IET Information Security

Research Article

IET Journals
The Institution of Engineering and Technology

Leakage-resilient lattice-based partially blind signatures

ISSN 1751-8709
Received on 26th March 2019
Revised 17th June 2019
Accepted on 24th July 2019
E-First on 10th September 2019
doi: 10.1049/iet-ifs.2019.0156
www.ietdl.org

Dimitrios Papachristoudis[1] ✉, Dimitrios Hristu-Varsakelis[1], Foteini Baldimtsi[2], George Stephanides[1]

[1]*Computational Systems and Software Engineering Laboratory, Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece*

[2]*Department of Computer Science, George Mason University, Fairfax, Virginia, USA*

✉ *E-mail: dpapachristoudis@uom.edu.gr*

Abstract: Blind signature schemes (BSS) play a pivotal role in privacy-oriented cryptography. However, with BSS, the signed message remains unintelligible to the signer, giving them no guarantee that the blinded message he signed actually contained valid information. Partially BSS (PBSS) were introduced to address precisely this problem. In this study, the authors present the first leakage-resilient, lattice-based PBSS in the literature. The proposed construction is provably secure in the random oracle model and offers quasi-linear complexity w.r.t. key/signature sizes and signing speed. In addition, it offers statistical partial blindness and its unforgeability is based on the computational hardness of worst-case ideal lattice problems for approximation factors in $\tilde{O}(n^4)$ in dimension n. The proposed scheme benefits from the subexponential hardness of ideal lattice problems and remains secure even if a $(1 - o(1))$ fraction of the signer's secret key leaks to an adversary via arbitrary side-channels. Several extensions of the security model, such as honest-user unforgeability and selective failure blindness, are also considered and concrete parameters for instantiation are proposed.

1 Introduction

Typical digital signatures allow one party, termed the *signer*, to issue signatures on messages or documents, validating their authenticity. Such schemes primarily safeguard against impersonation of parties, tampering with messages, and repudiation. However, when it comes to privacy-sensitive applications such as electronic voting, e-cash, e-auctions, anonymous authentication via digital credentials, wireless sensor networks, or other cases in which preserving the confidentiality of a user is paramount, the functionality of conventional digital signatures falls short.

Blind signature schemes (BSS) are a variant of digital signatures that were pioneered by Chaum in 1982 [1], and have since become a central point of industrial and academic interest. BSS separate the owner of a message from the signer by allowing the owner of the message to interact with the signer and obtain a signature on it that remains unintelligible from the signer's view. The resulting signature can still be verified against the signer's public key, just like with typical digital signatures. However, nobody – including the signer himself – can link a message–signature pair to a signing transcript. As one would suspect though, such a high level of privacy has some grave drawbacks. First, by design, blind signatures provide perfect confidentiality for the receiving user with regards to the message being signed. As a result, blind signatures can potentially provide a gateway for committing 'perfect' crimes [2] such as money laundering, blackmailing, and so on. Second, blind signing provides no guarantee to the signer that the blinded message he signed is of the right 'format' or contains some valid information that should be included in the message (e.g.: the denomination of a digital coin, the date a voucher was issued etc.). Moreover, given that the only attributes over which the signer has control are those bound to his public key, we might end up in a case where multiple keys need to be managed, resulting to an increased complexity for both the signer and verifiers [which is even more problematic if devices with constrained memory (e.g. smart-cards) are being used [3]]. Consider for example a signer that issues blind signatures which expire at the end of the week, then the signer's public key needs to be updated every week, or consider the case of e-cash with multiple denominations: the signer/bank will need to use a different public key for each allowable coin denomination. These major shortcomings of blind signatures spurred the research community to invent primitives with features that could bypass these issues.

The two major models that have been proposed, in an effort to overcome these issues are: fair BSS [4, 5] and partially BSS (PBSS) [3, 6]. Fair blind signatures allow a trusted third party to revoke blindness in order to identify either the session during which a given signature was issued (session tracing), or a signature, given a signer's view of a specific session (signature tracing). On the other hand, partially blind signatures allow a signer and a user to include a commonly agreed upon piece of information (denoted *info*) to the signature. The key idea for achieving this in [6] was to adapt a method proposed in [7] by letting the signer use a secret key, along with two public keys, one of which includes *info*, with the help of a public hash function. As a result, the final signature is bound to these public keys and thus, to *info* as well. This approach has the benefit of greatly simplifying key management, because the signer only needs a single key in order to be able to include any auxiliary information (i.e. expiration date or denomination value). Note that PBSS do not immediately solve the problem of whether the blinded message to be signed is of the right format (this problem would be solved generically by including a zero-knowledge proof of knowledge on the format of the message), however, they provide an efficient way to make sure that the *info* part of the message included the necessary to application information and is of the right format. We would also like to mention that the more recent work of Rückert and Schröder [8] proposed a unified security model called *fair partially blind signatures* (FPBSS), which combines the security models of both aforementioned primitives into a single. Building a construction in that model would be ideal for real-world applications, balancing the individual needs of customers (blindness), service providers (partial control), and authorities (fairness), and is currently an open problem.

However, when designing secure cryptographic schemes, one has to be mindful of developments both in technology and also in the field of cryptanalysis. Indeed, following the formulation of Shor's algorithm [9] in 1994, the need for alternative hardness assumptions that remain intractable even in the presence of

quantum computers became as imperative as ever. By now, lattice-based cryptography is one of the predominant approaches for constructing provably secure and efficient cryptographic primitives that can withstand attacks even by a quantum computer. This is largely due to the fact that unlike number-theoretic hardness assumptions, there are no known algorithms for solving the lattice problems that are typically used at the foundation of cryptographic constructions, which has led to their conjectured intractability even against quantum computer attacks. Aside from *quantum-resistance*, lattices additionally have the unique feature of allowing for worst-case to average-case reductions. Phrased differently, a randomly selected (according to some distribution) problem instance is at least as hard to solve as some related lattice problem in the worst case. This feature not only allows us to reliably base security on worst-case hardness, but also greatly simplifies key selection for constructed cryptosystems. This extraordinary observation was first made by Ajtai in [10]. Moreover, lattice-based constructions are characterised by simplicity, efficiency, and parallelisability as one typically has to perform linear operations on vectors and matrices, as well as reductions modulo some small integer. Finally, lattice-based cryptography offers great versatility and is suitable for a plethora of advanced applications like: fully-homomorphic encryption, attribute-based encryption, general-purpose code obfuscation, hierarchical ID-based constructions, and much more. For a more detailed listing of applications, the reader is referred to surveys like [11].

1.1 Contributions and related work

A previous attempt to construct partially blind signatures from lattices was made in [12]. However, the construction of [12] does not prove partial blindness concretely and in fact seems to prove something weaker than the required notion as it relies on qualitative (if not ambiguous) properties of the signer that *cannot* be captured by the security model of PBSs. Furthermore, its scope is more limited compared to our proposal because it allows disclosures of the signed message which are acceptable in some applications (e-cash) but unacceptable in others (e-voting, e-auctions). Finally, the scheme of [12] is vulnerable to side-channel attacks, because of the use of discrete Gaussian sampling for the blind signing step [13–15].

We propose the first leakage-resilient, lattice-based PBSS in the literature. Our construction is inspired by the work of Rückert [16] which is currently the best known leakage-resilient BSS based on lattices. However, being a regular BSS, it is subject to the limitations discussed above. Our approach represents a significant step forward for PBSS because:

- First, because the vast majority of previous PBSS proposals [3, 6, 17–20] are based on number-theoretic assumptions, such as the hardness of large integer factorisation, or the computation of discrete logarithms. Unfortunately, the security of these schemes would be in jeopardy should a reasonable scale quantum computer be constructed, thanks to Shor's algorithm [9]. Consequently, all of these constructions are ill-suited for the post-quantum era.
- Second, although a tremendous amount of progress has been made in the design of conventional digital signatures from lattices over the past decade [21–30], there is a serious relative dearth when it comes to lattice-based *blind* signatures [16, 31] (the latter of which has recently been shown to be problematic [32]) despite their importance for privacy-preserving applications.
- Regarding efficiency, our construction is as efficient as the state-of-the-art lattice-based BSS in [16], both in terms of key sizes (ours are slightly smaller) and in computational complexity. However, our construction is not only one step closer to practical applications by allowing the inclusion of a commonly agreed piece of information in the final signature, but also relies on a milder – by a factor of n (the security parameter) – hardness assumption for the underlying worst-case lattice problem. This is important because one has to rely on as mild assumptions as possible in anticipation of attacks arising from emerging technologies. We show that all of the extensions considered in [16] are also satisfied by our scheme, along with an additional extension discussed in Section 5.1. In that case, we show that the efficient transformation that was proposed in [33] can also be used for PBSS, which we believe might be a result of interest on its own when designing such schemes.

1.1.1 Our technique and main challenges: Extending [16] to a PBSS was conjectured to be possible in [34]. However, no suggestions as to how this could be realised were given, and the problem was not formally addressed until now as it apparently involved several technical challenges. As per the security model of PBSS [6], we need to show that our scheme is complete, partially blind, and unforgeable. Unfortunately, lattices lack the algebraic structure that is present in (finite) cyclic groups, and which very naturally allows one to achieve partial blindness by simply computing the product/sum of any group element with a *random* group element. This problem can be rectified through rejection sampling [16, 23], which allows us to make the distributions of exchanged messages, independent of the respective messages that they 'hide'. However, this comes at the price of added complexity. Reducing this complexity is by no means trivial: being able to avoid/simplify rejection sampling would in turn impact many other lattice-based constructions such as [16, 23, 26, 29]. The complexity introduced by rejection sampling makes all of the aforementioned security properties (as well as the extensions that we consider) non-trivial to achieve *simultaneously* because they are interconnected to one another. In particular:

- Completeness is hindered, meaning that even if both parties involved in the signature issuing protocol are honest, the protocol may need to be restarted. We address this issue in the same way as [16]. However, since it is possible for the signature issuing protocol to restart, it is important to make sure that both partial blindness and unforgeability hold, even across restarts.
- Regarding partial blindness, PBSS are built by combining the framework of [6] with witness-indistinguishable identification protocols. For this work, we will use (a slight variant of) the witness-indistinguishable identification scheme of [23] as a basis. However, due to the aforementioned rejection sampling strategy, it is not possible to apply the transformation of [6] in a straightforward manner. This is due to the fact that rejection sampling causes the coefficients of a blinded message to come from a larger set (roughly by a factor of at least n) than the original message's, whenever applied. This turns out to be problematic when we want to 'unblind' to produce the final signature. We address this issue by having the user send a 'shrinked' version of the blinded challenge to the signer (i.e. reducing it modulo the range of the challenge space's coefficients – typically, modulo 3), by carefully setting our scheme's multiple interconnected parameters, and analysing the distributions of messages exchanged between the two parties. Our scheme is shown to be partially blind and an important implication of our approach is obtaining a milder by n hardness assumption for our scheme's unforgeability property. In addition, we employ a statistically hiding commitment scheme to make sure that partial blindness is preserved across protocol restarts.
- Proving unforgeability is also non-trivial because a malicious user might falsely claim that he failed to obtain a valid signature out of a protocol execution, thus causing the protocol to abort and potentially 'buying' himself multiple valid signatures (this scenario would obviously be catastrophic for applications like e-cash or e-voting). We address this issue by introducing a fourth move to our signature issuing protocol, which serves as a special proof of failure in case the protocol has to be restarted and is akin to [16]. As in [16], we need to show that a malicious user cannot obtain a valid signature out of an aborted protocol execution, unless he is able to solve a computationally hard lattice problem. However, as we will see in Section 4.3.3, this is considerably trickier to achieve compared to [16] because in our PBSS setting there are multiple scenarios which may cause the protocol's restart (in [16] there is only one). Nevertheless, our construction's security will be formally proven in the random

oracle model (ROM) [35] under standard worst-case lattice problems pertaining to ideals [36].
- Finally, with respect to leakage resilience, we will show that if we impose an additional requirement on the size of one of our scheme's parameters, then it is also resistant against key-leakage via arbitrary side-channels.

1.1.2 Relationship between the present work and impossibility results for BSS: In [37], the authors give an impossibility result for three-move BSS with the help of a meta-reduction (i.e. a reduction between reductions). Their approach plays the two security requirements of BSS (blindness and unforgeability) against each other, resulting in a proof that finding black-box reductions from unforgeability to non-interactive problems (like RSA, or discrete logarithm) is hard, unless the problems involved were already easy. Their work covers a broad class of BSS in the literature [1, 17, 38] and subsumes many prior impossibility results for BSS [39–41]. However, the main result of [37] does not apply to our construction. First, the results of [37] are given for BSS rather than PBSS which means that one would first have to show that a corresponding result also holds for PBSS. Second, [37] does not rule out reductions in the ROM [42, p. 3]. Third and most importantly, [37] only applies to BSS with *at most three* moves, that admit *statistical signature-derivation checks* (i.e. an observer can determine only from the public data and messages exchanged between a malicious signer and an honest user, whether the user successfully obtained a valid signature or not). In our four-move scheme however, it is impossible for one to tell whether the user truly obtained a valid signature or not within three moves because the user has not revealed all of the relevant information that he uses to produce his final signature. This is important because the components of the final signature must satisfy a certain relation but also fall within certain bounded domains for the signature to be deemed valid. This originates from our rejection sampling strategy and is in sharp contrast to previous number-theoretic BSS (and PBSS), where all of the final signature's components would always fall within some finite group (e.g. $\mathbb{Z}_N$ in the case of [1]), and thus checks like these would trivially be true due to finite group arithmetic rules. This is in accordance with an observation made by Fischlin and Schröder [37], stating that if the user sends a second message to the signer, which depends on his first message, then the resetting strategy of their meta-reduction cannot be applied. The same argument can also be used for [16]. Additionally, the fairly more recent results of [42] also do not apply to our work. The reason is that the results of that paper only concern schemes with a *unique-witness* relation between the public and secret key. While many constructions like the original Schnorr BSS fall under that category, our construction relies on a *many-to-one* witness relation between its public and secret keys (see Lemma 5 in Section 4.3.3).

1.2 Organisation of the paper

The remainder of the paper is organised as follows. Section 2 sets the required theoretical and notational groundwork. In Section 3, we describe the formal security model of leakage-resilient PBSS. In Section 4, we give a detailed description of our construction and show that it abides by the formal security model of PBSS, and that it is leakage resilient. Once we have established the baseline security, we examine additional security properties for our proposal.

2 Preliminaries

2.1 Notation

Throughout this paper, n will be used to denote the main security parameter. In order to formally define partially blind signatures, we adopt the following notation from [43]. Let $\mathscr{X}$ and $\mathscr{Y}$ be two algorithms. We denote by $(a, b) \leftarrow \langle \mathscr{X}(x), \mathscr{Y}(y) \rangle$, the joint execution of $\mathscr{X}$ and $\mathscr{Y}$ in an interactive way with private inputs x and y, respectively. The respective private outputs are a for $\mathscr{X}$ and b for $\mathscr{Y}$. By $\langle \mathscr{X}(x), \mathscr{Y}(y) \rangle^k$, we mean that the interaction can occur at most k times, where $k \in \mathbb{N}^+ \cup \{\infty\}$. Accordingly, if $\mathscr{Y}$ can invoke an unbounded number of executions of an interactive protocol with $\mathscr{X}$ in arbitrarily interleaved order, we write $\mathscr{Y}^{\langle \mathscr{X}(x), . \rangle^\infty}(y)$. Finally, $\mathscr{Y}^{\langle \mathscr{X}(x_0), . \rangle^1, \langle \mathscr{X}(x_1), . \rangle^1}(y)$ means that $\mathscr{Y}$ can invoke arbitrarily ordered executions with $\mathscr{X}(x_0)$ and $\mathscr{X}(x_1)$, but interact with each algorithm only once. An algorithm is considered *efficient* if it runs in probabilistic polynomial time. For asymptotics, we assume the standard Landau notation [44]. Additionally, we will use 'soft-$\mathcal{O}$' notation to ignore any polylogarithmic factors.

We will write $x \leftarrow_\$ S$ if x is sampled uniformly from a finite set S. If $\mathscr{A}$ is a probabilistic algorithm, we will write $y \leftarrow_\$ \mathscr{A}$ to denote that the output of $\mathscr{A}$ is assigned to y, and that $\mathscr{A}$ is running with randomly chosen coins. All logarithms are considered to be base 2. We denote the concatenation of strings or matrix columns by $\|$. A positive function $f(n)$ is called *negligible* in n if for any polynomial $p(n)$, there exists a $n_0 \in \mathbb{N}$, such that $f(n) \leq 1/p(n), \forall n \geq n_0$. A positive function $f(n)$ is called *noticeable* (or *non-negligible*), if there exists a positive polynomial $p(n)$ and a $n_0 \in \mathbb{N}$, such that $f(n) \geq 1/p(n), \forall n \geq n_0$. A function $f(n)$ is called *overwhelming* if $1 - f(n)$ is negligible.

Statistical distance provides us with a means of quantifying how 'far apart' two probability distributions (or random variables) are. Although there are many definitions of statistical distance in the literature, our analysis uses the following.

Definition 1: (Statistical distance): Let X and Y be two discrete random variables over a (countable) set S. *The statistical distance* $\Delta(X, Y)$ between X and Y is defined as $\Delta(X, Y) := \frac{1}{2} \sum_{v \in S} \left| \text{Prob}[X = v] - \text{Prob}[Y = v] \right|$.

A well-known property of statistical distance is that it does not increase if we apply a function f to its arguments [45].

Lemma 1: : Let S and T be finite sets, X and Y are random variables taking values in S, *and* $f : S \rightarrow T$ *be a function. Then* $\Delta(f(X), f(Y)) \leq \Delta(X, Y)$.

2.2 Rejection sampling

Rejection sampling is a technique that allows us to draw samples from arbitrarily complex probability distributions. In [23], it was shown how this technique can be utilised to construct a canonical identification scheme from lattices. As this technique is a crucial component to understanding our construction, we give here a brief overview.

Let $0 < A \leq B$ be two integer numbers. Now, consider the set of constant random variables $\{X_c := c : c \in \{-A, \ldots, A\}\}$ with respective probability mass functions: $f_{X_c}(x) := 1$, if $x = c$, and 0 otherwise. Furthermore, let Y be an independent, discrete uniform random variable, taking values in the set $\{-B, \ldots B\} \supseteq \{-A, \ldots, A\}$ and with probability mass function: $g_Y(y) := (1/2B + 1)$, if $y \in \{-B, \ldots, B\}$, and 0 otherwise.

We now define a new random variable Z_c as the sum of X_c and Y, for any fixed $c \in \{-A, \ldots, A\}$. Obviously, Z_c takes values in the set $\{-(A + B), \ldots, A + B\}$. The distribution h_{Z_c} of Z_c is thus the *convolution* of distributions f_{X_c} and g_Y, and its probability mass function is given from the formula [46]: $h_{Z_c}(z) = \sum_{k=-\infty}^{\infty} f_{X_c}(k) g_Y(z - k) = \sum_{k=-A+B}^{A+B} f_{X_c}(k) g_Y(z - k)$.
$= \text{Prob}[Y = z - c]$
Notice that if $|z - c| > B$, then the above probability is zero. On the other hand, if $|z - c| \leq B$, i.e. if $-B + c \leq z \leq B + c$, then the above probability equals $1/(2B + 1)$. Therefore, the probability mass function of h_{Z_c} is $h_{Z_c}(z) := (1/(2B + 1))$, if $z \in \{-B + c, \ldots, B + c\}$, and 0 otherwise.

Thus, h_{Z_c} is just a 'shifted' version of g_Y by c 'places'. It is not difficult to notice that Z_c is uniformly distributed over $\{-(B - A), \ldots, B - A\}, \forall c \in \{-A, \ldots, A\}$. Thus, if we compute $Z_c := X_c + Y = c + Y$, and only output the result if it falls within

$\{-(B-A), \ldots, B-A\}$ (and resample Y otherwise), then each value $z \in \{-(B-A), \ldots, B-A\}$ will be equally likely to occur. As a result, we can use this technique to 'hide' the value of c (in other words, Z_c is distributed independently of c). We will revisit this discussion more formally in Section 4.3.2.

2.3 Commitment schemes

Commitment schemes are fundamental cryptographic primitives that lie at the heart of many modern cryptographic protocols. Informally, they allow a party to commit to a certain value (or statement), while keeping the actual value hidden from all others, with the ability to reveal that value at a later point.

Definition 2: (Commitment schemes): Let $\mathrm{com}: \{0,1\}^* \times \{0,1\}^n \to \{0,1\}^*$ be a deterministic polynomial time algorithm, where n is a security parameter. A (non-interactive) commitment scheme consists of two protocols between two parties which are typically named 'sender' and 'receiver':

Commit phase: The sender commits to a value $\mu \in \{0,1\}^*$ by computing $C \leftarrow \mathrm{com}(\mu, r)$, where randomness $r \leftarrow_{\$} \{0,1\}^n$, and sends C to the receiver.
Reveal phase: The sender 'opens' commitment $C \leftarrow \mathrm{com}(\mu, r)$ by revealing the 'decommitment' parameter r to the receiver. The receiver can then verify that $C = \mathrm{com}(\mu, r)$.

Commitment schemes need to satisfy two properties: hiding and binding. The *hiding property* requires that C does not reveal any information about the committed message μ, whereas the *binding property* requires that no algorithm can substitute the committed message μ with some other message $\mu' \neq \mu$, in such a way that $C = \mathrm{com}(\mu' r) = \mathrm{com}(\mu, r')$, for some randomness $r' \in \{0,1\}^n$. A commitment scheme is (t, θ)-*hiding* (resp. *binding*) if no algorithm exists running in time at most t, that can break the hiding (resp. binding) property with a probability of at least θ. Both properties can be satisfied computationally or unconditionally. It has been shown that a commitment scheme cannot be unconditionally hiding and unconditionally binding at the same time [47]. For our construction, we will assume a statistically $\theta_{\mathrm{com}}^{(h)}$-hiding and computationally $(t_{\mathrm{com}}, \theta_{\mathrm{com}}^{(b)})$-binding commitment scheme. As with [16], we can use a lattice-based cryptographic hash function such as [48] as a message authentication code to construct a purely lattice-based scheme.

2.4 Lattices

A *lattice* is a set of points in n-dimensional space with a periodic structure. The easiest way to represent a lattice is as the set of all integer linear combinations $\Lambda = \left\{ \sum_{i=1}^{d} x_i \boldsymbol{b}_i \middle| x_i \in \mathbb{Z} \right\}$ of d linearly independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d \in \mathbb{R}^n$. These vectors are called a *basis* for the lattice Λ and are often represented as a matrix $\boldsymbol{B} = [\boldsymbol{b}_1 \parallel \ldots \parallel \boldsymbol{b}_d] \in \mathbb{R}^{n \times d}$. We will write $\Lambda = \Lambda(\boldsymbol{B})$ to express this fact. We say that the *rank* of the lattice is d and its *dimension* is n. If $d = n$, the lattice is called *full-rank*. One of the main computationally hard problems involving lattices is the shortest vector problem (SVP) [10].

Definition 3: (The approximate SVP – $\mathrm{SVP}_{\gamma}^{p}$*):* Let $\Lambda = \Lambda(\boldsymbol{B})$ be a lattice and $\gamma \geq 1$. Find a vector $v \in \Lambda \backslash \{\boldsymbol{0}\}$, such that $\| \boldsymbol{v} \|_p \leq \gamma \min_{\boldsymbol{w} \in \Lambda \backslash \{\boldsymbol{0}\}} (\| \boldsymbol{w} \|_p)$.

SVP is conjectured to remain computationally intractable for polynomial approximation factors, even by quantum algorithms [49].

Here, we will focus on a special family of lattices that possess additional algebraic structure, called *ideal lattices*. In particular, throughout this paper, $\boldsymbol{R}$ will denote the polynomial ring $\mathbb{Z}_q[x]/\langle \boldsymbol{f} \rangle$, where q is a prime and $\boldsymbol{f} \in \mathbb{Z}[x]$ is any monic, irreducible polynomial of degree n. For efficiency reasons, the preferred choice for $\boldsymbol{f}$ is $x^n + 1$, where n is a power of 2 (although the ring-structure induced by this choice of $\boldsymbol{f}$ allows for much shorter key-sizes and makes operations more efficient through the fast Fourier transform, it provides no further functionality [24, p. 2]). Furthermore, the ring of integers modulo q will be identified with the set

$$\left\{ -\frac{q-1}{2}, \ldots, \frac{q-1}{2} \right\}.$$

It is not hard to see that $R^m \cong \mathbb{Z}_q^{mn}, \forall m \in \mathbb{N}^+$ with vector addition corresponding to polynomial addition, and matrix–vector multiplication corresponding to the convolution product $\sum_{i=0}^{m-1} a_i b_i$ (modulo $\boldsymbol{f}$ and q) of polynomials in $\boldsymbol{R}$. We will identify any polynomial $g \in \boldsymbol{R}$ with its coefficient vector $\boldsymbol{g} = (g_0, \ldots, g_{n-1}) \in \mathbb{Z}_q^n$ (i.e. we will treat polynomials of $\boldsymbol{R}$ and vectors of $\mathbb{Z}_q^n$ as equivalent). Conventionally, we will denote vectors in $\boldsymbol{R}$ with boldface letters and m-tuples of vectors in $\boldsymbol{R}^m$ with boldface letters and a hat. We slightly abuse notation and define $\| \boldsymbol{g} \|_\infty := \max_i |g_i|$ and $\| \hat{\boldsymbol{g}} \|_\infty := \max_i (\| \boldsymbol{g}_i \|_\infty)$. A lattice corresponds to an ideal $I \subset \boldsymbol{R}$, iff every lattice vector is the coefficient vector of a polynomial in I. The SVP problem easily translates to ideal lattices and is called Ring-SVP.

The average-case problem upon which we will base our construction's security is that of finding short vectors in the kernel of the family $\mathscr{H}(\boldsymbol{R}, m)$ of module homomorphisms $h_{\hat{\boldsymbol{a}} \in \boldsymbol{R}^m}: \boldsymbol{R}^m \to \boldsymbol{R}, \hat{\boldsymbol{x}} \mapsto \hat{\boldsymbol{a}} \circledast \hat{\boldsymbol{x}} := \sum_{i=0}^{m-1} \boldsymbol{a}_i \boldsymbol{x}_i$, when restricting the domain to $D \subset \boldsymbol{R}$, i.e. restricting the coefficients of the input to $\{-d, \ldots, d\}$. This is the collision problem [36], which we now formally state.

Definition 4: (Collision problem): Given a function $h \leftarrow_{\$} \mathscr{H}(\boldsymbol{R}, m)$, the collision problem $\mathrm{Col}(\mathscr{H}(\boldsymbol{R}, m), D)$ is to find a distinct pair of preimages $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}) \in D^m \times D^m$ such that $h(\hat{\boldsymbol{x}}) = h(\hat{\boldsymbol{y}})$.

Evidently, h is linear over $\boldsymbol{R}^m$, i.e. it satisfies $h(\boldsymbol{a}\hat{\boldsymbol{x}} + \boldsymbol{b}\hat{\boldsymbol{y}}) = \boldsymbol{a}h(\hat{\boldsymbol{x}}) + \boldsymbol{b}h(\hat{\boldsymbol{y}}), \forall \boldsymbol{a}, \boldsymbol{b} \in \boldsymbol{R}$, and $\forall \hat{\boldsymbol{x}}, \hat{\boldsymbol{y}} \in \boldsymbol{R}^m$. The collision problem can trivially be shown to be as hard as Ring-SIS [45] in the average case and transitively, at least as hard as Ring-SVP in the worst case. The next theorem from [36] provides this connection.

Theorem 1: (Worst-case to average-case reduction): Let $D = \{\boldsymbol{f} \in \boldsymbol{R}: \| \boldsymbol{f} \|_\infty \leq d\}$, where $m > \log(q)/\log(2d)$, and $q \geq 4dmn\sqrt{n}\log(n)$. An adversary $\mathscr{A}$ that solves the $Col(h, D)$ problem, i.e. finds preimages $\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}} \in D^m$ such that $\hat{\boldsymbol{x}} \neq \hat{\boldsymbol{y}}$ and $h(\hat{\boldsymbol{x}}) = h(\hat{\boldsymbol{y}})$, can then use them to solve $\mathrm{Ring} - \mathrm{SVP}_{\gamma}^{\infty}$ with approximation factors $\gamma \geq 16dmn\log^2(n)$ in the worst case.

3 Syntax and security model of leakage-resilient PBSS

PBSS is an extension of regular blind signatures [1, 38, 50] and a simplification of FPBSS [8]. The security model for PBSS was formalised in [6]. A PBSS is comprised by three algorithms $(\mathrm{KG}, \mathrm{Sign} = \langle \mathscr{S}, \mathscr{U} \rangle, \mathrm{Vf})$, where Sign is an interactive protocol executed between $\mathscr{S}$ and $\mathscr{U}$. Their specification is the following:

Key generation: Algorithm $\mathrm{KG}(1^n)$ outputs a private signing key sk and a corresponding public verification key pk.
Signature issuing protocol: Protocol $\mathrm{Sign}(sk, \mu, \mathrm{info})$ jointly executes algorithms $\mathscr{S}(sk, \mathrm{info})$ and $\mathscr{U}(pk, \mu, \mathrm{info})$ in an interactive manner. The signer's private output is a view $\mathscr{V}$ consisting of all messages exchanged between the parties, and the user's private output is a signature σ on message μ and the common information *info* under sk. The common information *info* is agreed upon by the signer and the user prior to the protocol's execution and is assumed to be a common input to both parties. We also assume that the protocol generates a status message like 'ok' or $\perp$ for the signer, denoting success or failure, respectively.

Signature verification: Algorithm $\mathrm{Vf}(pk, \mu, \mathrm{info}, \sigma)$ returns 1 if σ is a valid signature on message μ and common information *info* under public key *pk*, and 0 otherwise.

Signer views can be interpreted as random variables and we will consider two views $\mathcal{V}_1$ and $\mathcal{V}_2$ 'equal' if no computationally unbounded algorithm $\mathcal{A}$ exists that distinguishes them with non-negligible probability.

A PBSS needs to satisfy three properties: completeness, partial blindness, and unforgeability [6, 34].

Completeness for PBSS is defined as in regular digital signatures, i.e. if both the signer and the user comply with the signature issuing protocol, then the user successfully obtains a valid signature with overwhelming probability.

Partial blindness generalises the notion of *blindness* [38, 50], and informally requires that it is infeasible for a malicious signer to link any valid signature to the exact instance of the signature issuing protocol in which it was created. A formal definition is given by means of the following experiment [6, 34]:

Definition 5: (Partial blindness): A partially blind signature scheme $\mathrm{PBSS} = (KG, \langle \mathcal{S}, \mathcal{U} \rangle, \mathrm{Vf})$ is partially blind if for any efficient algorithm $\mathcal{S}^*$ (working in modes find, issue, and guess), the probability that experiment $\mathrm{Exp}^{\mathrm{partiallyblind}}_{\mathcal{S}^*,\mathrm{PBSS}}(n)$ evaluates to 1 is negligibly close to 1/2 (as a function of n), where

Experiment $\mathrm{Exp}^{\mathrm{partiallyblind}}_{\mathcal{S}^*,\mathrm{PBSS}}(n)$

$(pk, sk) \leftarrow_{\$} \mathrm{PBSS}.\mathrm{KG}(1^n)$

$(\mu_0, \mu_1, \mathrm{info}, \mathrm{state}_{\mathrm{find}}) \leftarrow_{\$} \mathcal{S}^*(\mathrm{find}, 1^n)$

$b \leftarrow_{\$} \{0, 1\}$

$\mathrm{state}_{\mathrm{issue}} \leftarrow_{\$} \mathcal{S}^{*\langle \cdot, \mathcal{U}(pk, \mu_b, info)\rangle^1, \langle \cdot, \mathcal{U}(pk, \mu_{1-b}, \mathrm{info})\rangle^1}(\mathrm{issue}, \mathrm{state}_{\mathrm{find}})$

Let σ_b and σ_{1-b} be the private outputs of $\mathcal{U}(pk, \mu_b,$ info) and $\mathcal{U}(pk, \mu_{1-b},$ info), respectively.

If $\sigma_0 = \perp$ or $\sigma_1 = \perp$

$b' \leftarrow_{\$} \mathcal{S}^*(\mathrm{guess}, \perp, \perp, \mathrm{state}_{\mathrm{issue}})$

Else

$b' \leftarrow_{\$} \mathcal{S}^*(\mathrm{guess}, \sigma_0, \sigma_1, \mathrm{state}_{\mathrm{find}})$

Return 1 iff $b' = b$.

Notice that the notion of partial blindness closely resembles that of blindness [38], the only difference being that now there is an additional commonly known factor, *info*, that needs to be taken into account. In the above experiment, the malicious signer, $\mathcal{S}^*$, generates his public/secret keys via the scheme's key generation algorithm (we relax this requirement in Section 5.2). He then selects messages μ_0, μ_1 and common information *info* on his own (mode *find*). He then interacts with honest users $\mathcal{U}(pk, \mu_b, \mathrm{info})$ and $\mathcal{U}(pk, \mu_{1-b}, \mathrm{info})$, after a secret coin flip $b \leftarrow_{\$} \{0, 1\}$ (mode *issue*). If either user instance aborts before completion, the signer is merely notified of the event, but receives no signature. After seeing the unblinded signatures in the original order, the signer's task is to correctly guess b (mode *guess*). We further parameterise the definition of partial blindness. We will say that a PBSS is (t, θ)-*partially blind*, if there is no adversary $\mathcal{S}^*$, running in time at most t, that wins in the above experiment with advantage of at least θ, where $\mathcal{S}^*$'s advantage is defined as

$$\mathrm{Adv}^{\mathrm{partiallyblind}}_{\mathcal{S}^*,\mathrm{PBSS}} = \left|\mathrm{Prob}[\mathrm{Exp}^{\mathrm{partiallyblind}}_{\mathcal{S}^*,\mathrm{PBSS}}(n) = 1] - \frac{1}{2}\right|.$$

We will call a PBSS *statistically partially blind* if it is (∞, θ)-partially blind for a negligible θ, and *perfectly partially blind* if θ is 0.

Unforgeability of PBSS is stronger than the one defined for regular blind signatures [38, 50], since 'recombination' attacks should be ruled out [8]. Additionally, the adversarial user is allowed to select both the messages and the common information *info* that he queries, in an adaptive manner. Put another way, a malicious user should be unable to generate a valid signature for a new *info*, instead of just for a new message [8]. The notion of unforgeability of PBSS is defined in terms of the following game, which we derive from the more general game of [8], where $\mathcal{H}$ denotes a family of random oracles:

Definition 6: (Unforgeability of PBSS): An interactive partially blind signature scheme $\mathrm{PBSS} = (\mathrm{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathrm{Vf})$ is unforgeable if the following holds: for any efficient algorithm $\mathcal{U}^*$, the probability that experiment $\mathrm{Exp}^{omf}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$ evaluates to 1 is negligible (as a function of n), where

Experiment $\mathrm{Exp}^{\mathrm{omf}}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$

$H \leftarrow_{\$} \mathcal{H}(1^n)$

$(pk, sk) \leftarrow_{\$} \mathrm{PBSS}.\mathrm{KG}(1^n)$

For each info, let k_{info} denote the number of successful, complete interactions:

$((\mu_1, \mathrm{info}, \sigma_1), \ldots, (\mu_{k_{\mathrm{info}}+1}, \mathrm{info}, \sigma_{k_{\mathrm{info}}+1})) \leftarrow_{\$}$

$\mathcal{U}^{*H(\cdot), \langle \mathcal{S}(sk), \cdot\rangle^\infty}(pk)$

Return 1 iff

1. $\mu_i \neq \mu_j, \forall i, j = 1, \ldots, k_{\mathrm{info}} + 1$ with $i \neq j$, and
2. $\mathrm{PBSS}.\mathrm{Vf}(pk, \mu_i, \mathrm{info}, \sigma_i) = 1, \forall i = 1, \ldots, k_{\mathrm{info}} + 1$.

Note that in the above experiment, the adversarial user outputs $k_{\mathrm{info}} + 1$ valid message–signature pairs that correspond to a single *info*, where $0 \leq k_{\mathrm{info}} \leq q_{\mathrm{sig}}$ denotes the number of successful, complete interactions that took place. To further parameterise matters, we say that a PBSS is $(t, q_{\mathrm{Sig}}, q_H, \theta)$-*unforgeable* if there is no adversary $\mathcal{U}^*$, running in time at most t, making at most q_{Sig} signature queries and at most q_H hash oracle queries, that wins at the above experiment with probability at least θ.

Leakage-resilient cryptographic primitives are designed to remain secure even if an arbitrary, but bounded portion of the secret key (and/or other internal state information in general) of an honest party leaks to an adversary during computation. This augmentation of the notion of unforgeability helps safeguard against various forms of side-channel attacks, such as: timing attacks [13, 14], data remanence attacks, power-monitoring attacks [15], or implementations using poor random number generation. Unfortunately, the authors in [13, 14] provide clear evidence that cache timing attacks in particular are a practical threat to post-quantum cryptographic constructions. As a result, proving that a scheme is resistant against key leakage is a very important property if we want to consider long-term security, and constructions possessing it grant us a very high level of confidence when deploying them in practice.

To model leakage resilience in the context of unforgeable PBSS, we refer to [51], and grant the adversarial user access to a leakage oracle, Leak(.), in the above unforgeability experiment (our scheme satisfies the properties required by Katz and Vaikuntanathan [51]). The adversary can adaptively query a series of functions $f_i, i \in \{1, \ldots, \kappa\}$ to this oracle, and receive $f_i(sk)$. We consider the signer's secret state to consist solely of his secret key and that his secret key does not change over time. We also consider the same bounded leakage model as in [16]. More precisely, we impose the constraint $\sum_{i=1}^{n} |f_i(sk)| < \lambda(|sk|)$, where $\lambda = \lambda(\cdot)$ is a function of the length of the secret key, and dictates the amount of tolerable leakage. Of course, this extension only makes sense as long as $\lambda(|sk|) < \min\{|sk|, |\sigma|\}$, where $|\cdot|$ denotes bit-length, and σ is a signature. The experiment modelling leakage resilience for the unforgeability of PBSS is defined below.

Definition 7: (Leakage resilience of PBSS): An interactive partially blind signature scheme $\mathrm{PBSS} = (\mathrm{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathrm{Vf})$ is leakage-resilient with parameter λ, if the following holds: for any

efficient algorithm $\mathcal{U}^*$, the probability that experiment $\mathrm{Exp}^{\mathrm{omf},\lambda-\mathrm{Leak}}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$ evaluates to 1 is negligible (as a function of n), where

Experiment $\mathrm{Exp}^{\mathrm{omf},\lambda-\mathrm{Leak}}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$

$H \leftarrow_{\$} \mathcal{H}(1^n)$

$(pk, sk) \leftarrow_{\$} \mathrm{PBSS}.\mathrm{KG}(1^n)$

For each info, let k_{info} denote thenumber of successful, complete interactions:

$((\mu_1, \mathrm{info}, \sigma_1), \ldots, (\mu_{k_{info}+1}, \mathrm{info}, \sigma_{k_{\mathrm{info}}+1})) \leftarrow_{\$}$

$\mathcal{U}^{*H(.),\langle\mathcal{S}(sk),.\rangle^{\infty},\mathrm{Leak}(sk,.)}(pk)$

Let $f_1, \ldots, f_\kappa$ be the leakage queries of $\mathcal{U}^*$, each with output length λ_i.

Return 1 iff

1. $\mu_i \neq \mu_j, \forall i, j = 1, \ldots, k_{\mathrm{info}}+1$ with $i \neq j$,
2. $\mathrm{PBSS}.\mathrm{Vf}\ (pk, \mu_i, \mathrm{info}, \sigma_i) = 1, \forall i = 1, \ldots, k_{\mathrm{info}}+1$, and
3. $\sum_{i=1}^{\kappa} \lambda_i \leq \lambda(|sk|)$.

4 PBSS from Ring-SIS

We now present our lattice-based PBSS. Its time and space complexity are quasi-linear, $\tilde{\mathcal{O}}(n)$ in the security parameter, and its security will be proven in the ROM under the worst-case assumption that $\mathrm{Ring}-\mathrm{SVP}^{\infty}_{\gamma}$ is hard to solve in the ring $\boldsymbol{R}$ for $\gamma = \tilde{\mathcal{O}}(n^4)$. Notice that it is possible for our scheme to be instantiated with regular q-ary lattices and thus have its security based on regular SIS and SVP instead. Here we describe only the more efficient ideal lattice variant. Our scheme relies on carefully setting multiple interconnected parameters which are detailed in Table 1 (sorted by order of appearance in our construction). All sets are subsets of $\boldsymbol{R} = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ and are defined by means of a $l_\infty-$norm bound. The third column gives an indication of the asymptotic magnitude of the corresponding parameter/set w.r.t. the main security parameter n. The last column provides insight as to the role(s) that the corresponding parameter/set has in the interactive protocol, shown in its entirety in Fig. 1. Some sets introduce a completeness defect which can be rectified by increasing the value of parameter ϕ, which improves performance but requires a slightly stronger hardness assumption (by some constant factor). As in [16], we do not unwind the parameters d_s and d_ϵ in favour of making the proofs of some lemmas that involve them, easier to understand. In particular, for our scheme d_ϵ will be the constant 1, but one can increase it in order to be able to sign hash values of bit-length > $n\log(3)$.

4.1 Our construction

We go on to provide definitions for the triplet of algorithms $(\mathrm{KG}, \mathrm{Sign} = \langle\mathcal{S}, \mathcal{U}\rangle, \mathrm{Vf})$ comprising our PBSS. Sample parameters are given in Table 2.

- *Key generation*: $\mathrm{PBSS.KG}(1^n)$ chooses a secret key $\hat{\boldsymbol{s}} \leftarrow_{\$} D_s^m$ (see Table 1), and a homomorphic hash function $h \leftarrow_{\$} \mathcal{H}(\boldsymbol{R}, m)$. Next, it selects a function $com \leftarrow_{\$} \mathcal{C}(1^n)$ and a hash function $H \leftarrow_{\$} \mathcal{H}(1^n)$ mapping $\{0,1\}^* \to D_\epsilon \subset D$, where $\mathcal{C}(1^n)$ is a family of commitment schemes, mapping $\{0,1\}^* \times \{0,1\}^n \to \{0,1\}^n$. It also selects a public hash function $\mathcal{F}: \{0,1\}^* \to \boldsymbol{R}$ that maps arbitrary strings to a random public key, whose secret key is not known by anyone [6]. The algorithm computes the public key $\boldsymbol{S} \leftarrow h(\hat{\boldsymbol{s}})$ and gives the pair $(\hat{\boldsymbol{s}}, \boldsymbol{S})$ to the signer. For simplicity, we will treat $h, \mathrm{com}, H, \mathcal{F}$ and the rest of the parameters in Table 1 as globally known. Alternatively, the signer can set the parameter values and include them in the public key.
- *Signature issuing protocol*: The signature issuing protocol is described by the joint execution of algorithms $\mathcal{S}$ and $\mathcal{U}$ as depicted in Fig. 1. The signer's private input is his secret key $\hat{\boldsymbol{s}}$, whereas the user's private input is the message to-be-signed, μ. The common information *info* is assumed to be negotiated outside the signature scheme and is thus treated as common input to both parties. Eventually, the user obtains a signature $(r, \hat{\boldsymbol{z}}, \boldsymbol{\omega}, \hat{\boldsymbol{\sigma}}, \boldsymbol{\delta})$ for message μ and common information *info*. If the protocol needs to be restarted during step 2, the user only selects new $\boldsymbol{a} \leftarrow_{\$} D_a$ and $\boldsymbol{a}' \leftarrow_{\$} D_{a'}$, and repeats the operations that involve those, while keeping the same $r \in \{0,1\}^n$. However, if the protocol is aborted during either step 3 or step 5, the user must select a new r as well, to make the protocol executions

Table 1 Scheme parameters for main security parameter n

Parameter	Value	Asymptotics	Purpose
n	power of 2	—	main security parameter
d_s	positive integer constant $< q/(4n)$	$\mathcal{O}(1)$	secret key size, unforgeability
D_s	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_s\}$	$\mathcal{O}(1)$	secret key space
c_m	$> 1/\log(2d_s)$	$\tilde{\mathcal{O}}(1)$	witness indistinguishability, leakage resilience
m	$\lfloor c_m \log q \rfloor + 1$	$\Omega(\log(n))$	worst-case to average-case reduction
D_ϵ	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_\epsilon := 1\}$	$\mathcal{O}(1)$	hash output size
ϕ	positive integer constant ≥ 1	$\mathcal{O}(1)$	completeness, speed
D_a	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_a := \phi n d_\epsilon\}$	$\mathcal{O}(n)$	partial blindness
$D_{a'}$	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_{a'} := \phi n(d_a + d_\epsilon) + d_\epsilon\}$	$\mathcal{O}(n^2)$	partial blindness
G_ϵ	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_{a'} - (d_a + d_\epsilon)\}$	$\mathcal{O}(n^2)$	partial blindness
D_y	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_y := \phi m n^2 d_s d_\epsilon\}$	$\tilde{\mathcal{O}}(n^2)$	witness indistinguishability
G_*	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_{G_*} := d_y - n d_s d_\epsilon\}$	$\tilde{\mathcal{O}}(n^2)$	witness indistinguishability, completeness defect
D_β	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_\beta := \phi m n d_{G_*}\}$	$\tilde{\mathcal{O}}(n^3)$	partial blindness
G	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_G := d_\beta - d_{G_*}\}$	$\tilde{\mathcal{O}}(n^3)$	partial blindness, completeness defect
G_ω	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_\omega := d_a - d_\epsilon\}$	$\tilde{\mathcal{O}}(n)$	partial blindness, completeness defect
G_σ	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_\sigma := d_\beta - d_{G_*}\}$	$\tilde{\mathcal{O}}(n^3)$	partial blindness, completeness defect
G_δ	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_\delta := d_{a'} - d_\epsilon\}$	$\mathcal{O}(n^2)$	partial blindness, completeness defect
D	$\{\boldsymbol{f} \in \boldsymbol{R}: \|\boldsymbol{f}\|_\infty \leq d_D := d_{G_*} + d_\beta + n d_s d_\omega\}$	$\tilde{\mathcal{O}}(n^3)$	collisions under h
q	$\geq 4 d_D m n \sqrt{n} \log(n)$	$\tilde{\Theta}(n^4\sqrt{n})$	worst-case to average-case reduction

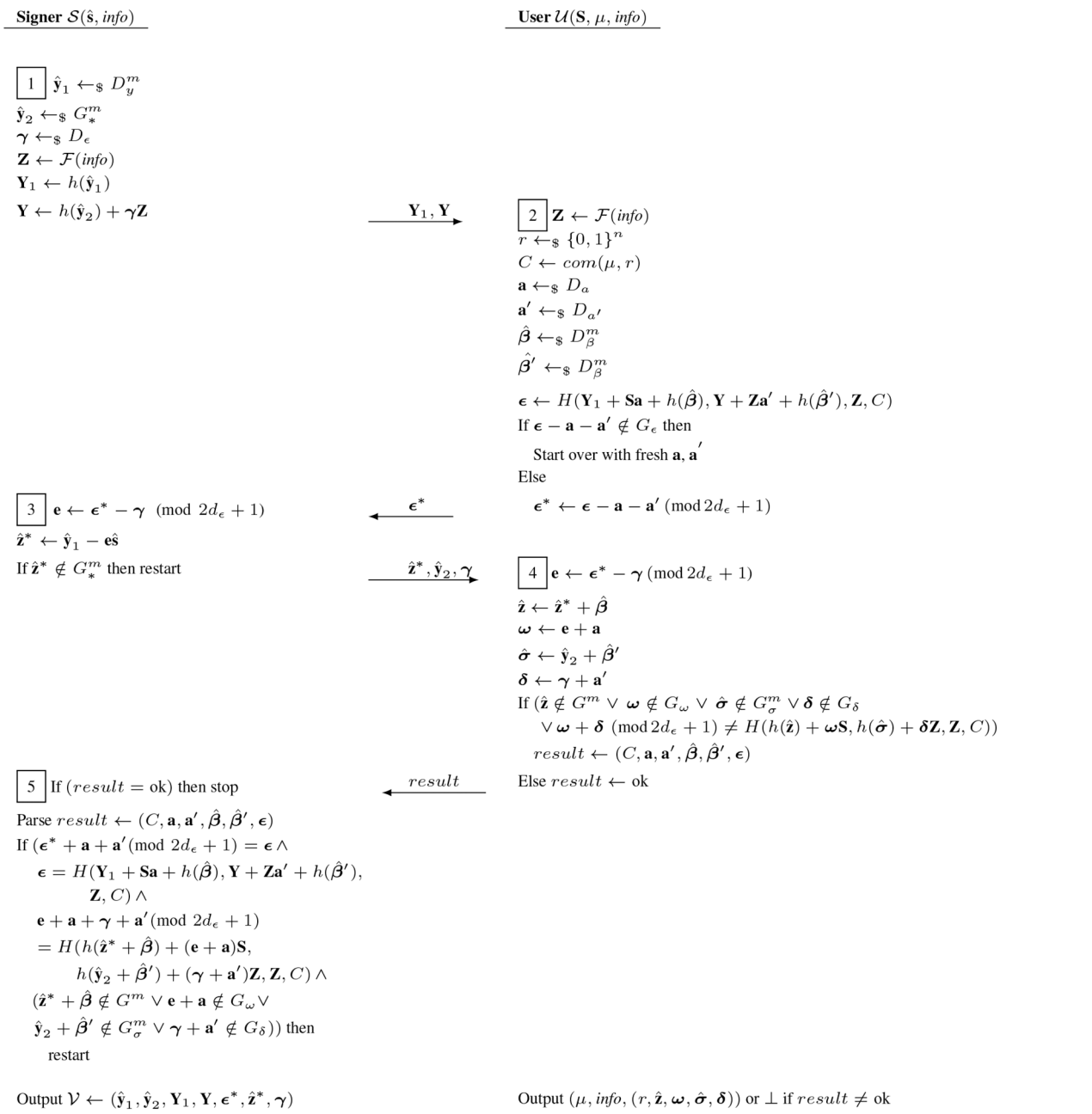

Fig. 1 *Five-step, four-move signature issuing protocol (steps shown in boxed numbers) for the proposed PBSS. All parameter and set definitions are given in Table 1. For brevity, we omit any verifications performed by the two parties w.r.t. the domains from which the protocol messages come from*

Table 2 Sample parameter instantiations for our PBSS

Parameter	Sample instantiations		
n (power of 2)	2048	2048	2048
q (prime $\approx n^7$)	$\approx 2^{77}$	$\approx 2^{77}$	$\approx 2^{77}$
ϕ	1	29	16
d_s	1	1	21,619
m	78	78	5
repetitions	148	**1.19**	1.37
secret key size	31.65 kB	31.65 kB	19.71 kB
public key size	19.71 kB	19.71 kB	19.71 kB
signature size	1868.8 kB	2260.6 kB	**168.3** kB
communication	3078.84 kB	3664.6 kB	320.72 kB

Parameters are set so that the collision problem is hard to solve [23, 52]. The parameters in the first column use the mildest hardness assumption, the set of the second column aims to reduce the number of required repetitions, and the third set aims to decrease the signature size, while keeping the number of required repetitions small (other trade-offs are also possible). For the second and third column, the optimisation goal is denoted in bold face. In all cases, the Hermite factor is taken to be 1.007, and the estimated security level is 92 bits [49, 53]. To decrease the expected number of repetitions ($e^{5/\phi}$ as we prove in Theorem 2), we need to increase the value of the parameter ϕ, thus sampling our masking vectors throughout the protocol from larger sets. Finally, as we discuss in Section 4.2, ϕ must not be a multiple of 3 (in case $d_\epsilon = 1$).

independent of one another. Finally, by means of step 5 the signer can thwart a cheating user who has obtained a valid signature but claims the contrary. In that case, the signer simply terminates the protocol, leaving the user with what he has obtained.

- *Signature verification*: $\text{PBSS}.\text{Vf}(\boldsymbol{S}, \mu, \text{info}, (r, \hat{\boldsymbol{z}}, \boldsymbol{\omega}, \hat{\boldsymbol{\sigma}}, \boldsymbol{\delta}))$ returns 1 as output iff $\hat{\boldsymbol{z}} \in G^m$, $\boldsymbol{\omega} \in G_\omega$, $\hat{\boldsymbol{\sigma}} \in G_\sigma^m$, $\boldsymbol{\delta} \in G_\delta$ and $\boldsymbol{\omega} + \delta(\bmod 2d_\epsilon + 1) = H(h(\hat{\boldsymbol{z}}) + \boldsymbol{\omega}\boldsymbol{S}, h(\hat{\sigma}) + \delta\mathcal{F}(\text{info}), \mathcal{F}(\text{info}),, \text{com}(\mu, r))$
  and 0 otherwise.

4.2 Protocol description

Our protocol is based on the three-move witness-indistinguishable identification protocol of [23], in which the signer proves knowledge of a secret key $\hat{\boldsymbol{s}} \in D_s^m$ such that $h(\hat{\boldsymbol{s}}) = \boldsymbol{S}$, where $\boldsymbol{S}$ is the corresponding public key. The signer also uses a second public key $\boldsymbol{Z}$ (the 'tag' public key), which is generated from the common information *info* with the help of a hash function. These two keys are used in conjunction by the signer to sign a message in such a way that the resulting protocol is witness-indistinguishable. We construct our protocol by combining [23] with the framework of [6].

Upon commencing, the signer selects random nonce vectors $\hat{\boldsymbol{y}}_1 \in D_y^m$ and $\hat{\boldsymbol{y}}_2 \in G_*^m$ and computes commitments $\boldsymbol{Y}_1 = h(\hat{\boldsymbol{y}}_1)$ and $\boldsymbol{Y} = h(\hat{\boldsymbol{y}}_2) + \gamma\boldsymbol{Z}$, where $\boldsymbol{Z} = \mathcal{F}(\text{info})$, which he then sends to the user. As is the case with all constructions that rely on the Fiat-Shamir heuristic [54], the user computes the challenge ϵ as a function (involving H) of $\boldsymbol{Y}_1, \boldsymbol{Y}$, the 'tag' public key $\boldsymbol{Z}$, and the message to-be-signed, μ, and then 'blinds' it by computing $\epsilon^* = \epsilon - \boldsymbol{a} - \boldsymbol{a}'(\bmod 2d_\epsilon + 1)$, before sending it to the signer. The signer computes $\boldsymbol{e} = \epsilon^* - \gamma(\bmod 2d_\epsilon + 1)$, and then the 'blinded' signature $\hat{\boldsymbol{z}}^* = \hat{\boldsymbol{y}}_1 - \boldsymbol{e}\hat{\boldsymbol{s}}$. As h is a homomorphism, the user can check that $h(\hat{\boldsymbol{z}}^*) = \boldsymbol{S}\boldsymbol{e} + \boldsymbol{Y}_1$ using public knowledge only. Finally, the user 'unblinds' the signature by computing $\hat{\boldsymbol{z}} = \hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}}$ and $\boldsymbol{\omega} = \boldsymbol{e} + \boldsymbol{a}$, as well as $\hat{\boldsymbol{\sigma}} = \hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}'$ and $\boldsymbol{\delta} = \boldsymbol{\gamma} + \boldsymbol{a}'$, which correspond to common information *info*. There are a few issues that need to be addressed at this point. First, the protocol must be complete. Second, the messages transmitted by the user must be distributed independently of the signed message μ, in order to achieve partial blindness. Finally, to prove unforgeability, we need to make sure that the messages transmitted by the signer do not leak information about his secret key to the user. All issues are addressed via rejection sampling [22, 23].

In step 2, we need to make sure that the blinded challenge ϵ^* that the user computes, leaks no information about the message being signed, and that it is uniformly distributed. This is necessary because $\boldsymbol{\omega} + \boldsymbol{\delta}(\bmod 2d_\epsilon + 1) = \epsilon$ (both $\boldsymbol{\omega}$ and $\boldsymbol{\delta}$ will be part of the final signature) and thus ϵ^* needs to hide ϵ. This is done in two steps: computing the blinded challenge, and then 'shrinking' it modulo the range of coefficients in D_ϵ. First, to hide ϵ we rejection-sample $\epsilon - \boldsymbol{a} - \boldsymbol{a}'$ to make sure that it falls within G_ϵ. For that purpose, $\boldsymbol{a}'$ will need to be picked from a relatively larger set than $\epsilon - \boldsymbol{a}$ to 'mask' the difference (and thus ϵ too). Otherwise, the user performs a 'local restart' by picking fresh $\boldsymbol{a}$ and $\boldsymbol{a}'$. The completeness defect introduced here can effectively be lowered to 0 because the user can repeat it locally. Second, provided that $\epsilon - \boldsymbol{a} - \boldsymbol{a}' \in G_\epsilon$, we have to ensure that $\epsilon^* := \epsilon - \boldsymbol{a} - \boldsymbol{a}'(\bmod 2d_\epsilon + 1)$ is also distributed uniformly over D_ϵ before sending it to the signer. We achieve this by imposing a restriction on the 'shape' of G_ϵ. For our case of $d_\epsilon = 1$, this can be achieved by requiring that the range of coefficients in G_ϵ is a multiple of $2d_\epsilon + 1 = 3$. However, notice that if we require that $2[d_{a'} - (d_a + d_\epsilon)] + 1 = 2(\phi^2 n^2 - 1) + 1 \equiv 0(\bmod 3)$, this is equivalent to $\phi^2 \equiv 2(\bmod 3)$, which has no solutions. To fix this, we set the upper bound for the coefficients in $D_{a'}$ to be slightly higher, i.e. $d_{a'} := \phi n(d_a + 1) + 1$ (or $d_{a'} := \phi n(d_a + d_\epsilon) + d_\epsilon$ in general). By following the same rationale as above, for the case of $d_\epsilon = 1$, we obtain the congruence $\phi^2 \equiv 1(\bmod 3)$, which is satisfied by all natural numbers that are not a multiple of 3. Thus, we need to select ϕ to be non-congruent to 0 modulo 3, which is not a steep requirement at all, given the natural density of such numbers. All of the parameter sets proposed in Table 2 satisfy this condition.

Upon receiving the 'shrinked' blinded challenge ϵ^*, the signer computes $\boldsymbol{e} \leftarrow \epsilon^* - \gamma \pmod{2d_\epsilon + 1}$. Notice that this computation is done modulo $2d_\epsilon + 1$ in correspondence to the computation of ϵ^* performed by the user during step 2. Since both ϵ^* and γ are uniform over D_ϵ (which is isomorphic to $\mathbb{Z}_{2d_\epsilon+1}^n$), $\boldsymbol{e}$ is also uniform over D_ϵ. The rationale behind the reduction modulo $2d_\epsilon + 1$ is to make the masking of $\boldsymbol{e}$ possible during the next step of the protocol (it is otherwise impossible to apply Lemmas 2 and 4). Next, we use rejection sampling to hide $\boldsymbol{e}\hat{\boldsymbol{s}}$ (and thus $\hat{\boldsymbol{s}}$) by adding to it a vector $\hat{\boldsymbol{y}}_1$ from a relatively larger set, compared to $\| \boldsymbol{e}\hat{\boldsymbol{s}} \|_\infty$, and outputting the result only if it falls within G_*^m. This results in $\hat{\boldsymbol{z}}^* = \hat{\boldsymbol{y}}_1 - \boldsymbol{e}\hat{\boldsymbol{s}}$ appearing to be uniform over G_*^m, despite actually being related to secret key $\hat{\boldsymbol{s}}$. However, if $\hat{\boldsymbol{z}}^* \notin G_*^m$, the protocol must be restarted. As we show in the next section, the number of required trials can be greatly reduced by increasing one of our scheme's parameters.

Finally, rejection sampling is used again in step 4 when the user attempts to 'unblind' the components of the final signature. More specifically, the user masks $\boldsymbol{e}, \gamma, \hat{\boldsymbol{z}}^*$, and $\hat{\sigma}$ with the help of $\boldsymbol{a}, \boldsymbol{a}', \hat{\boldsymbol{\beta}}$, and $\hat{\boldsymbol{\beta}}'$, respectively (which were prepared during step 2). Unfortunately, rejection sampling needs to be applied four times in total, which considerably decreases the user's chance of obtaining a signature without having to restart the protocol (see, e.g. the first column of Table 2). However, the completeness defect introduced during step 4 can also be ameliorated by increasing one of the scheme's parameters (namely, ϕ) at the expense of a slightly stronger hardness assumption. In particular, if any of $\hat{\boldsymbol{z}}^* + \beta, \boldsymbol{e} + \boldsymbol{a}, \hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}'$ or $\gamma + \boldsymbol{a}'$ does not fall within $G^m, G_\omega, G_\sigma^m$ or G_δ, respectively, the user sends $(C, \boldsymbol{a}, \boldsymbol{a}', \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}')$ to the signer, who then verifies whether the user has indeed failed to obtain a valid signature, or not. The signer does so by tracing the computations performed on the user's side. We stress that without this fifth final step, it is impossible for the signer to know whether the user successfully produced a valid signature during step 4, or not. Indeed, the signer does not know if $\hat{\boldsymbol{z}} \in G^m \wedge \boldsymbol{\omega} \in G_\omega \wedge \hat{\boldsymbol{\sigma}} \in G_\sigma^m \wedge \boldsymbol{\delta} \in G_\delta$, because he has never seen any of the masking terms $\hat{\boldsymbol{\beta}}, \boldsymbol{a}, \hat{\boldsymbol{\beta}}', \boldsymbol{a}'$ that were used to compute $\hat{\boldsymbol{z}}, \boldsymbol{\omega}, \hat{\boldsymbol{\sigma}}$, and $\boldsymbol{\delta}$, respectively. However, as we will prove in Section 4.3.3, the signer cannot be tricked into restarting the protocol by a malicious user, unless the latter is able to find collisions for h in $D \times D$. Additionally, for proving unforgeability we will require that *com* is binding. Finally, to prevent the signer from learning information about the signed message, μ, across restarts, we will require that *com* is also hiding.

4.3 Analysis and security

We now provide theorems and supporting lemmas showing that our proposed scheme satisfies the basic security requirements of leakage-resilient PBSS, namely: completeness, partial blindness, unforgeability, and leakage resilience. Once we have established the baseline security of our scheme, we consider further extensions of the security model.

4.3.1 Completeness: To prove the completeness of our proposed scheme, we require the following lemma from [16]. Informally, it guarantees that the number of restarts of our protocol is small, effectively constant.

Lemma 2: Let $k = \Omega(n), \boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k$ with arbitrary $\boldsymbol{a} \in \{\boldsymbol{v} \in \mathbb{Z}^k : \| \boldsymbol{v} \|_\infty \leq A\}$ and random

$\boldsymbol{b} \leftarrow_{\$} \{\boldsymbol{v} \in \mathbb{Z}^k: \| \boldsymbol{v} \|_\infty \leq B\}$. If $B \geq \phi k A$ for $\phi \in \mathbb{N}^+$, then $\text{Prob}_{\boldsymbol{b}}[\| \boldsymbol{a} + \boldsymbol{b} \|_\infty \leq B - A] > e^{-1/\phi} - o(1)$.

The next lemma from [16] is also required for our analysis, as it provides a bound (w.r.t. the infinity norm) for the product of any pair of polynomials in $\boldsymbol{R}$, when they are reduced modulo $x^n + 1$.

Lemma 3: Let $\boldsymbol{a}, \boldsymbol{b} \in \boldsymbol{R}$ be arbitrary polynomials. Then $\| \boldsymbol{ab} \bmod (x^n + 1) \|_\infty \leq n \| \boldsymbol{a} \|_\infty \| \boldsymbol{b} \|_\infty$.

Theorem 2: (Completeness): Let $g(n) = \omega(\log^5(n))$. Our PBSS is complete after at most $g(n)$ (or, an expected number of $e^{5/\phi}$) repetitions.

Proof: First, note that if no restarts occur, the protocol produces a valid signature. That is, for all honestly generated key pairs $(\hat{\boldsymbol{s}}, \boldsymbol{S})$, all messages $\mu \in \{0,1\}^*$, all common information info $\in \{0,1\}^*$, and all signatures $(r, \hat{\boldsymbol{z}}, \boldsymbol{\omega}, \hat{\boldsymbol{\sigma}}, \boldsymbol{\delta})$ we have:

$$\hat{\boldsymbol{z}} \in G^m, \boldsymbol{\omega} \in G_\omega, \hat{\boldsymbol{\sigma}} \in G_\sigma^m, \boldsymbol{\delta} \in G_\delta, \tag{1}$$

and

$$h(\hat{\boldsymbol{z}}) + \boldsymbol{\omega S} = h(\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}}) + (\boldsymbol{e} + \boldsymbol{a})\boldsymbol{S} = h(\hat{\boldsymbol{y}}_1 - \boldsymbol{e}\hat{\boldsymbol{s}} + \hat{\boldsymbol{\beta}}) + (\boldsymbol{e} + \boldsymbol{a})\boldsymbol{S} = \boldsymbol{Y}_1 + \boldsymbol{aS} + h(\hat{\boldsymbol{\beta}}). \tag{2}$$

Additionally, we have:

$$\boldsymbol{\omega} + \boldsymbol{\delta} = (\boldsymbol{e} + \boldsymbol{a}) + (\boldsymbol{\gamma} + \boldsymbol{a}') = (\boldsymbol{e} + \boldsymbol{\gamma}) + (\boldsymbol{a} + \boldsymbol{a}'). \tag{3}$$

Therefore, by reducing modulo $2d_\epsilon + 1$, we obtain:

$$\begin{aligned} \boldsymbol{\omega} + \boldsymbol{\delta} (\bmod 2d_\epsilon + 1) &= (\boldsymbol{e} + \boldsymbol{\gamma}) + (\boldsymbol{a} + \boldsymbol{a}') (\bmod 2d_\epsilon + 1) \\ &= \epsilon^* + \boldsymbol{a} + \boldsymbol{a}' (\bmod 2d_\epsilon + 1) = \epsilon. \end{aligned} \tag{4}$$

Thus, we have shown that:

$$\boldsymbol{\omega} + \boldsymbol{\delta} (\bmod 2d_\epsilon + 1) = H(h(\hat{\boldsymbol{z}}) + \omega \boldsymbol{S}, h(\hat{\boldsymbol{\sigma}}) + \boldsymbol{\delta}\mathcal{F}(\text{info}), \mathcal{F}(\text{info}), \text{com}(\mu, r)), \tag{5}$$

and $\text{PBSS}.\text{Vf}(\boldsymbol{S}, \mu, \text{info}, (r, \hat{\boldsymbol{z}}, \boldsymbol{\omega}, \hat{\boldsymbol{\sigma}}, \boldsymbol{\delta})))$ returns 1 as its output.

Next, we consider all possible restart cases and address the introduced completeness defect in each one of them:

Restarts occurring at step 2: Restarts during this step do not affect completeness at all, because the user just performs them locally. By applying Lemma 2, with $k = n, A = d_a + d_\epsilon$ and $B = d_{a'} = \phi n(d_a + d_\epsilon) + d_\epsilon$ to ensure that $\epsilon - \boldsymbol{a} - \boldsymbol{a}' \in G_\epsilon$, we obtain an expected number of trials which is constant $(e^{1/\phi})$, and which decreases as ϕ increases.

Restarts occurring at step 3: In step 3, the signer rejection-samples $\hat{\boldsymbol{z}}^* = \hat{\boldsymbol{y}}_1 - \boldsymbol{e}\hat{\boldsymbol{s}}$ to ensure that it lies in G_*^m. According to Lemma 3, $\| \boldsymbol{e}\hat{\boldsymbol{s}} \bmod (x^n + 1) \|_\infty \leq n d_s d_\epsilon$. Therefore, if we apply Lemma 2 with $k = mn$, $A = n d_s d_\epsilon$, and $B = d_y$, we conclude that the probability of success is $e^{-1/\phi}$ and the maximum number of trials is $\omega(\log(n))$ during this step. Thus, after an expected number of $e^{1/\phi}$ trials, the protocol successfully proceeds to step 4.

Restarts occurring after step 4: During the 'unblind phase' of step 4, the user requires that $\hat{\boldsymbol{z}}^* + \hat{\beta} \in G^m, \hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}' \in G_\sigma^m, \boldsymbol{e} + \boldsymbol{a} \in G_\omega$, and $\gamma + \boldsymbol{a}' \in G_\delta$. Otherwise, he requests a protocol restart from the signer. By applying Lemma 2 with $k = mn, A = d_{G_*}, B = d_\beta$ to $\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}}$, we obtain a success probability $e^{-1/\phi}$ and a maximum number of trials of $\omega(\log(n))$. Similarly, for $\hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}'$ with $k = mn, A = d_{G_*}, B = d_\beta = \phi mn d_{G_*}$, Lemma 2 yields a success probability $e^{-1/\phi}$ and a maximum number of trials of $\omega(\log(n))$. For $\boldsymbol{e} + \boldsymbol{a}$, Lemma 2 with $k = n, A = d_\epsilon$, and $B = d_a = \phi n d_\epsilon$ yields a success probability of $e^{-1/\phi}$. Finally, for $\gamma + \boldsymbol{a}'$, if we apply Lemma 2 with $k = n, A = d_a + d_\epsilon$, and $B = \phi n(d_a + d_\epsilon) + d_\epsilon$ yields a success probability of $e^{-1/\phi}$.

In total, after at most $g(n) = \omega(\log^5(n))$, or an expected number of $e^{5/\phi}$ restarts, the protocol is indeed complete. □

Remark 1: Note that all operations involved in our scheme (including restarts), as well as sizes of private keys, public keys and signatures are of quasi-linear complexity.

Remark 2: Also note that the parameter ϕ controls the number of trials. Increasing its value decreases the expected number of protocol restarts, and vice-versa.

4.3.2 Partial blindness: The following lemma from [16] is essential for proving the partial blindness of our scheme. It can be viewed as a rejection sampling lemma similar to that of [24]. Essentially, it states that all protocol messages are distributed independently of the message μ, and thus leak no information.

Lemma 4: Let $k \in \mathbb{N}$, $\boldsymbol{a}, \boldsymbol{a}', \boldsymbol{b} \in \mathbb{Z}^k$ with arbitrary $\boldsymbol{a}, \boldsymbol{a}' \in \{\boldsymbol{v} \in \mathbb{Z}^k: \| \boldsymbol{v} \|_\infty \leq A\}$, and a random $\boldsymbol{b} \leftarrow_{\$} \{\boldsymbol{v} \in \mathbb{Z}^k: \| \boldsymbol{v} \|_\infty \leq B\}$ for $B > A$. If $\boldsymbol{b}$ is such that $\max\{\| \boldsymbol{a} + \boldsymbol{b} \|_\infty, \| \boldsymbol{a}' + \boldsymbol{b} \|_\infty\} \leq B - A$, we define the random variables $\boldsymbol{c} \leftarrow \boldsymbol{a} + \boldsymbol{b}$ and $\boldsymbol{c}' \leftarrow \boldsymbol{a}' + \boldsymbol{b}$, otherwise, re-sample $\boldsymbol{b}$. Then, $\Delta(\boldsymbol{c}, \boldsymbol{c}') = 0$.

In proving that our construction is partially blind, we follow an approach similar to [16, p. 14] and show that all protocol messages exchanged between the user and the signer, along with the final output, are distributed independently from the signed message. For our analysis, we treat each of the exchanged messages and the output signature as random variables.

Theorem 3: (Partial blindness): If com is $\theta_{\text{com}}^{(h)}$ – hiding, then our PBSS is $(\infty, \theta_{\text{com}}^{(h)})$ – partially blind.

Proof: As per $\text{Exp}_{\mathcal{S}^*, \text{PBSS}}^{\text{partiallyblind}}(n)$ (see Section 3), the malicious signer chooses common information *info*, and two messages μ_0, μ_1, and then interacts with two honest users, $\mathcal{U}(\boldsymbol{S}, \mu_b, \text{info})$ and $\mathcal{U}(\boldsymbol{S}, \mu_{1-b}, \text{info})$, after a secret coin flip $b \leftarrow_{\$} \{0,1\}$.

Distribution of ϵ^*: Let $\epsilon_b^*, \epsilon_{1-b}^*$ be the first protocol messages of users $\mathcal{U}(\boldsymbol{S}, \mu_0, \text{info})$ and $\mathcal{U}(\boldsymbol{S}, \mu_1, \text{info})$, respectively. Both are of the form $\epsilon - \boldsymbol{a} - \boldsymbol{a}' (\bmod 2d_\epsilon + 1)$, with $\epsilon - \boldsymbol{a} \in \{\boldsymbol{f} \in \boldsymbol{R}: \| \boldsymbol{f} \|_\infty \leq d_a + d_\epsilon\}$ and $\boldsymbol{a}'$ is distributed uniformly over $D_{a'}$. First, notice that by Lemma 4 with $k = n, A = d_a + d_\epsilon$ and $B = d_{a'}$, it follows that $\Delta(\epsilon_b - \boldsymbol{a}_b - \boldsymbol{a}'_b, \epsilon_{1-b} - \boldsymbol{a}_{1-b} - \boldsymbol{a}'_{1-b}) = 0$. By applying Lemma 1 to random variables $\epsilon_b - \boldsymbol{a}_b - \boldsymbol{a}'_b$ and $\epsilon_{1-b} - \boldsymbol{a}_{1-b} - \boldsymbol{a}'_{1-b}$, with $f(X) = X (\bmod 2d_\epsilon + 1)$, we have $\Delta(\epsilon_b^*, \epsilon_{1-b}^*) = 0$.

Distribution of $\hat{\boldsymbol{z}}$: Let $\hat{\boldsymbol{z}}_0, \hat{\boldsymbol{z}}_1$ be part of the final output of $\mathcal{U}(\boldsymbol{S}, \mu_0, \text{info})$ and $\mathcal{U}(\boldsymbol{S}, \mu_1, \text{info})$, respectively; Note that both are of the form $\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}}$, for $\hat{\boldsymbol{z}}^* \in G_*^m$ and $\hat{\boldsymbol{\beta}} \leftarrow_{\$} D_\beta^m$. Additionally, both $\hat{\boldsymbol{z}}_0$ and $\hat{\boldsymbol{z}}_1$ lie in G^m because the users perform rejection sampling (step 4) on these random variables. Therefore, their coefficients are bounded in absolute value by $d_\beta - d_{G_*}$. From Lemma 4 with $k = mn, A = d_{G_*}$ and $B = d_\beta$, we infer that $\Delta(\hat{\boldsymbol{z}}_0, \hat{\boldsymbol{z}}_1) = 0$.

Distribution of $\boldsymbol{\omega}$: Let $\boldsymbol{\omega}_0, \boldsymbol{\omega}_1$ be part of the final output of $\mathcal{U}(\boldsymbol{S}, \mu_0, \text{info})$ and $\mathcal{U}(\boldsymbol{S}, \mu_1, \text{info})$, respectively. Both are of the form $\boldsymbol{e} + \boldsymbol{a}$, for $\boldsymbol{e} \in D_\epsilon$ and $\boldsymbol{a} \leftarrow_{\$} D_a$. Additionally, both $\boldsymbol{\omega}_0$ and $\boldsymbol{\omega}_1$ lie in G_ω because the users perform rejection sampling (during step 4) on these random variables. Therefore, their coefficients are bounded in absolute value by $d_a - d_\epsilon$. By applying Lemma 4 with $k = n, A = d_\epsilon$ and $B = d_a = \phi n d_\epsilon$, we infer that $\Delta(\omega_0, \omega_1) = 0$.

Distribution of $\hat{\sigma}$: Let $\hat{\sigma}_0, \hat{\sigma}_1$ be part of the final output of $\mathcal{U}(\boldsymbol{S}, \mu_0, \text{info})$ and $\mathcal{U}(\boldsymbol{S}, \mu_1, \text{info})$, respectively. Both are of the form

$\hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}'$, for $\hat{\boldsymbol{y}}_2 \in G_*^m$ and $\hat{\boldsymbol{\beta}}' \leftarrow_\$ D_\beta^m$. Additionally, both $\hat{\boldsymbol{\sigma}}_0$ and $\hat{\boldsymbol{\sigma}}_1$ lie in G_σ^m because the users perform rejection sampling (during step 4) on these random variables. Therefore, their coefficients are bounded in absolute value by $d_\beta - d_{G_*}$. By applying Lemma 4 with $k = mn$, $A = d_{G_*}$ and $B = d_\beta$, we infer that $\Delta(\hat{\boldsymbol{\sigma}}_0, \hat{\boldsymbol{\sigma}}_1) = 0$.

Distribution of $\boldsymbol{\delta}$: Let $\boldsymbol{\delta}_0, \boldsymbol{\delta}_1$ be part of the final output of $\mathcal{U}(\boldsymbol{S}, \mu_0, \text{info})$ and $\mathcal{U}(\boldsymbol{S}, \mu_1, \text{info})$, respectively. Both are of the form $\boldsymbol{\gamma} + \boldsymbol{a}'$, for $\boldsymbol{\gamma} \in D_\epsilon$ and $\boldsymbol{a}' \leftarrow_\$ D_{a'}$. Additionally, both $\boldsymbol{\delta}_0$ and $\boldsymbol{\delta}_1$ lie in G_δ because the users perform rejection sampling (step 4) on these random variables. Therefore, their coefficients are bounded in absolute value by $d_{a'} - d_\epsilon$. From Lemma 4 with $k = n, A = d_\epsilon$ and $B = d_{a'} = \phi n(d_a + d_\epsilon) + d_\epsilon > d_\epsilon$, we infer that $\Delta(\boldsymbol{\delta}_0, \boldsymbol{\delta}_1) = 0$.

Distribution of $\boldsymbol{Y}_1, \boldsymbol{Y}, \hat{\boldsymbol{y}}_2, \boldsymbol{\gamma}$ *and* r: These random variables are all either sampled uniformly at random from some domain, or distributed independently from the signed message μ. We note that $\boldsymbol{e}$ (which can be computed from ϵ^* and $\boldsymbol{\gamma}$) is also uniform over D_ϵ, since its computation is done within D_ϵ.

Restarts: Restarts are distinguished into two types: those that occur during step 2 and can be handled locally by the user, and those that occur after step 4 and cause the protocol to start over. Notice that we do not need to deal with restarts occurring in step 3, because they do not affect partial blindness as per experiment $\text{Exp}^{\text{partiallyblind}}_{\mathcal{S}^*,\text{PBSS}}(n)$.

- *Restarts during step 2*: As *com* is statistically hiding and the user selects a new set of $r, \boldsymbol{a}, \boldsymbol{a}', \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}'$ every time he performs a restart during step 2 of the signature issuing protocol, each protocol execution is statistically independent from any preceding execution. Therefore, our scheme is $(\infty, \theta^{(h)}_{com})$ – partially blind, since *com* is statistically $\theta^{(h)}_{com}$ – hiding.
- *Restarts caused after step 4*: The user submits $(C, \boldsymbol{a}, \boldsymbol{a}', \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}', \epsilon)$ to the signer. The signer is then able to trace the computations performed on the user's side and determines whether a restart is truly necessary. Note that the signer works with the commitment, C, instead of the original message, μ. Again, due to *com*'s statistical hiding property, μ remains statistically hidden from the signer, since he does not possess the corresponding decommitment parameter r which would allow him recovery of μ. Thus, our scheme achieves statistical instead of perfect partial blindness. □

Remark 3: Based on the previous discussion, if *com* is perfectly hiding (i.e. $\theta_{\text{com}} = 0$), then PBSS is partially blind in a perfect sense, whereas if *com* is statistically hiding, PBSS is partially blind in a statistical sense. In either case, a malicious signer only gains a negligible amount of information from protocol restarts, at best.

4.3.3 Unforgeability: The generalised forking lemma from [55] is a probabilistic result that lies at the core of proving the unforgeability of our scheme, and we include it in the Appendix. Additionally, to simulate the signing oracle in the unforgeability experiment of Section 3, we will also need two supporting lemmas. The first is from [16] and it states that for each public key $\boldsymbol{S}$ in our protocol, there exist (with overwhelming probability) at least two distinct corresponding secret keys $\hat{\boldsymbol{s}}, \hat{\boldsymbol{s}}'$.

Lemma 5: Let $h \in \mathcal{H}(\boldsymbol{R}, m)$. For every secret key $\hat{\boldsymbol{s}} \leftarrow_\$ D_s^m$, there exists (with overwhelming probability) a second $\hat{\boldsymbol{s}}' \in D_s^m \setminus \{\hat{\boldsymbol{s}}\}$ with $h(\hat{\boldsymbol{s}}) = h(\hat{\boldsymbol{s}}')$.

The next lemma is based on Lemma 3.7 from [16], suitably adapted for our construction (its proof can be found in the Appendix). Informally, it states that if we interpret the components of a (malicious) user's view as random variables, then the user is unable to tell which of (at least) two possible keys $\hat{\boldsymbol{s}}, \hat{\boldsymbol{s}}' \in h^{-1}(\boldsymbol{S}) \cap D_s^m$ was used during the signature issuing protocol, except with negligible advantage.

Lemma 6: **:** Let $h \in \mathcal{H}(\boldsymbol{R}, m)$ and $\boldsymbol{S} \in \boldsymbol{R}$. For any message μ and any two distinct secret keys $\hat{\boldsymbol{s}}, \hat{\boldsymbol{s}}' \in D_s^m$ with $h(\hat{\boldsymbol{s}}) = h(\hat{\boldsymbol{s}}')$, the resulting protocol views $(\boldsymbol{Y}_1, \boldsymbol{Y}, \epsilon^*, \hat{\boldsymbol{z}}^*, \hat{\boldsymbol{y}}_2, \boldsymbol{\gamma})$ and $(\boldsymbol{Y}'_1, \boldsymbol{Y}', \epsilon^{*'}, \hat{\boldsymbol{z}}^{*'}, \hat{\boldsymbol{y}}_{2'}, \boldsymbol{\gamma}')$ are witness-indistinguishable.

We now prove that our construction is unforgeable, provided that the commitment scheme is binding, and the collision problem $Col(\mathcal{H}(\boldsymbol{R}, m, D))$ being hard.

Theorem 4: (Unforgeability): Let Sig denote the signature issuing oracle and H the hashing oracle. Let T_{Sig} and T_H denote the cost functions for simulating the oracles Sig and H, respectively, and let $0 \le c < 1$ be the probability of restarting the protocol. Our PBSS is $(t, q_{\text{sig}}, q_H, \theta)$-unforgeable if com is $(t', \theta/2)$-binding, and $\text{Col}(\mathcal{H}(\boldsymbol{R}, m, D))$ is $(t', \theta_{\text{overall}}/2)$-hard, where $t' = t + q_H^{q_{\text{sig}}}(q_{\text{sig}} T_{\text{sig}} + q_H T_H)$ and θ_{overall} is noticeable if θ is noticeable.

Proof: Let $\mathcal{A}$ be an efficient forger who successfully breaks unforgeability within time t and with noticeable probability, θ. By exploiting $\mathcal{A}$'s capability of forging signatures in a black-box manner, we will construct a simulator $\mathcal{B}$, such that $\mathcal{B}$ either breaks the binding property of *com*, or solves the collision problem.

Setup: Simulator $\mathcal{B}$ flips a coin $b \leftarrow_\$ \{0, 1\}$. If $b = 0$, $\mathcal{B}$ selects $h \leftarrow_\$ \mathcal{H}(\boldsymbol{R}, m)$. Otherwise, it is given the description of h as input. $\mathcal{B}$ initialises a list $L_H \leftarrow \varnothing$ of query–hash pairs of the form $(\boldsymbol{R} \times \boldsymbol{R} \times \boldsymbol{R} \times \{0, 1\}^*, D_\epsilon)$ and a list $L_{\text{Sig}} \leftarrow \varnothing$ of message–signature pairs of the form $(\{0, 1\}^* \times \{0, 1\}^*, G^m \times G_\omega \times G_\sigma^m \times G_\delta)$. It then picks $\hat{\boldsymbol{s}} \leftarrow_\$ D_s^m$ and computes $\boldsymbol{S} \leftarrow h(\hat{\boldsymbol{s}})$. Moreover, $\mathcal{B}$ randomly pre-selects random oracle answers $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{q_H} \leftarrow_\$ D_\epsilon$, a random tape ρ, and runs $\mathcal{A}(\boldsymbol{S}; \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{q_H}; \rho)$ in a black-box way.

RO queries: On input $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{Z}, C)$, $\mathcal{B}$ determines if $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{Z}, C)$ has previously been queried to H by checking whether $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{Z}, C) \in L_H$. If the answer is affirmative, $\mathcal{B}$ returns the same output ϵ as before, to remain consistent. Otherwise, $\mathcal{B}$ returns the first unused $\boldsymbol{h}_i$ and stores $((\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{Z}, C), \boldsymbol{h}_i)$ in L_H.

PBS queries: $\mathcal{B}$ acts as the signer according to the protocol in Fig. 1 and fills in L_{Sig} after $\mathcal{A}$ produces his output.

Forgery: Since adversary $\mathcal{A}$ is efficient, he eventually stops, outputting:
$(\mu_1, \text{info}, (r_1, \hat{\boldsymbol{z}}_1, \boldsymbol{\omega}_1, \hat{\boldsymbol{\sigma}}_1, \boldsymbol{\delta}_1)), \ldots, (\mu_j, \text{info}, (r_j, \hat{\boldsymbol{z}}_j, \boldsymbol{\omega}_j, \hat{\boldsymbol{\sigma}}_j, \boldsymbol{\delta}_j))$, where $j = k_{\text{info}} + 1$ for pairwise distinct messages. If $b = 0$, the reduction tries to find two pairs $(\mu_1^*, \text{info}, (r_1^*, \hat{\boldsymbol{z}}^*, \boldsymbol{\omega}^*, \hat{\boldsymbol{\sigma}}^*, \boldsymbol{\delta}^*))$ and $(\mu_2^*, \text{info}, (r_2^*, \hat{\boldsymbol{z}}^*, \boldsymbol{\omega}^*, \hat{\boldsymbol{\sigma}}^*, \boldsymbol{\delta}^*))$ with $\mu_1^* \neq \mu_2^*$, and returns $(\mu_1^*, r_1^*), (\mu_2^*, r_2^*)$ to break *com*'s binding property. If no such pair is found, it simply aborts. If $b = 1$, the simulator locates a message–signature pair $((\mu^\dagger, \text{info}), (r^\dagger, \hat{\boldsymbol{z}}^\dagger, \boldsymbol{\omega}^\dagger, \hat{\boldsymbol{\sigma}}^\dagger, \boldsymbol{\delta}^\dagger))$, where $(\mu^\dagger, \text{info})$ has never been queried to the signing oracle. The algorithm computes $\boldsymbol{u}^\dagger = h(\hat{\boldsymbol{z}}^\dagger) + \boldsymbol{S}\boldsymbol{\omega}^\dagger$ and $\boldsymbol{v}^\dagger = h(\hat{\sigma}^\dagger) + \mathcal{F}(\text{info})\delta^\dagger$ and rewinds the adversary to the point where $(\boldsymbol{u}^\dagger, \boldsymbol{v}^\dagger, \mathcal{F}(\text{info}), \text{com}(\mu^\dagger, r^\dagger))$ was queried to the hashing oracle H. Let $1 \le I \le q_H$ be the index of that query. $\mathcal{B}$ then re-runs $\mathcal{A}(\boldsymbol{S}; \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{I-1}, \boldsymbol{h}'_I, \ldots, \boldsymbol{h}'_{q_H}; \boldsymbol{\rho})$ with new random responses to queries with index $\ge I$, but using the same random tape $\boldsymbol{\rho}$. Eventually, $\mathcal{A}$ will output a new forgery $((\mu^\ddagger, \text{info}), (r^\ddagger, \hat{\boldsymbol{z}}^\ddagger, \boldsymbol{\omega}^\ddagger, \hat{\boldsymbol{\sigma}}^\ddagger, \boldsymbol{\delta}^\ddagger))$ using the same random oracle query as in the first run (after polynomially bounded time because $\mathcal{A}$ is efficient and all of his queries are handled efficiently). $\mathcal{B}$ then returns $(\hat{\boldsymbol{z}}^\dagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\dagger, \hat{\boldsymbol{z}}^\ddagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\ddagger)$, if $\boldsymbol{\omega}^\dagger \neq \boldsymbol{\omega}^\ddagger$, as a solution to the collision problem and aborts otherwise (an event that as we will explain, occurs with negligible probability).

Analysis: $\mathcal{A}$'s environment is perfectly simulated and restarts occur with the same probability as in the original protocol. Therefore, $\mathcal{A}$ has no advantage whatsoever in distinguishing the simulation.

For $b = 0$, $\mathcal{B}(t', \theta/2)$ breaks *com*'s binding property, if $\mathcal{A}$ successfully attacks *com*'s binding property to break unforgeability.

For $b = 1$, we assume that $\mathscr{A}$ breaks unforgeability without attacking *com*. Since at least one of the produced signatures was not obtained via an interaction, the probability that $\mathscr{B}$ correctly guesses its index is at least $1/(k_{\text{info}} + 1)$. Next, notice that $\mathscr{A}$ can successfully predict the output of the random oracle H with probability $1/|D_\epsilon|$. By applying the general forking lemma of [55], we can determine that after rewinding, $\mathscr{A}$ is again successful in producing a forgery, using the same random oracle query as in the first run with probability

$$\theta_{\text{frk}} \geq \left(1 - c)(\theta - \frac{1}{|D_\epsilon|}\right)\left(\frac{\theta - 1/|D_\epsilon|}{q_H} - \frac{1}{|D_\epsilon|}\right),$$

where the additional $(1 - c)$ factor accounts for a potential restart during the second run. Therefore, with probability at least θ_{frk}, the following relation holds: $h(\hat{z}^\dagger) + \boldsymbol{S}\boldsymbol{\omega}^\dagger = h(\hat{z}^\ddagger) + \boldsymbol{S}\boldsymbol{\omega}^\ddagger$. This can equivalently be written as: $h(\hat{z}^\dagger - \hat{z}^\ddagger + \hat{\boldsymbol{s}}(\boldsymbol{\omega}^\dagger - \boldsymbol{\omega}^\ddagger)) = \boldsymbol{0}$. We observe that with overwhelming probability, $\boldsymbol{\omega}^\dagger \neq \boldsymbol{\omega}^\ddagger$. Indeed: $\boldsymbol{\omega}^\dagger = ((\epsilon^{*\dagger} - \boldsymbol{\gamma}^\dagger) \bmod (2d_\epsilon + 1)) + \boldsymbol{a}^\dagger = ((\epsilon^\dagger - \boldsymbol{a}^\dagger - \boldsymbol{a}'^\dagger - \gamma^\dagger) \bmod .$
$(2d_\epsilon + 1)) + \boldsymbol{a}^\dagger$
Similarly, we have:

$$\boldsymbol{\omega}^\ddagger = \boldsymbol{e}^\ddagger + \boldsymbol{a}^\ddagger = ((\epsilon^\ddagger - \boldsymbol{a}^\ddagger - \boldsymbol{a}'^\ddagger - \boldsymbol{\gamma}^\ddagger) \bmod (2d_\epsilon + 1)) + \boldsymbol{a}^\ddagger \quad (6)$$

By subtracting, we get:

$$\boldsymbol{\omega}^\dagger - \boldsymbol{\omega}^\ddagger = ((\epsilon^\dagger - \boldsymbol{a}^\dagger - \boldsymbol{a}'^\dagger - \boldsymbol{\gamma}^\dagger - \epsilon^\ddagger + \boldsymbol{a}^\ddagger + \boldsymbol{a}'^\ddagger + \boldsymbol{\gamma}^\ddagger) \bmod (2d_\epsilon + 1)) + \boldsymbol{a}^\dagger - \boldsymbol{a}^\ddagger. \quad (7)$$

If $\boldsymbol{\omega}^\dagger - \boldsymbol{\omega}^\ddagger = \boldsymbol{0}$, then $\epsilon^\ddagger - \boldsymbol{\gamma}^\ddagger (\bmod 2d_\epsilon + 1)$ is determined by polynomials selected by $\mathscr{A}$ and polynomials determined by $\mathscr{B}$ before rewinding. However, both $\epsilon^\ddagger$ and $\gamma^\ddagger$ are randomly selected by $\mathscr{B}$ after rewinding. Therefore, the probability that $\omega^\dagger = \omega^\ddagger$ is $1/|D_\epsilon| = 1/(2d_\epsilon + 1)^n$ which is negligible in n. Thus, $\omega^\dagger \neq \omega^\ddagger$ with overwhelming probability $1 - 1/|D_\epsilon|$.

Next, if $\omega^\dagger \neq \omega^\ddagger$ then with a probability of at least 1/4, we have $\hat{z}^\dagger - \hat{z}^\ddagger + \hat{\boldsymbol{s}}(\boldsymbol{\omega}^\dagger - \boldsymbol{\omega}^\ddagger) \neq \boldsymbol{0}$. Indeed, by Lemma 5, there exists another $\hat{\boldsymbol{s}}' \neq \hat{\boldsymbol{s}}$ (with overwhelming probability). Furthermore, because of Lemma 6, the signing protocol is witness-indistinguishable and therefore there is a probability of at least 1/2 that the signer's output corresponds to $\hat{\boldsymbol{s}}'$. As the signer possesses the secret key while the user does not, and because of Lemma 6, all protocol messages are distributed independently of the secret key, even if $\hat{z}^\dagger - \hat{z}^\ddagger + \hat{\boldsymbol{s}}'(\boldsymbol{\omega}^\dagger - \boldsymbol{\omega}^\ddagger) = \boldsymbol{0}$, $\mathscr{B}$ has at least 1/2 chance of claiming that $\hat{z}^\dagger - \hat{z}^\ddagger + \hat{\boldsymbol{s}}(\boldsymbol{\omega}^\dagger - \boldsymbol{\omega}^\ddagger) \neq \boldsymbol{0}$. Since $\hat{z}^\dagger - \hat{z}^\ddagger + \hat{\boldsymbol{s}}(\boldsymbol{\omega}^\dagger - \boldsymbol{\omega}^\ddagger) \neq \boldsymbol{0}$, we deduce that $\hat{z}^\dagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\dagger \neq \hat{z}^\ddagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\ddagger$. Furthermore, since $\| \hat{z}^\dagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\dagger \|_\infty$, $\| \hat{z}^\ddagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\ddagger \|_\infty \leq d_G + nd_s d_\omega < d_D$, we obtain $(\hat{z}^\dagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\dagger, \hat{z}^\ddagger + \hat{\boldsymbol{s}}\boldsymbol{\omega}^\ddagger)$ as a collision in $D \times D$, with probability

$$\theta_{\text{col}} \geq \frac{1}{4(k_{\text{info}} + 1)}\left(1 - \frac{1}{|D_\epsilon|}\right)\theta_{\text{frk}},$$

which is noticeable due to θ.

Restarts: Finally, we argue that the only way for a user to obtain a valid signature from an aborted interaction is if he can solve the collision problem for h in D. Indeed, for an abort to occur in step 5, the user needs to 'convince' the honest signer by sending him result $= (C, \boldsymbol{a}, \boldsymbol{a}', \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}', \epsilon)$, which together with his view of the interaction $(\boldsymbol{Y}_1, \boldsymbol{Y}, \epsilon^*, \hat{\boldsymbol{z}}^*, \hat{\boldsymbol{y}}_2, \boldsymbol{\gamma}, \boldsymbol{e})$, satisfy the abort criteria

$$\epsilon^* + \boldsymbol{a} + \boldsymbol{a}' (\bmod 2d_\epsilon + 1) = \epsilon \quad (8)$$

$$H(\boldsymbol{Y}_1 + \boldsymbol{S}\boldsymbol{a} + h(\hat{\boldsymbol{\beta}}), \boldsymbol{Y} + \boldsymbol{Z}\boldsymbol{a}' + h(\hat{\boldsymbol{\beta}}'), \boldsymbol{Z}, C) = \epsilon \quad (9)$$

$$\boldsymbol{e} + \boldsymbol{a} + \boldsymbol{\gamma} + \boldsymbol{a}' (\bmod 2d_\epsilon + 1) = H(h(\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}}) + \boldsymbol{S}(\boldsymbol{e} + \boldsymbol{a}), h(\hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}') + \boldsymbol{Z}(\boldsymbol{\gamma} + \boldsymbol{a}'), \boldsymbol{Z}, C) \quad (10)$$

$$\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}} \notin G^m \vee \boldsymbol{e} + \boldsymbol{a} \notin G_\omega \vee \hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}' \notin G_\sigma^m \vee \boldsymbol{\gamma} + \boldsymbol{a}' \notin G_\delta \quad (11)$$

Suppose that the malicious user successfully obtains a forged signature $(r_0, \hat{\boldsymbol{z}}_0, \boldsymbol{\omega}_0, \hat{\boldsymbol{\sigma}}_0, \boldsymbol{\delta}_0)$ from an aborted interaction. Thus, we may assume that $(r_0, \hat{\boldsymbol{z}}_0, \boldsymbol{\omega}_0, \hat{\boldsymbol{\sigma}}_0, \boldsymbol{\delta}_0)$ satisfies all of the verification criteria from Section 4.1. First, observe that the adversarial user may succeed by hiding $\epsilon' \neq \epsilon$ in the computation of ϵ^*. However, to achieve this he would need to predict the output of H, which happens with a negligible probability of $1/|D_\epsilon|$. Thus, we have $\epsilon = \epsilon'$ with an overwhelming probability of $1 - (1/|D_\epsilon|)$. As $\epsilon = \omega_0 + \delta_0 (\bmod 2d_\epsilon + 1) = \omega + \delta (\bmod 2d_\epsilon + 1)$, it follows from (10) that $h(\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}}) + \boldsymbol{S}(\boldsymbol{e} + \boldsymbol{a}) = h(\hat{\boldsymbol{z}}_0 + \boldsymbol{\omega}_0 \hat{\boldsymbol{s}})$. Equivalently, this can be written as

$$h(\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}} + \hat{\boldsymbol{s}}(\boldsymbol{e} + \boldsymbol{a})) = h(\hat{\boldsymbol{z}}_0 + \omega_0 \hat{\boldsymbol{s}}). \quad (12)$$

Next, notice that with an overwhelming probability of at least $1 - (1/|D_\epsilon|)$, we have $\boldsymbol{\omega}_0 = \boldsymbol{e} + \boldsymbol{a}$ (unless $\boldsymbol{e} + \boldsymbol{a} \notin G_\omega$, in which case we have a contradiction because we know that $\boldsymbol{\omega}_0 \in G_\omega$). Indeed, the only way for the malicious user to obtain a $\boldsymbol{\omega}_0 \neq \boldsymbol{e} + \boldsymbol{a}$, is if during step 2 he used an $\boldsymbol{a}_0 = \boldsymbol{\omega}_0 - \boldsymbol{\omega} + \boldsymbol{a}$, which implies that he would have to successfully guess $\boldsymbol{\omega}$, which he can do only with a negligible probability of

$$\frac{1}{|G_\omega|} \leq \frac{1}{|D_\epsilon|} = \frac{1}{(2d_\epsilon + 1)^n}.$$

From Bayes' rule, we can determine that the probability that $\boldsymbol{e} + \boldsymbol{a} \in G_\omega$, given that (11) holds is $(e^{-1/\phi} - e^{-4/\phi})/(1 - e^{-4/\phi})$, a constant. Similarly, with an overwhelming probability of at least $1 - (1/|D_\epsilon|)$, we have $\boldsymbol{\delta}_0 = \boldsymbol{\gamma} + \boldsymbol{a}'$ (unless $\boldsymbol{\gamma} + \boldsymbol{a}' \notin G_\delta$, in which case we have a contradiction because $\boldsymbol{\delta}_0 \in G_\delta$). Finally, with an overwhelming probability of at least $1 - (1/|D_\epsilon|)$, we have $\hat{\boldsymbol{\sigma}}_0 = \hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}'$ (unless $\hat{\boldsymbol{y}}_2 + \hat{\boldsymbol{\beta}}' \notin G_\sigma^m$, in which case we have a contradiction because we know that $\hat{\sigma}_0 \in G_\sigma^m$). Thus, the only possible case for condition (11) to hold, is if $\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}} \notin G^m$. Observe that in that case, the arguments of h in (12) cannot be equal because then $\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}} = \hat{\boldsymbol{z}}_0$, which contradicts the hypothesis that $\hat{\boldsymbol{z}}_0 \in G^m$. Therefore, we have $\hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}} \neq \hat{\boldsymbol{z}}_0$, and since $\| \hat{\boldsymbol{z}}_0 + \boldsymbol{\omega}_0 \hat{\boldsymbol{s}} \|_\infty < d_D$ and $\| \hat{\boldsymbol{z}}^* + \hat{\boldsymbol{\beta}} + \hat{\boldsymbol{s}}(\boldsymbol{e} + \boldsymbol{a}) \|_\infty \leq d_{G_*} + d_\beta + nd_s d_\omega = d_D$, we have a collision in D. Thus, by applying the law of total probability, we can deduce that the overall probability of obtaining valid signatures out of aborted interactions is

$$\theta_{\text{abort}} \geq \left(1 - \frac{1}{|D_\epsilon|}\right)^4 \left(\frac{e^{-1/\phi} - e^{-4/\phi}}{1 - e^{-4/\phi}}\right)^3 \theta,$$

which is noticeable if θ is noticeable. In conclusion, if $b = 1$, $\mathscr{A}$'s overall success probability is $\theta_{\text{overall}} \geq \min(\theta_{\text{col}}, \theta_{\text{abort}})$, which is noticeable if θ is noticeable. □

By combining Theorem 4 with Theorem 1, we obtain the following.

Corollary 1: **:** The proposed PBSS is unforgeable if solving $\text{Ring} - \text{SVP}_\gamma^\infty$ is hard in the worst case, for approximation factors $\gamma \geq 16 d_D mn \log^2(n) = \tilde{\mathcal{O}}(n^4)$, in ideal lattices of $\boldsymbol{R}$.

Remark 4: As a consequence of Theorem 4, if we require that $q_{\text{sig}} = o(n)$, our construction benefits from the subexponential hardness of ideal lattice problems.

Remark 5: As $\boldsymbol{e}$ is reduced modulo $2d_e + 1$ in step 3 of our signing protocol, we have a milder worst-case hardness assumption of $\tilde{\mathcal{O}}(n^4)$, compared with the BSS from [16], which is based on the worst-case hardness of Ring-SVP for approximation factors in $\tilde{\mathcal{O}}(n^5)$. We believe that this 'trick' could also be used on [16] to improve the hardness assumption therein.

4.3.4 Leakage resilience: In proving leakage-resilience for our scheme, we rely on the core observation of [51], which states that any collision-resistant hash function (our underlying hash-function is proven collision-resistant in [36]) is a leakage-resilient one-way hash function when certain conditions are imposed on the leakage oracle (these conditions are necessary because the recent work of [56] shows that for some leakage scenarios, leakage-resilience is impossible to achieve). This observation is also used by other works to construct leakage-resilient primitives [16, 51].

In the next theorem, we establish leakage resilience for our construction. In proving leakage resilience, we will show that the secret key's conditional min-entropy: $H_\infty(sk|\text{Leak}(sk)) = \min_{sk'}\{-\log(Prob[sk = sk'|\text{Leak}(sk)])\}$ is large enough for the scheme to be secure. The proof closely follows the corresponding proof of [16], with the additional observation that $\boldsymbol{Z} = \mathcal{F}(\text{info})$ is not related to the signer's secret key, and thus does not leak information about $\hat{\boldsymbol{s}}$ (the proof is included in the Appendix).

Theorem 5: (Leakage resilience): Let $c_m = \omega(1)$ and let $L := \log(|D_s|^m) = mn\log(2d_s + 1)$ denote the length of the signer's secret key in the proposed PBSS. Given $\boldsymbol{S} = h(\hat{\boldsymbol{s}})$ and a total secret-key leakage $f(\hat{\boldsymbol{s}})$ of $\lambda = \delta L = (1 - o(1))L$ bits, the conditional min-entropy H_∞ of $\hat{\boldsymbol{s}}$, is positive with overwhelming probability.

Remark 6: From Theorem 6, we see that if we additionally require that $c_m = \omega(1)$ (e.g. by choosing $c_m = \log(n)$) for $m := \lfloor c_m \log(q) \rfloor + 1$, then our PBSS retains its quasi-optimal performance and is also leakage-resilient.

5 Extensions

In this section, we discuss several extensions of the classic security model of PBSS that are applicable to our construction. We consider honest-user unforgeability, selective-failure blindness, and dishonest-key blindness. To the best of our knowledge, none of these properties have previously been examined in the context of PBSS.

5.1 Honest-user unforgeability

In [33], the authors propose a strengthened notion of one-more unforgeability for blind signatures, called *unforgeability in the presence of honest users* (or *honest-user unforgeability*, for short). The idea is that an adversary could exploit the presence of an honest user, and use him as an intermediary to indirectly obtain signatures from the signer (it is not difficult to see that the absence of such honest users leads to the classic notion of unforgeability of BSS [38, 50]. However, unforgeability is shown to be weaker than honest-user unforgeability [33]). That way, the adversary may be able to produce more signatures than the number of times he directly interacted with the signer. These kinds of attacks are not captured by the notion of unforgeability for regular blind signatures.

Honest-user unforgeability however is given with regular BSS in mind. Here, we adapt it for PBSS, thus obtaining an even stronger notion of unforgeability for PBSS. We also show that the transformation given in [33] is still relevant when it comes to PBSS, a result which we believe may be of interest in its own right. Before giving the new definition, we must fix some notation. Let $\mathcal{P}(sk, pk, ., .)$ be an oracle that on input μ (a message) and common information *info*, executes the signature issuing protocol $\langle \mathcal{S}, \mathcal{U} \rangle$, thus obtaining a signature σ. Let *trans* denote the transcript comprised of all messages exchanged between the parties in such an interaction. When the protocol terminates, $\mathcal{P}$ returns (σ, trans). The execution of $\langle \mathcal{S}(sk, \text{info}), \mathcal{U}(pk, \mu, \text{info}) \rangle$ by $\mathcal{P}$ is considered to be atomic, i.e. during a call to $\mathcal{P}$, no other interactions occur. If the interaction aborts, $\mathcal{P}$ returns $(\perp, \text{trans})$, where *trans* is the transcript up to that point of execution.

Definition 8: (Honest-user unforgeability of PBSS): An interactive partially blind signature scheme $\text{PBSS} = (KG, \langle \mathcal{S}, \mathcal{U} \rangle, Vf)$ is honest-user unforgeable if Vf is deterministic, and for any efficient algorithm $\mathcal{U}^*$, the probability that experiment $\text{Exp}^{hu-\text{omf}}_{\mathcal{U}^*,\text{PBSS}}(n)$ evaluates to 1 is negligible (as a function of n), where

Experiment $\text{Exp}^{hu-\text{omf}}_{\mathcal{U}^*,\text{PBSS}}(n)$

$(pk, sk) \leftarrow_\$ \text{PBSS}.\text{KG}(1^n)$

For each info, let k_{info} denote the number of successful, complete, direct interactions with the signer $\mathcal{S}$:

$((\mu_1^*, \text{info}, \sigma_1^*), \ldots, (\mu_{k_{\text{info}}+1}^*, \text{info}, \sigma_{k_{\text{info}}+1}^*)) \leftarrow_\$ \mathcal{U}^{*\langle \mathcal{S}(sk), . \rangle^\infty, \mathcal{P}(sk, pk, ., .)}(pk)$

andlet $\mu_1, \ldots, \mu_{n_{\text{info}}}$ be the messages pertaining to info that were queriedto $\mathcal{P}(sk, pk, ., .)$.

Return1iff

1. $\mu_i^* \neq \mu_j,\ \forall i = 1, \ldots, k_{\text{info}},$ and $\forall j = 1, \ldots, n_{\text{info}},$
2. $\mu_i^* \neq \mu_j^*,\ \forall i, j = 1, \ldots, k_{\text{info}} + 1$ with $i \neq j$, and
3. $\text{PBS}.\text{Vf}(pk, \mu_i^*, \text{info}, \sigma_i^*) = 1, \forall i = 1, \ldots, k_{\text{info}} + 1.$

Note that when counting the interactions in which $\mathcal{S}$ returns 'ok', we do not count the interactions simulated by $\mathcal{P}$.

We now present a way to turn any unforgeable PBSS into an honest-user unforgeable PBSS, that is analogous to the one from [33] (for brevity, we include the proof in the Supplementary Material). This transformation comes at the expense of a negligible overhead compared to the original PBSS.

Construction 1: Let $\text{PBSS}' = (\text{KG}', \langle \mathcal{S}', \mathcal{U}' \rangle, Vf')$ be an interactive PBSS. We define a new partially blind signature scheme $\text{PBSS} = (\text{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \text{Vf})$ through the following algorithms:

- *Key generation*: Algorithm $\text{KG}(1^n)$ runs $(sk', pk') \leftarrow \text{KG}'(1^n)$ and returns the key pair.
- *Signature issuing protocol*: Signer $\mathcal{S}$ is identical to the original signer $\mathcal{S}'$. User $\mathcal{U}(pk, \mu, \text{info})$ chooses $r \leftarrow_\$ \{0,1\}^n$, sets $\mu' \leftarrow \mu \parallel r$, and then invokes the original user $\mathcal{U}'(pk, \mu', \text{info})$, who then interacts with $\mathcal{S}'(sk, \text{info})$. When $\mathcal{U}'$ outputs a signature σ, $\mathcal{U}$ computes $\sigma' \leftarrow (\sigma, r)$ and returns σ'.
- *Signature verification*: Algorithm $Vf(pk, \mu, \text{info}, \sigma')$ parses σ' as (σ, r) and returns the result of $Vf'(pk, \mu \parallel r, \text{info}, \sigma)$.

Theorem 6: : If complete, partially blind, and unforgeable PBSS exist, then there exist PBSS which are complete, partially blind, unforgeable, and also honest-user unforgeable.

Remark 7: Our scheme can easily be modified to use this transformation by having the user commit to $\mu \parallel r'$ for some $r' \leftarrow_\$ \{0,1\}^n$ instead of μ during step 2. If any restarts occur during step 2, r' needs to be resampled as well. Finally, r' will be included in the final signature and the verification condition becomes: $\boldsymbol{\omega} + \boldsymbol{\delta}(\bmod 2d_e + 1) = H(h(\hat{\boldsymbol{z}}) + \boldsymbol{\omega}\boldsymbol{S}, h(\hat{\boldsymbol{\sigma}}) + \boldsymbol{\delta}\mathcal{F}(\text{info}), \mathcal{F}(\text{info}), \text{com}(\mu \parallel r', r))$.

5.1.1 Dishonest-key blindness: In the definition of (partial) blindness, we implicitly assumed that the signer generates his secret and public keys through the scheme's key generation algorithm. Abdalla *et al.* [57] proposed an augmented notion of

blindness that allows the signer to select *pk* on his own. This notion can also be transferred to PBSS. Our scheme's partial blindness proof does not rely on any specific properties of the secret key and thus satisfies this strengthened notion of partial-blindness as well.

5.2 Selective-failure blindness

The notion of blindness does not cover cases in which the protocol has to be aborted prematurely. However, we would like to ensure that blindness also holds in cases where the signer is able to cause one of the protocol executions to abort by choosing one of the messages μ_0 or μ_1 from some secret distribution. For that purpose, Camenisch *et al.* [58] introduced the stronger notion of *selective-failure blindness*, and Fischlin and Schröder [43] later expanded upon that work by providing a generic transformation for turning any BSS into a selective-failure BSS, at the expense of only a negligible computational overhead. This notion can easily be adapted for PBSS because *info* is a common input to both user instances in the partial blindness experiment. Our scheme is selective-failure blind because it makes use of the transformation of [43]. Indeed, the signer's view is limited to commitments on the messages he signs, and uncovering them would require him to break *com*'s hiding property.

6 Conclusions

In this work, we presented the first leakage-resilient, lattice-based PBSS in the literature. Our construction has the same four-move structure and uses a commitment scheme like the scheme from [16]. Its performance is quasi-optimal and its security is proven in the ROM under milder worst-case ideal lattice assumptions compared to [16]. Besides being quantum-resistant, our construction is also honest-user unforgeable, selective-failure blind, dishonest-key blind, and can withstand sub-exponential-time attacks, and limited side-channel attacks against the signer's secret key thanks to its leakage resilience.

7 Acknowledgments

The authors thank the anonymous reviewers for their helpful comments in improving this work. Foteini Baldimtsi has received funding from NSF with award number 1717067.

8 References

[1] Chaum, D.: 'Blind signatures for untraceable payments', in Chaum, D., Rivest, R.L., Sherman, A.T. (Eds): '*Advances in cryptology*' (Springer US, Boston, MA, 1983), pp. 199–203

[2] Von-Solms, S., Naccache, D.: 'On blind signatures and perfect crimes', *Comput. Secur.*, 1992, **11**, (6), pp. 581–583

[3] Abe, M., Fujisaki, E.: 'How to date blind signatures'. Proc. of the Int. Conf. on the Theory and Applications of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT), Kyongju, Korea, 1996, pp. 244–251

[4] Stadler, M., Piveteau, J.M., Camenisch, J.: 'Fair blind signatures'. Advances in Cryptology – (EUROCRYPT '95), Saint-Malo, France, 1995, pp. 209–219

[5] Fuchsbauer, G., Vergnaud, D.: 'Fair blind signatures without random oracles'. Proc. of the 3rd Int. Conf. on Cryptology, Africa, 2010, pp. 16–33

[6] Abe, M., Okamoto, T.: 'Provably secure partially blind signatures'. Proc. of the 20th Annual Int. Cryptology Conf. on Advances in Cryptology, Santa Barbara, California, USA, 2000, pp. 271–286

[7] Cramer, R., Damgård, I., Schoenmakers, B.: 'Proofs of partial knowledge and simplified design of witness hiding protocols'. Proc. of the 14th Annual Int. Cryptology Conf. on Advances in Cryptology, Santa Barbara, California, USA, 1994, pp. 174–187

[8] Rückert, M., Schröder, D.: 'Fair partially blind signatures'. Proc. of the 3rd Int. Conf. on Cryptology, Africa, 2010, pp. 34–51

[9] Shor, P.W.: 'Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer', *SIAM J. Comput.*, 1997, **26**, (5), pp. 1484–1509

[10] Ajtai, M.: 'Generating hard instances of lattice problems (extended abstract)'. Proc. of the 28th Annual ACM Symp. on Theory of Computing (STOC '96), Philadelphia, Pennsylvania, USA, 1996, pp. 99–108

[11] Peikert, C.: 'A decade of lattice cryptography', *Found. Trends Theor. Comput. Sci.*, 2016, **10**, (4), pp. 283–424

[12] Tian, H., Zhang, F., Wei, B.: 'A lattice-based partially blind signature', *Secur. Commun. Netw.*, 2016, **9**, (12), pp. 1820–1828

[13] Groot-Bruinderink, L., Hülsing, A., Lange, T., *et al.*: 'Flush, Gauss, and reload – a cache attack on the bliss lattice-based signature scheme', in Gierlichs, B., Poschmann, A.Y. (Eds): '*IACR-CHES*' (Springer-Verlag, Berlin, Heidelberg, 2016), pp. 323–345

[14] Pessl, P., Bruinderink, L.G., Yarom, Y.: 'To bliss-b or not to be: attacking strongswan's implementation of post-quantum signatures'. Proc. of the 2017 ACM SIGSAC Conf. on Computer and Communications Security (CCS '17), Dallas, Texas, USA, 2017, pp. 1843–1855

[15] Espitau, T., Fouque, P.A., Gérard, B., *et al.*: 'Side-channel attacks on bliss lattice-based signatures: exploiting branch tracing against strongswan and electromagnetic emanations in microcontrollers'. Proc. of the 2017 ACM SIGSAC Conf. on Computer and Communications Security (CCS '17), Dallas, Texas, USA, 2017, pp. 1857–1874

[16] Rückert, M.: 'Lattice-based blind signatures'. Advances in Cryptology (ASIACRYPT 2010), Singapore, 2010, pp. 413–430

[17] Abe, M.: 'A secure three-move blind signature scheme for polynomially many signatures'. Proc. of the Int. Conf. on the Theory and Application of Cryptographic Techniques: advances in Cryptology. (EUROCRYPT '01), Innsbruck (Tyrol), Austria, 2001, pp. 136–151

[18] Chow, S.S.M., Hui, L.C.K., Yiu, S.M., *et al.*: 'Two improved partially blind signature schemes from bilinear pairings'. Proc. of the 10th Australasian Conf. on Information Security and Privacy, Brisbane, Australia, 2005, pp. 316–328

[19] Okamoto, T.: 'Efficient blind and partially blind signatures without random oracles'. Proc. of the 3rd Conf. on Theory of Cryptography, New York, NY, 2006, pp. 80–99

[20] Li, F., Zhang, M., Takagi, T.: 'Identity-based partially blind signature in the standard model for electronic cash', *Math. Comput. Model.*, 2013, **58**, (1), pp. 196–203. Financial IT & Security and 2010 International Symposium on Computational Electronics

[21] Gentry, C., Peikert, C., Vaikuntanathan, V.: 'Trapdoors for hard lattices and new cryptographic constructions'. Proc. of the 40th Annual ACM Symp. on Theory of Computing (STOC '08), Victoria, British Columbia, Canada, 2008, pp. 197–206

[22] Lyubashevsky, V.: 'Lattice-based identification schemes secure under active attacks'. 11th Int. Conf. on Public Key Cryptography Proc. of the Practice and Theory in Public Key Cryptography (PKC '08), Barcelona, Spain, 2008, pp. 162–179

[23] Lyubashevsky, V.: 'Fiat-Shamir with aborts: applications to lattice and factoring-based signatures'. Proc. of the 15th Int. Conf. on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, Tokyo, Japan, 2009, pp. 598–616

[24] Lyubashevsky, V.: 'Lattice signatures without trapdoors'. Proc. of the 31st Annual Int. Conf. on Theory and Applications of Cryptographic Techniques, Cambridge, UK, 2012, pp. 738–755

[25] Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: 'Practical lattice-based cryptography: a signature scheme for embedded systems'. Cryptographic Hardware and Embedded Systems (CHES 2012), Leuven, Belgium, 2012, pp. 530–547

[26] Ducas, L., Durmus, A., Lepoint, T., *et al.*: 'Lattice signatures and bimodal gaussians'. Advances in Cryptology (CRYPTO 2013), Santa Barbara, California, USA, 2013, pp. 40–56

[27] Alkim, E., Bindel, N., Buchmann, J.A., *et al.*: 'Tesla: tightly-secure efficient signatures from standard lattices'. Cryptology ePrint Archive, Report 2015/755, 2015. Available at https://eprint.iacr.org/2015/755

[28] Lyubashevsky, V.: 'Digital signatures based on the hardness of ideal lattice problems in all rings'. Proc. Part II, of the 22nd Int. Conf. on Advances in Cryptology (ASIACRYPT 2016), Hanoi, Vietnam, vol. 10032, 2016, pp. 196–214

[29] Ducas, L., Kiltz, E., Lepoint, T., *et al.*: 'Crystals-dilithium: a lattice-based digital signature scheme', *IACR Trans. Cryptographic Hardware Embedded Syst.*, 2018, **2018**, (1), pp. 238–268

[30] Kiltz, E., Lyubashevsky, V., Schaffner, C.: 'A concrete treatment of fiat-shamir signatures in the quantum random-oracle model'. Advances in Cryptology (EUROCRYPT 2018), Tel Aviv, Israel, 2018, pp. 552–586

[31] Zhu, H., Tan, Y.A., Zhang, X., *et al.*: 'A round-optimal lattice-based blind signature scheme for cloud services', *Future Gener. Comput. Syst*, 2017, **73**, (C), pp. 106–114

[32] Cheon, J.H., Jeong, J., Shin, J.S.: 'Cryptoanalysis on á round-optimal lattice-based blind signature scheme for cloud services', *Fut Gener. Comput. Syst.*, 2019, **95**, pp. 100–103

[33] Schröder, D., Unruh, D.: 'Security of blind signatures revisited'. Public Key Cryptography (PKC 2012), Darmstadt, Germany, 2012, pp. 662–679

[34] Rückert, M.: '*Lattice-based signature schemes with additional features*' (Technische Universität, Darmstadt, 2011). Available at http://tuprints.ulb.tu-darmstadt.de/2393/

[35] Bellare, M., Rogaway, P.: 'Random oracles are practical: A paradigm for designing efficient protocols'. Proc. of the 1st ACM Conf. on Computer and Communications Security (CCS '93), Fairfax, Virginia, USA, 1993, pp. 62–73

[36] Lyubashevsky, V., Micciancio, D.: 'Generalized compact knapsacks are collision resistant'. Proc. of the 33rd Int. Conf. on Automata, Languages and Programming - Volume Part II, Venice, Italy, 2006, pp. 144–155

[37] Fischlin, M., Schröder, D.: 'On the impossibility of three-move blind signature schemes'. Proc. of the 29th Annual Int. Conf. on Theory and Applications of Cryptographic Techniques (EUROCRYPT '10), Monaco, French Riviera, 2010, pp. 197–215

[38] Pointcheval, D., Stern, J.: 'Security arguments for digital signatures and blind signatures', *J. Cryptol.*, 2000, **13**, (3), pp. 361–396

[39] Bresson, E., Monnerat, J., Vergnaud, D.: 'Separation results on the 'one-more' computational problems'. Topics in Cryptology (CT-RSA 2008), San Francisco, USA, 2008, pp. 71–87

[40] Lindell, Y.: 'Bounded-concurrent secure two-party computation without setup assumptions'. Proc. of the 35th Annual ACM Symp. on Theory of Computing (STOC '03), San Diego, USA, 2003, pp. 683–692

[41] Brown, D.R.L.: 'Irreducibility to the one-more evaluation problems: more may be less'. 2007, Cryptology ePrint Archive, Report 2007/435. Available at https://eprint.iacr.org/2007/435
[42] Baldimtsi, F., Lysyanskaya, A.: 'On the security of one-witness blind signature schemes'. Advances in Cryptology (ASIACRYPT 2013), Bengaluru, India, 2013, pp. 82–99
[43] Fischlin, M., Schröder, D.: 'Security of blind signatures under aborts'. Proc. of the 12th Int. Conf. on Practice and Theory in Public Key Cryptography (PKC '09), Irvine, CA, USA, 2009, pp. 297–316
[44] Cormen, T.H., Leiserson, C.E., Rivest, R.L., *et al.*: '*Introduction to algorithms*' (The MIT Press, UK, 2009, 3rd edn.)
[45] Micciancio, D., Regev, O.: 'Worst-case to average-case reductions based on Gaussian measures', *SIAM J. Comput.*, 2007, **37**, (1), pp. 267–302
[46] Petrov, V.: '*Sums of independent random variables*' (Springer-Verlag, Berlin, Heidelberg, 1975, 1st edn.)
[47] Damgård, I.: 'Commitment schemes and zero-knowledge protocols', in Damgård, I.B. (Ed.): '*Lectures on data security, modern cryptology in theory and practice, summer school, aarhus, Denmark, July 1998*' (Springer- Verlag, Berlin, Heidelberg, 1999), pp. 63–86
[48] Arbitman, Y., Dogon, G., Lyubashevsky, V., *et al.*: 'Swifftx: a proposal for the sha-3 standard', 2008. Available at: https://www.eecs.harvard.edu/~alon/PAPERS/lattices/swifftx.pdf
[49] Micciancio, D., Regev, O.: 'Lattice-based cryptography', in Bernstein, D.J., Buchmann, J., Dahmen, E. (Eds): '*Post-quantum cryptography*' (Springer, Berlin, Heidelberg, 2009), pp. 147–191
[50] Juels, A., Luby, M., Ostrovsky, R.: 'Security of blind digital signatures'. Advances in Cryptology (CRYPTO '97), Santa Barbara, California, USA, 1997, pp. 150–164
[51] Katz, J., Vaikuntanathan, V.: 'Signature schemes with bounded leakage resilience'. Advances in Cryptology (ASIACRYPT 2009), Tokyo, Japan, 2009, pp. 703–720
[52] Rückert, M., Schneider, M.: 'Estimating the security of lattice-based cryptosystems'. 2010, Cryptology ePrint Archive, Report 2010/137, eprint.iacr.org/2010/137
[53] Gama, N., Nguyen, P.Q.: 'Predicting lattice reduction'. Proc. of the Theory and Applications of Cryptographic Techniques 27th Annual Int. Conf. on Advances in Cryptology, Istanbul, Turkey, 2008, pp. 31–51
[54] Fiat, A., Shamir, A.: 'How to prove yourself: practical solutions to identification and signature problems'. Proc. on Advances in cryptology (CRYPTO '86), Santa Barbara, California, USA, 1987, pp. 186–194
[55] Bellare, M., Neven, G.: 'Multi-signatures in the plain public-key model and a general forking lemma'. Proc. of the 13th ACM Conf. on Computer and Communications Security (CCS '06), Alexandria, Virginia, USA, 2006, pp. 390–399
[56] Komargodski, I.: 'Leakage resilient one-way functions: the auxiliary-input setting'. Proc. Part I of the 14th Int. Conf. on Theory of Cryptography, Beijing, China, vol. 9985, 2016, pp. 139–158
[57] Abdalla, M., Namprempre, C., Neven, G.: 'On the (im)possibility of blind message authentication codes'. Proc. of the 2006 The Cryptographers Track at the RSA Conf. on Topics in Cryptology, San Jose, CA, USA, 2006, pp. 262–279
[58] Camenisch, J., Neven, G., Shelat, A.: 'Simulatable adaptive oblivious transfer'. Proc. of the 26th Annual Int. Conf. on Advances in Cryptology, Barcelona, Spain, 2007, pp. 573–590

9 Appendix

9.1 Forking Lemma

The generalised forking lemma from [55] is a probabilistic tool for proving security of cryptographic constructions in the ROM. Informally, it states that if an algorithm $\mathscr{A}$ outputs a pair of values (I, σ) with $I > 0$ with noticeable probability *acc*, then the forking algorithm $F_{\mathscr{A}}$ defined below will with noticeable probability return $(1, \sigma, \sigma')$ based on two executions of $\mathscr{A}$, sharing an identical prefix up to the Ith query to H. In other words, the probability of getting two related runs with the same value of I, and a common prefix of length $I - 1$ is not too small.

Lemma 7: Fix an integer $q \geq 1$ and a set H of size $h \geq 2$. Let $\mathscr{A}$ be a randomised algorithm that on input $x, h_1, \ldots, h_q$ returns a pair, the first element of which is an integer in the range $0, \ldots, q$ and the second element of which we refer to as a side output. Let IG be a randomised algorithm that we call the input generator. The accepting probability of $\mathscr{A}$, denoted as acc, is defined as the probability that $J \geq 1$ in the experiment $x \leftarrow_{\$} IG; h_1, \ldots, h_q \leftarrow_{\$} H; (J, \sigma) \leftarrow_{\$} \mathscr{A}(x; h_1, \ldots, h_q)$. The forking algorithm $F_{\mathscr{A}}$ associated to $\mathscr{A}$ is the randomised algorithm that takes input x and proceeds as follows:

Algorithm $F_{\mathscr{A}}(x)$

Pick coins ρ for $\mathscr{A}$ at random

$h_1, \ldots, h_q \leftarrow_{\$} H$

$(I, \sigma) \leftarrow \mathscr{A}(x; h_1, \ldots, h_q; \rho)$

If $I = 0$ then return $(0, \epsilon, \epsilon)$

$h'_I, \ldots, h'_q \leftarrow_{\$} H$

$(I', \sigma') \leftarrow \mathscr{A}(x; h_1, \ldots, h_{I-1}, h'_I, \ldots, h'_q; \rho)$

If $I = I'$ and $h_I \neq h'_I$, then return $(1, \sigma, \sigma')$

Elsereturn $(0, \epsilon, \epsilon)$.

Let $frk = \text{Prob}[b = 1 : x \leftarrow_{\$} IG; (b, \sigma, \sigma') \leftarrow F_{\mathscr{A}}(x)]$.

Then $frk \geq \text{acc}\left(\frac{\text{acc}}{q} - \frac{1}{h}\right)$.

9.2 Proofs of results from sections 4 and 5

9.2.1 Proof of Lemma 6: Initially, observe that θ_{frk} and $\boldsymbol{Y}'_1$ do not depend on the choice of secret key. The same holds for $\boldsymbol{Y}$ and $\boldsymbol{Y}'$. Furthermore, ϵ^* and $\epsilon^{*'}$ are independent of any particular $\hat{\boldsymbol{y}}_1 \in h^{-1}(\boldsymbol{Y}_1) \cap D_y^m$ because $\boldsymbol{Y}_1$ statistically hides $\hat{\boldsymbol{y}}_1$ through h. Moreover, $\hat{\boldsymbol{y}}_2$ and $\hat{\boldsymbol{y}}'_2$, as well as $\boldsymbol{\gamma}$ and $\boldsymbol{\gamma}'$ are all sampled independently of the secret key. Finally, we have to show that $\hat{\boldsymbol{z}}^*$ and $\hat{\boldsymbol{z}}^{*'}$ are also distributed independently of the secret key. For that, let $\boldsymbol{e}$ be any factor used by the signer during step 3 of our protocol, to compute $\hat{\boldsymbol{z}}^*$, i.e.: $\hat{\boldsymbol{z}}^* = \hat{\boldsymbol{y}}_1 - \boldsymbol{e}\hat{\boldsymbol{s}} \in G_*^m$. Next, we set $\hat{\boldsymbol{y}}'_1 \leftarrow \hat{\boldsymbol{y}}_1 - \hat{\boldsymbol{s}}\boldsymbol{e} + \hat{\boldsymbol{s}}'e$, which implies that $\hat{\boldsymbol{z}}^* = \hat{\boldsymbol{y}}'_1 - \hat{\boldsymbol{s}}'e$. We then easily see that $\hat{\boldsymbol{y}}'_1 \in h^{-1}(\boldsymbol{Y}_1) \cap D_y^m$. Indeed, $\hat{\boldsymbol{y}}'_1 \in h^{-1}(\boldsymbol{Y}_1)$ because $h(\hat{\boldsymbol{y}}'_1) = h(\hat{\boldsymbol{y}}_1 - \hat{\boldsymbol{s}}\boldsymbol{e} + \hat{\boldsymbol{s}}'e) = \boldsymbol{Y}_1 - \boldsymbol{e}\boldsymbol{S} + \boldsymbol{e}\boldsymbol{S} = \boldsymbol{Y}_1$. Additionally, $\hat{\boldsymbol{y}}_1 \in D_y^m$ since:

$$\| \hat{\boldsymbol{y}}'_1 \|_\infty = \| \hat{\boldsymbol{z}}^* + \hat{\boldsymbol{s}}'e \|_\infty \leq \| \hat{\boldsymbol{z}}^* \|_\infty + \| \hat{\boldsymbol{s}}'e \|_\infty \leq d_y - nd_sd_e + nd_sd_e = d_y, \quad (13)$$

where the last inequality follows from Lemma 3. In conclusion, no malicious user can distinguish whether the honest signer is using secret key $\hat{\boldsymbol{s}}$ with a masking term $\hat{\boldsymbol{y}}_1$ or $\hat{\boldsymbol{s}}'$ with a masking term $\hat{\boldsymbol{y}}'_1$, both of which yield the same output.

9.2.2 Proof of Theorem 5: We follow the same conservative approach as in [16] and treat the public key $\boldsymbol{S}$ as additional leakage. Notice that $\boldsymbol{Z} = \mathscr{F}(\text{info})$ is not related to the signer's secret key, and thus we do not treat it as a source of additional leakage for $\hat{\boldsymbol{s}}$. Define the function $g(\hat{\boldsymbol{s}}) := f(\hat{\boldsymbol{s}}) \parallel \boldsymbol{S}$ with a total tolerated leakage of at most $\lambda' = \lambda + n\log(q)$ bits. Next, apply Lemma 1 from [51] to g, λ', and $H' = 1$, with $\hat{\boldsymbol{s}}$ being the random variable. As $H = L = mn\log(2d_s + 1)$, we have

$$\text{Prob}[g(\hat{\boldsymbol{s}}) \in Y] \geq 1 - 2^{\lambda' - H + H'} = 1 - 2^{\lambda + n\log(q) - L + 1}, \quad (14)$$

which we want to be $\geq 1 - 2^{-p(n)}$. For any function $p(n)$ such that $\omega(\log(n)) \leq p(n) \leq \mathcal{O}(n\log(n))$, we bound the relative leakage from above by

$$\delta \leq 1 - \frac{p(n) + n\log(q) + 1}{L} = 1 - \frac{\Theta(n\log(n))}{c_m\Theta(n\log(n))} = 1 - \frac{1}{\omega(1)} = 1 - \frac{1}{o(1)}.$$

As a result, (14) becomes

$$\text{Prob}[g(\hat{\boldsymbol{s}}) \in Y] \geq 1 - 2^{\left(1 - \frac{p(n) + n\log(q) + 1}{L}\right)L + n\log(q) - L + 1} = 1 - 2^{p(n)}.$$

Thus, $\delta L = (1 - o(1))L$ leakage bits yield a non-zero conditional min-entropy with overwhelming probability $1 - 2^{-p(n)} \geq 1 - 2^{-\omega(\log(n))}$.

9.2.3 Proof of Theorem 6: It is trivial to see that if PBSS′ is complete and partially blind, then so is PBSS. Thus, we only need to show that PBSS is honest-user unforgeable, if PBSS′ is unforgeable. We will prove this by contradiction. Assume that PBSS′ is unforgeable but PBSS is not honest-user unforgeable. Thus, as per Definition 8, there exists an efficient adversary $\mathcal{U}^*$ that wins at experiment $\mathrm{Exp}^{\mathrm{hu-omf}}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$ with noticeable probability. We will construct an attacker $\mathcal{B}$ that breaks the unforgeability of PBSS′:

Setup: Algorithm $\mathcal{B}$ receives a public key *pk* as input and runs $\mathcal{U}^*$ in a black-box manner, simulating the oracles as follows:

Direct signing queries: If $\mathcal{U}^*$ directly invokes the signing oracle $\mathcal{S}'$, $\mathcal{B}$ simply relays all messages exchanged between the malicious user and the signer.

Indirect signing queries: If $\mathcal{U}^*$ indirectly invokes $\mathcal{S}'$ through oracle $\mathcal{P}$ on message, $\mu \in \{0,1\}^*$, and common information info $\in \{0,1\}^*$, then $\mathcal{B}$ chooses a random $r \leftarrow_{\$} \{0,1\}^n$, sets $\mu' \leftarrow \mu \parallel r$, and engages in an interactive PBSS with the signer $\mathcal{S}'$, by assuming the role of the honest user $\mathcal{U}'$. When the protocol terminates, $\mathcal{B}$ obtains a signature σ on message μ', and common information *info*. He sets $\sigma' \leftarrow (\sigma, r)$, stores the tuple $(\mu', \mathrm{info}, \sigma')$ in a list L, and outputs σ', along with the corresponding transcript *trans* to the adversary $\mathcal{U}^*$.

Forgery: Since $\mathcal{U}^*$ is efficient, he eventually stops and outputs a single *info*, and a sequence of message–signature pairs: $(\mu^*_1, \sigma^*_1), \ldots, (\mu^*_{k_{\mathrm{info}}+1}, \sigma^*_{k_{\mathrm{info}}+1})$. In turn, $\mathcal{B}$ retrieves all message–signature pairs $(\mu'_1, \sigma'_1), \ldots, (\mu'_{n_{\mathrm{info}}}, \sigma'_{n_{\mathrm{info}}})$ pertaining to that particular *info* from L (and discards the rest). He then parses σ^*_i as $(\tilde{\sigma}_i, r^*_i)$, sets $\tilde{\mu}_i \leftarrow \mu^*_i \parallel r^*_i, \forall i = 1, \ldots, k_{\mathrm{info}}+1$, and outputs $(\mu'_1, \sigma'_1), \ldots, (\mu'_{n_{\mathrm{info}}}, \sigma'_{n_{\mathrm{info}}})$, and $(\tilde{\mu}_1, \tilde{\sigma}_1), , \ldots, (\tilde{\mu}_{k_{\mathrm{info}}+1}, \tilde{\sigma}_{k_{\mathrm{info}}+1})$.

Analysis: As $\mathcal{U}^*$ runs in polynomial–time and all queries are handled efficiently, $\mathcal{B}$ runs in polynomial time as well. Since $\mathcal{U}^*$ succeeds in $\mathrm{Exp}^{\mathrm{hu-omf}}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$, he outputs a single *info* and $k_{\mathrm{info}}+1$ valid message–signature pairs. $\mathcal{B}$ simulated the honest-user algorithm $\mathcal{U}'$ to compute the message–signature pairs: $(\mu'_1, \sigma'_1), \ldots, (\mu'_{n_{\mathrm{info}}}, \sigma'_{n_{\mathrm{info}}})$, thus all these pairs are valid with overwhelming probability (due to completeness).

Observe that all messages are pairwise distinct. Indeed, consider the messages $(\mu'_1, \ldots, \mu'_{n_{\mathrm{info}}})$ and $(\tilde{\mu}_1, \ldots, \tilde{\mu}_{k_{\mathrm{info}}+1})$, pertaining to common information *info*. These are of the form $\mu_{i'} = \mu_i \parallel r_i, \forall i = 1, \ldots, n_{\mathrm{info}}$ and $\tilde{\mu}_j = \mu^*_j \parallel r^*_j, \forall j = 1, \ldots, k_{\mathrm{info}}+1$, respectively. As the r_i are chosen uniformly at random from $\{0,1\}^n$, it follows that $(\mu'_1, \ldots, \mu'_{n_{\mathrm{info}}})$ are pairwise distinct with overwhelming probability. Similarly, because $\mathcal{U}^*$ wins in $\mathrm{Exp}^{\mathrm{hu-omf}}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$, messages $(\mu^*_1, \ldots, \mu^*_{k_{\mathrm{info}}+1})$ are pairwise distinct and thus, $(\tilde{\mu}_1, \ldots, \tilde{\mu}_{k_{\mathrm{info}}+1})$ are also distinct. Moreover, by definition we have $\{\mu_1, \ldots, \mu_{n_{\mathrm{info}}}\} \cap \{\mu^*_1, \ldots, \mu^*_{k_{\mathrm{info}}+1}\} = \emptyset$, and thus, $\mu'_i \neq \tilde{\mu}_j, \forall i, j$.

Next, we show that $\mathcal{B}$ could produce one more message–signature pair than the number of successful, complete protocol interactions with $\mathcal{S}'$. As $\mathcal{U}^*$ wins in experiment $\mathrm{Exp}^{\mathrm{hu-omf}}_{\mathcal{U}^*,\mathrm{PBSS}}(n)$, it follows that in at most k_{info} of the protocol executions that $\mathcal{B}$ relayed between $\mathcal{U}^*$ and $\mathcal{S}'$, the signer returned 'ok'. Furthermore, $\mathcal{B}$ executed a total of n_{info} honest-user instances to simulate oracle $\mathcal{P}$. Since $\mathcal{U}^*$ successfully outputs $k_{\mathrm{info}}+1$ message–signature pairs for pairwise distinct messages μ_i, it follows that $\mathcal{B}$ has asked a total of at most $k_{\mathrm{info}} + n_{\mathrm{info}}$ queries in which $\mathcal{S}'$ returned 'ok'. However, $\mathcal{B}$ returned a total of $n_{\mathrm{info}} + k_{\mathrm{info}} + 1$ message–signature pairs for *info*, which contradicts our assumption that PBSS is unforgeable.