signatures. In ASIACRYPT 1996, volume 1163 of LNCS, pages 244–251. Springer, November 1996.
3. Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In CRYPTO 2000, volume 1880 of LNCS, pages 271–286. Springer, August 2000.
4. Olivier Blazy, Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Signatures on randomizable ciphertexts. In PKC 2011, volume 6571 of LNCS. pages 403–422, Springer, 2010.

[3] $X \neq Y$ so i_0 exists, and thanks to the hardwired 1 we do not have to worry about Y' being a multiple of X'

5. Olivier Blazy, David Pointcheval, and Damien Vergnaud. Round-optimal privacy-preserving protocols with smooth projective hash functions. In TCC 2012, volume 7194 of LNCS, pages 94–111, Springer, 2012.
6. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In CRYPTO 2004, volume 3152 of LNCS, pages 41–55. Springer, August 2004.
7. Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In EUROCRYPT 2006, volume 4004 of LNCS, pages 427–444. Springer, May / June 2006.
8. David Chaum. Blind signatures for untraceable payments. In CRYPTO 1982, pages 199–203. Plenum Press, New York, USA, 1983.
9. Burgess Davis and David McDonald. An elementary proof of the local central limit theorem. *Journal of Theoretical Probability*, 8(3), jul 1995.
10. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In CRYPTO 2006, volume 4117 of LNCS, pages 60–77. Springer, August 2006.
11. Georg Fuchsbauer. Commuting signatures and verifiable encryption and an application to non-interactively delegatable credentials. Cryptology ePrint Archive, Report 2010/233, 2010.
12. Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round optimal blind signatures. In CRYPTO 2011, pages 630–648. Springer, August 2011.
13. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In EUROCRYPT 2008, volume 4965 of LNCS, pages 415–432. Springer, April 2008.
14. Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In CRYPTO 2008, volume 5157 of LNCS, pages 21–38. Springer, August 2008.
15. Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In EUROCRYPT 2006, volume 4004 of LNCS, pages 465–485. Springer, 2006.
16. David Naccache. Secure and *practical* identity-based encryption. Cryptology ePrint Archive, Report 2005/369, 2005.
17. Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In TCC 2006, volume 3876 of LNCS, pages 80–99. Springer, March 2006.
18. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.
19. Jae Hong Seo and Jung Hee Cheon. Beyond the limitation of prime-order bilinear groups, and round optimal blind signatures. In TCC 2012, volume 7194 of LNCS, pages 133–150. Springer, 2012.
20. Brent R. Waters. Efficient identity-based encryption without random oracles. In EUROCRYPT 2005, volume 3494 of LNCS, pages 114–127. Springer, May 2005.

A Blind Signatures

Definition 17 (Blind Signature Scheme). $\mathcal{BS} = (\mathsf{Setup}_{\mathcal{BS}}, \mathsf{KeyGen}_{\mathcal{BS}}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Verif}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, m, \sigma))$ where

- $\mathsf{Setup}_{\mathcal{BS}}(1^\lambda)$, where λ is the security parameter, generates the global parameters param_{bs} of the system;
- $\mathsf{KeyGen}_{\mathcal{BS}}(\mathsf{param}_{bs})$ generates a pair of keys $(\mathsf{pk}_{\mathcal{BS}}, \mathsf{sk}_{\mathcal{BS}})$;
- Signature Issuing: this is an interactive protocol between the algorithms $\mathcal{S}(\mathsf{sk}_{\mathcal{BS}})$ and $\mathcal{U}(\mathsf{pk}_{\mathcal{BS}}, m)$, for a message $m \in \{0,1\}^n$. It generates an output σ for the user: $\sigma \leftarrow \langle \mathcal{S}(\mathsf{sk}_{\mathcal{BS}}), \mathcal{U}(\mathsf{pk}_{\mathcal{BS}}, m) \rangle$.
- $\mathsf{Verif}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, m, \sigma)$ outputs 1 if the signature σ is valid with respect to m and $\mathsf{pk}_{\mathcal{BS}}$, 0 otherwise.

The security of a blind signature scheme is defined through two different notions, blindness and unforgeability. An adversary $\mathcal{U}$ against the unforgeability tries to generate $q_s + 1$ valid signatures after at most q_s complete interactions with the honest signer. The blindness condition is, on the other hand, linked to the signer. It states that a malicious signer $\mathcal{S}^*$ should be unable to decide which of two messages m_0, m_1 has been signed first in two executions with an honest user $\mathcal{U}$. In the following we note σ_b the signature on m_b. If $\mathcal{S}^*$ refuses to sign one of the input (*i.e.* $\sigma_i = \perp$), then the two resulting signatures are set to $\perp$, therefore he cannot have any advantage if he decides to prevent the normal game execution and he has to sign both inputs. We also define an unforgeability notion, which slightly differs from the original one [18], in the sense that we do not exclude malleability since we will eventually use randomizable signature. We thus count the number of distinct signed messages, which should not be larger than the number of interactions with the signer, whereas the initial definition counted the number of distinct message-signature pairs: $\mathcal{BS}$ is *unforgeable* if, for any polynomial adversary $\mathcal{U}^*$ (malicious user), the advantage $\mathsf{Succ}^{\mathsf{uf}}_{\mathcal{BS},\mathcal{U}^*}(\lambda)$ is negligible,

$\mathsf{Exp}^{\mathsf{bl}_b}_{\mathcal{BS},\mathcal{S}^*}(\lambda)$
$(\mathsf{pk}_{\mathcal{BS}}, m_0, m_1, st_{\texttt{FIND}}) \leftarrow \mathcal{S}^*(\texttt{FIND}, 1^\lambda)$;
$b \leftarrow \{0,1\}$;
$st_{\texttt{ISSUE}} \leftarrow \mathcal{S}^{*\langle\cdot,\mathcal{U}(\mathsf{pk}_{\mathcal{BS}},m_b)\rangle^1,\langle\cdot,\mathcal{U}(\mathsf{pk}_{\mathcal{BS}},m_{1-b})\rangle^1}(\texttt{ISSUE}, st_{\texttt{FIND}})$;
IF $\sigma_0 = \perp$ OR $\sigma_1 = \perp, (\sigma_0, \sigma_1) \leftarrow (\perp, \perp)$;
$b^* \leftarrow \mathcal{S}^*(\texttt{GUESS}, \sigma_0, \sigma_1, st_{\texttt{ISSUE}})$;
IF $b = b^*$RETURN 1 ELSE RETURN 0.

Figure 3. Blindness for blind signatures

$\mathsf{Exp}^{\mathsf{uf}}_{\mathcal{BS},\mathcal{U}^*}(\lambda)$
$(\mathsf{param}_{bs}) \leftarrow \mathsf{Setup}_{\mathcal{BS}}(1^\lambda)$;
$(\mathsf{pk}_{\mathcal{BS}}, \mathsf{sk}_{\mathcal{BS}}) \leftarrow \mathsf{KeyGen}_{\mathcal{BS}}(\mathsf{param}_{bs})$;
$\big((m_1, \sigma_1), \ldots, (m_{q_s+1}, \sigma_{q_s+1})\big) \leftarrow \mathcal{U}^{*\mathcal{S}^{q_s}(\mathsf{sk}_{\mathcal{BS}},\cdot)}(\mathsf{pk}_{\mathcal{BS}})$;
IF $\exists i \neq j, m_i = m_j$ OR $\exists i, \mathsf{Verif}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, m_i, \sigma_i) = 0$ RETURN 0
ELSE RETURN 1

Figure 4. Unforgeability for blind signatures (One-More Forgery)

where $\mathsf{Succ}^{\mathsf{uf}}_{\mathcal{BS},\mathcal{U}^*}(\lambda) = \Pr[\mathsf{Exp}^{\mathsf{uf}}_{\mathcal{BS},\mathcal{U}^*}(\lambda) = 1]$, in the security game presented in Figure 4. In this experiment, the adversary $\mathcal{U}^*$ can interact q_s times with the signing oracle $\mathcal{S}(\mathsf{sk}_{\mathcal{BS}}, \cdot)$ (hence the notation $\mathcal{U}^{*\mathcal{S}^{q_s}(\mathsf{sk}_{\mathcal{BS}},\cdot)}(\mathsf{pk}_{\mathcal{BS}})$) to execute the blind signature protocol: the adversary should not be able to produce more signatures on distinct messages than interactions with the signer. Our relaxation from the original *One-More Forgery* security comes from the fact that we will come up with randomizable signatures: from a message-signature pair, one can generate many signatures on the same message.

B Security Games of User-Friendly Partially Blind Signatures

$\mathsf{Exp}^{\mathsf{bl}_b}_{\mathcal{PBS},\mathcal{S}^*}(\lambda)$
$(\mathsf{pk}_{\mathcal{BS}}, m_0, m_1, st_{\texttt{FIND}}, \mathsf{info}_c, \mathsf{info}_s) \leftarrow \mathcal{S}^*(\texttt{FIND}, 1^\lambda)$;
$b \leftarrow \{0,1\}$;
$st_{\texttt{ISSUE}} \leftarrow \mathcal{S}^{*\langle\cdot,\mathcal{U}(\mathsf{pk}_{\mathcal{BS}},m_b)\rangle^1,\langle\cdot,\mathcal{U}(\mathsf{pk}_{\mathcal{BS}},m_{1-b},\mathsf{info}_c,\mathsf{info}_s)\rangle^1}(\texttt{ISSUE}, st_{\texttt{FIND}})$;
IF $\sigma_0 = \perp$ OR $\sigma_1 = \perp, (\sigma_0, \sigma_1) \leftarrow (\perp, \perp)$;
$b^* \leftarrow \mathcal{S}^*(\texttt{GUESS}, \sigma_0, \sigma_1, st_{\texttt{ISSUE}})$;
IF $b = b^*$RETURN 1 ELSE RETURN 0.

Figure 5. Blindness for User-Friendly Partially Blind signatures

$\mathcal{PBS}$ is *blind* if, for any polynomial adversary $\mathcal{S}^*$ (malicious signer), the advantage $\mathsf{Succ}^{\mathsf{bl}_b}_{\mathcal{PBS},\mathcal{S}^*}(k)$ is negligible, where $\mathsf{Succ}^{\mathsf{bl}}_{\mathcal{PBS},\mathcal{S}^*}(k) = |\Pr[\mathsf{Exp}^{\mathsf{bl}}_{\mathcal{PBS},\mathcal{S}^*}(k) = 1] - 1/2|$, in the security game presented in Figure 5. If $\mathcal{S}^*$ refuses to sign one of the input (*i.e.* $\sigma_i = \perp$), then the two resulting signatures are set to $\perp$, therefore he cannot have any advantage if he decides to prevent the normal game execution and he has to sign both inputs. $\mathcal{S}^*$ is able to chose both pieces of the public information, in the real case the signer can abort as long as the user's public information doesn't suit him, however the public information should be the same on both challenged message.

$\mathsf{Exp}^{\mathsf{uf}}_{\mathcal{PBS},\mathcal{U}^*}(\lambda)$
$(\mathsf{param}_{bs}) \leftarrow \mathsf{Setup}_{\mathcal{BS}}(1^\lambda)$;
$(\mathsf{pk}_{\mathcal{BS}}, \mathsf{sk}_{\mathcal{BS}}) \leftarrow \mathsf{KeyGen}_{\mathcal{BS}}(\mathsf{param}_{bs})$;
$\big((m_1, \mathsf{info}_{c,1}, \mathsf{info}_{s,1}, \sigma_1), \ldots, (m_{q_s+1}, \mathsf{info}_{c,q_s+1}, \mathsf{info}_{s,q_s+1}, \sigma_{q_s+1})\big) \leftarrow \mathcal{U}^{*\mathcal{S}^{q_s}(\mathsf{sk}_{\mathcal{BS}},\cdot)}(\mathsf{pk}_{\mathcal{BS}})$;
IF $\exists i \neq j, (m_i, \mathsf{info}_{c,i}, \mathsf{info}_{s,i}) = (m_j, \mathsf{info}_{c,j}, \mathsf{info}_{s,j})$ OR $\exists i, \mathsf{Verif}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, m_i, \mathsf{info}_{c,i}, \mathsf{info}_{s,i}, \sigma_i) = 0$ RETURN 0
ELSE RETURN 1

Figure 6. Unforgeability for User-Friendly Partially Blind signatures (One-More Forgery)

$\mathcal{PBS}$ is *unforgeable* if, for any polynomial adversary $\mathcal{U}^*$ (malicious user), the advantage $\mathsf{Succ}^{\mathsf{uf}}_{\mathcal{PBS},\mathcal{U}^*}(\lambda)$ is negligible, where $\mathsf{Succ}^{\mathsf{uf}}_{\mathcal{PBS},\mathcal{U}^*}(\lambda) = \Pr[\mathsf{Exp}^{\mathsf{uf}}_{\mathcal{PBS},\mathcal{U}^*}(\lambda) = 1]$, in the security game presented in Figure 6. In this experiment, the adversary $\mathcal{U}^*$ can interact q_s times with the signing oracle $\mathcal{S}(\mathsf{sk}_{\mathcal{BS}}, \cdot)$ (hence the notation $\mathcal{U}^{*\mathcal{S}^{q_s}(\mathsf{sk}_{\mathcal{BS}},\cdot)}(\mathsf{pk}_{\mathcal{BS}})$) to execute the user-friendly partially blind signature protocol: the adversary should not be able to produce more signatures on distinct tuple $(m, \mathsf{info}_c, \mathsf{info}_s)$ than interactions with the signer. Once again we consider the adversary has full control over the public information.

C Building Blocks

First, let us briefly sketch the basic building blocks: Groth-Sahai commitments, and a variation of the Waters signature. They both need a pairing-friendly environment $(p, \mathbb{G}, \mathbb{G}_T, e, g)$, where $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an admissible, non-degenerated, bilinear map, for two groups $\mathbb{G}$ and $\mathbb{G}_T$, of prime order p, generated by g and $g_t = e(g, g)$ respectively. From the following descriptions, it is clear that the different primitives are randomizable.

Groth-Sahai Commitments. In the following, several group elements will have to be committed so that proofs can be done on them. We will use perfectly hiding Groth-Sahai commitments: The commitment parameter is of the form $(\mathbf{u}_1 = (u_{1,1} = g^{x_1}, 1, g), \mathbf{u}_2 = (1, u_{2,2} = g^{x_2}, g), \mathbf{u}_3 = (u_{3,1}, u_{3,2}, u_{3,3})) \in (\mathbb{G}^3)^3$.

- To commit a group element $X \in \mathbb{G}$, one chooses three random scalars $r_1, r_2, r_3 \in \mathbb{Z}_p$ and sets $\mathcal{C}(X) := (c_1 = u_{1,1}^{r_1} \cdot u_{3,1}^{r_3}, c_2 = u_{2,2}^{r_2} \cdot u_{3,2}^{r_3}, c_3 = X \cdot g^{r_1+r_2} \cdot u_{3,3}^{r_3})$.
- To commit a scalar $x \in \mathbb{Z}_p$, one chooses two random scalars $\gamma_1, \gamma_2 \in \mathbb{Z}_p$ and sets (where $\odot$ is the component-wise multiplication) $\mathcal{C}'(x) := (u_{3,1}^x, u_{3,2}^x, (u_{3,3}g)^x) \odot \mathbf{u}_1^{\gamma_1} \odot \mathbf{u}_3^{\gamma_2} = (c'_1 = u_{3,1}^{x+\gamma_2} \cdot u_{1,1}^{\gamma_1}, c'_2 = u_{3,2}^{x+\gamma_2}, c'_3 = u_{3,3}^{x+\gamma_2} \cdot g^{x+\gamma_1})$.

A Groth Sahai proof will be a vector of group elements constructed to help the commitments to verify a pairing equation derived from the one verified by the associated plaintext. The idea is that with a regular initialization of the commitment parameters ($\mathbf{u}_3 = \mathbf{u}_1^\nu \odot \mathbf{u}_2^\mu$, for two random scalars $\nu, \mu \in \mathbb{Z}_p$), these commitments are perfectly binding and thus the proofs will be perfectly sound. The committed group elements can even be extracted if one knows x_1, x_2: $c_3/(c_1^{1/x_1} c_2^{1/x_2}) = X$, and $c'_3/({c'_1}^{1/x_1} {c'_2}^{1/x'_2}) = g^x$. However, if $\mathbf{u}_3$ is defined as $\mathbf{u}_3 = \mathbf{u}_1^\nu \odot \mathbf{u}_2^\mu \odot (1, 1, g^{-1}) = (u_{3,1} = u_{1,1}^\nu, u_{3,2} = u_{2,2}^\mu, u_{3,3} = g^{\nu+\mu-1})$, for two random scalars $\nu, \mu \in \mathbb{Z}_p$, the commitments are perfectly hiding and thus the proofs will perfectly hide the witnesses used in the instantiations. However the two parameter initializations being indistinguishable under the DLin assumptions, we will be able to use the perfectly binding setting in some simulations for the security proofs, whereas the real situation will use the perfectly hiding setting.

Waters Signature. The *Waters signature scheme* was formally described in [20]. It has been proven existentially unforgeable against chosen-message attacks under the CDH assumption.

- $\mathsf{Setup}(1^\lambda)$: The scheme is defined over a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, e, g)$. The parameters are a randomly chosen generator $h \xleftarrow{\$} \mathbb{G}$, and a vector $(u_0, \ldots, u_k) \xleftarrow{\$} \mathbb{G}^{k+1}$, those define the Waters function $\mathcal{F}$ such that for a message $M, \mathcal{F}(M) = u_0 \prod u_i^{M_i}$. We set $\mathsf{param} := (p, \mathbb{G}, \mathbb{G}_T, e, g, h, (u_0, \ldots, u_k))$.
- $\mathsf{SKeyGen}(\mathsf{param})$: Choose a random scalar $y \xleftarrow{\$} \mathbb{Z}_p$, which defines $\mathsf{vk} = Y = g^y$, and $\mathsf{sk} = Z = h^y$.
- $\mathsf{Sign}(\mathsf{sk}, M; s)$: To sign a message $M = (M_1, \ldots, M_k) \in \{0,1\}^k$, choose $s \xleftarrow{\$} \mathbb{Z}_p$ and define $\sigma = \big(\sigma_1 = Z \cdot \mathcal{F}(M)^s, \sigma_2 = g^s\big)$.
- $\mathsf{Verif}(\mathsf{vk} = Y, M, \sigma)$: Check whether $e(g, \sigma_1) \stackrel{?}{=} e(Y, h) \cdot e(\mathcal{F}(M), \sigma_2)$.

We also use another useful result on the Waters signature (like used in [15]):

Property 18 (Randomizability). The Waters signature scheme is randomizable: for a valid pair (M, σ), if we define $\sigma' = (\sigma_1 \cdot \mathcal{F}(M)^{s'}, \sigma_2 \cdot g^{s'})$, for a random scalar s', σ' is a random signature of M.

Proof. If the initial signature has been generated with s as random, the modified signature corresponds to the signature of M with $s + s'$ as random coins. Since this scalar lies in the group $\mathbb{Z}_p$, it leads to a perfectly random signature of M. □

Suffixed Waters Signatures. We will use Waters signatures, however instead of signing one message, we will sign, with the same additional parameters, a concatenation of 3 messages:

$$m = M||\mathsf{info}_c||\mathsf{info}_s = (M_1, \ldots, M_\ell, \mathsf{info}_1, \ldots, \mathsf{info}_f) \in \{0,1\}^k$$

D Multi-Blind Signature: Concatenation

With the previous building blocks, we will sign several commitments of $F_i = \mathcal{F}_i(M_i)$, instead of the standard $(\mathcal{U}, \mathcal{S})$ interactions we now have three main kind of users, $\mathcal{U}_i$, the user i will blind a commitment on $\mathcal{F}_i(M_i)$, $\mathcal{S}$ who signs the blinded message, and $\mathcal{T}$ the tallier who will verify/unblind/randomize this signature:

- $\mathsf{Setup}_{\mathcal{BS}}(1^\lambda)$: In a pairing-friendly environment $(p, \mathbb{G}, \mathbb{G}_T, e, g)$, the algorithm outputs a vector

$$\boldsymbol{u} = (u_0, (u_{i,1}, \ldots, u_{i,\lambda})_{1 \le i \le j}) \xleftarrow{\$} \mathbb{G}^{jk+1}$$

  where k is a polynomial in λ, and a generator $h \xleftarrow{\$} \mathbb{G}$. We define $\mathcal{F}_i(M_i) = \prod_\ell u_{i,\ell}^{m_{i,\ell}}$.
- $\mathsf{KeyGen}_{\mathcal{BS}}(\mathsf{param}_{bs})$: Choose $x \xleftarrow{\$} \mathbb{Z}_p$, which defines $\mathsf{pk}_{\mathcal{BS}} = Y = g^x$, and $\mathsf{sk}_{\mathcal{BS}} = Z = h^x$ and generates a pair of perfectly-binding Groth-Sahai generators, which define a decryption key $\mathsf{dk}_{\mathcal{BS}} = (x_1, x_2)$ composed of two scalars.
- $(\mathcal{U}_i, \mathcal{S}, \mathcal{T})$:
  - $\mathsf{Blind}_{\mathcal{BS}}(M, \mathsf{pk}_{\mathcal{BS}}; (r_1, r_2, r_3))$ (where we omit the subscripts i): For a message $M \in \{0,1\}^k$ and random scalars in $\mathbb{Z}_p$, define the commitment $c = \mathcal{C}(\mathcal{F}(M)) = (c_1, c_2, c_3)$. We also add, as before, proofs of validity of this commitment:
    - A proof Π_M of knowledge of M in c, the encrypted $\mathcal{F}(M)$, which consists of a bit-by-bit commitment $C_M = (\mathcal{C}'(M_1), \ldots, \mathcal{C}'(M_k))$ and proofs that each committed value is a bit. A proof that c_3 is well-formed *i.e.c* is a double linear encryption of the message M committed in C_M.
    - A proof Π_r containing the commitments $C_r = (\mathcal{C}(Y^{r_1+r_2}), \mathcal{C}(Y^{r_3}))$ together with proofs asserting that they are well-formed.
  - $\mathsf{Sign}_{\mathcal{BS}}(\mathsf{sk}_{\mathcal{BS}}, (c = (c_{1,i}, c_{2,i}, c_{3,i}), \Pi_i)_{1 \le i \le j}; s)$: To sign several commitments, first check if they are valid with respect to the proofs Π's, and after some randomization of those commitments, compute the global commitment $C = (\prod c_{1,i}, \prod c_{2,i}, u_0 \prod c_{3,i})$ which is still verifiable thanks to the previous (randomized) proofs, and then output $C = (C_1, C_2, C_3)$ and $\sigma = (C_1^s, C_2^s, Z \cdot C_3^s;\ g^s)$.

- $\mathsf{Verif}(\mathsf{pk}_{\mathcal{BS}}, (C = (C_1, C_2, C_3)), \sigma = (\sigma_1, \sigma_2, \sigma_3; \sigma_4))$: In order to check the validity of the signature, one checks whether the following equations are verified: $e(\sigma_1, g) = e(C_1, \sigma_4)$, $e(\sigma_2, g) = e(C_2, \sigma_4)$, and $e(\sigma_3, g) = e(h, \mathsf{pk}_{\mathcal{BS}}) \cdot e(C_3, \sigma_4)$
- $\mathsf{Unblind}_{\mathcal{BS}}(\mathsf{dk}_{\mathcal{BS}}, \mathsf{pk}_{\mathcal{BS}}, (c = (C_1, C_2, C_3), \Pi, \sigma))$: On a valid signature, knowing the decryption key (x_1, x_2), one can obtain $F = \mathcal{F}(M)$, and extract the message M from the bit-by-bit commitments. One can also extract the corresponding valid signature: $\sigma' = (\sigma'_1 = \sigma_3/(\sigma_1^{1/x_1}\sigma_2^{1/x_2}), \sigma'_2 = \sigma_4)$, which is a valid Waters signature on the concatenation of the messages.
- $\mathsf{Random}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, M, \sigma'; s')$: The latter can eventually be rerandomized to get $\Sigma = (\sigma'_1 \cdot \mathcal{F}(M)^{s'}, \sigma'_2 \cdot g^{s'})$.

– $\mathsf{Verif}_{\mathcal{BS}}(\mathsf{pk}_{\mathcal{BS}}, M, \sigma = (\sigma_1, \sigma_2))$: In order to check the validity of the signature, one checks whether: $e(\sigma_1, g) \stackrel{?}{=} e(h, \mathsf{pk}_{\mathcal{BS}}) e(\mathcal{F}(M), \sigma_2)$.

Theorem 19. *This multi-source blind signature scheme for concatenation is blind and unforgeable under the* CDH *and* DLin *assumptions: no adversary can generate more message-signature pairs on distinct messages, than the number of interactions with the signer.*

It directly follows from the previous result, combining the different partial Waters hashes into a global one does not weaken the security as we are still using single exponents on the u_i elements. Groth-Sahai proofs are in the perfectly binding setting to guarantee that each user really outputs Waters hash of their message on their generators and so no strange collision may occur and alter the final message.

E Asymmetric Version

All the previous schemes can be updated to work in asymmetric groups. The main, and only difference, comes from the Groth-Sahai commitments.

As symmetric bilinear groups are in general less efficient than *asymmetric* groups, we show how to instantiate our primitive with Groth-Sahai commitments in an asymmetric pairing-friendly group setting, relying on the SXDH assumption.

E.1 Assumptions

The security of Waters signatures in asymmetric bilinear groups was proven under the following variant of the CDH assumption, which states that CDH is hard in $\mathbb{G}_1$ when one of the random scalars is also given as an exponentiation in $\mathbb{G}_2$.

Definition 20 (The Advanced Computational Diffie-Hellman problem (CDH^+)). Let us be given two (multiplicative) groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order p with (g_1, g_2) as respective generators and e an admissible bilinear map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. The CDH^+ assumption states that given $(g_1, g_2, g_1^a, g_2^a, g_1^b)$, for random $a, b \in \mathbb{Z}_p$, it is hard to compute g_1^{ab}.

ElGamal encryption is secure under the DDH assumption. Since Groth-Sahai commitments are basically double ElGamal encryption, we assume SXDH, defined below.

Definition 21 (Decisional Diffie-Hellman Assumption (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 22 (Symmetric external Diffie-Hellman Assumption (SXDH) [6]). Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic groups of prime order, $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map. The SXDH assumption states that the DDH assumption holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

E.2 Groth-Sahai Commitments

As above, several elements will have to be committed so that proofs can be done on them. We will use SXDH-based Groth-Sahai commitments, which are a direct transposition of the previous ones in an asymmetric setting and replace double linear encryption by a double ElGamal one.

Proofs. This time, a Groth-Sahai proof, is a pair of elements $(\pi, \theta) \in \mathbb{G}_1^{2\times 2} \times \mathbb{G}_2^{2\times 2}$. As above, we will note $\langle x \rangle_1$ for a committed scalar x in $\mathbb{G}_1$, $\langle x \rangle_2$ for a committed scalar x in $\mathbb{G}_2$,or $\langle X \rangle$ for a committed group element X.

One has to pay attention to the fact that Groth-Sahai bit-by-bit proofs in SXDH require bits to be committed both in $\mathbb{G}_1$ and $\mathbb{G}_2$ and thus require to use 2 quadratic equations by bit.

E.3 Asymmetric Partially-Blind Signature with Perfect Blindness

The construction is really straightforward. If we follow the steps from the DLin-version: We will need 2 group elements for the commitment of M in $\mathbb{G}_1$, 4 group elements to commit Y_1, Y_2 in $\mathbb{G}_1$, the proofs will require 4 group elements in $\mathbb{G}_2$. We will need 6ℓ elements in each group to commit M and prove we indeed committed it bit-by-bit, and 2 extra group elements in $\mathbb{G}_2$ to prove c_2 is well-formed. The signatures on the committed elements will require 3 groups elements in $\mathbb{G}_1$ and one in $\mathbb{G}_2$. Therefore the overall scheme will require $(6\ell + 9, 6\ell + 7)$ group elements communication.