Designs, Codes and Cryptography (2023) 91:497–544
https://doi.org/10.1007/s10623-022-01114-3

Code-based signatures from new proofs of knowledge for the syndrome decoding problem

Loïc Bidoux[1] · Philippe Gaborit[2] · Mukul Kulkarni[1] · Victor Mateu[1]

Received: 13 January 2022 / Revised: 6 September 2022 / Accepted: 8 September 2022 /
Published online: 13 October 2022

Abstract
In this paper, we study code-based signatures constructed from Proofs of Knowledge (PoK). This line of work can be traced back to Stern who introduced the first efficient PoK for the syndrome decoding problem in 1993 (Stern in A new identification scheme based on syndrome decoding. In: International cryptology conference (CRYPTO), 1993). Afterwards, different variations were proposed in order to reduce signature's size. In practice, obtaining a smaller signature size relies on the interaction of two main considerations: (i) the underlying protocol and its soundness error and (ii) the types of optimizations which are compatible with a given protocol. In particular, optimizations related to the possibility of using random seeds instead of long vectors have a great impact on the final signature length. Over the years, different variations were proposed to improve the Stern scheme such as the Veron scheme (with public key as a noisy codeword rather than a syndrome) (Véron in Appl Algebra Eng Commun Comput 8(1):57-69, 1997), the AGS scheme which is a 5-pass protocol with soundness error asymptotically equal to 1/2 (Aguilar et al. in A new zero-knowledge code based identification scheme with reduced communication. In: IEEE information theory workshop, 2011) and more recently the FJR approach which permits to decrease the soundness probability to 1/N but induces a performance overhead (Feneuil et al. in Shared permutation for syndrome decoding: new zero-knowledge protocol and code-based signature. Cryptology ePrint archive, report 2021/1576, 2021). Overall the length of the signature depends on a trade-off between: the scheme in itself, the possible optimizations and the cost of the implementation. For instance, depending on the application one may prefer a 30% shorter signature at the cost of a ten times slower implementation rather than a longer signature but a faster implementation. The recent approaches which increase the cost of the implementation open the door to many different types of trade-offs. In this paper we propose three new schemes and different trade-offs, which are all interesting in themselves, since depending on potential future optimizations a scheme may eventually become more efficient than another. All the schemes we propose use a trusted helper: the first scheme permits to get a soundness error

Communicated by M. Albrecht.

✉ Loïc Bidoux
loic.bidoux@tii.ae

1 Technology Innovation Institute, Abu Dhabi, UAE

2 University of Limoges, Limoges, France

of 1/2, the second scheme permits to decrease the soundness error to 1/N but with a different approach than the recent FJR scheme and at last the third scheme proposes a Veron-like adaptation of the FJR scheme in which the public key is a noisy codeword rather than a syndrome. We provide extensive comparison which lists various trade-offs between our schemes and previous ones. The table highlights the benefits of our constructions for certain types of trade-offs.

Keywords Code-based cryptography · Signature · Proof of knowledge

Mathematics Subject Classification 11T71

1 Introduction

The goal of post-quantum cryptography is to provide cryptographic schemes that are secure against adversaries using both classical and quantum computers. Code-based cryptography was introduced by McEliece in 1978 [35] and nowadays is one of the main alternatives to classical cryptography. This is illustrated by the ongoing NIST Post-Quantum Cryptography standardization process [18] whose round 3 features three code-based Key Encapsulation Mechanisms (KEM) [3–5]. Unlike the code-based KEM, designing digital signatures from coding theory has historically been challenging. Two approaches have been studied in this regard namely signatures from the hash-and-sign paradigm and signatures based on proofs of knowledge (PoK). Regarding signatures based on the hash-and-sign paradigm, a first (although inefficient) construction was proposed in 2001 [19]. The Wave construction [20] follows the same approach and features small signature sizes. Regarding code-based signatures from PoK, the Schnorr-Lyubashevsky [34, 38] approach has been successfully used in the rank metric setting by the Durandal scheme [6]. In this paper, we focus on the Fiat–Shamir paradigm [24, 36] which relies on zero-knowledge PoK. In this approach, one transforms an honest verifier zero-knowledge interactive PoK into a signature scheme using the so called Fiat–Shamir heuristic.

The first efficient PoK for the syndrome decoding (SD) problem over $\mathbb{F}_2$ was introduced by Stern in 1993 [40]. In 1997, Véron improved the Stern protocol by designing a protocol based on the general decoding problem (GD) problem rather than the SD one [44]. The SD and GD problem are equivalent and only differ in the way used to represent the underlying code namely using a parity-check matrix in the former and using a generator matrix in the latter. Stern and Véron protocols feature a soundness error equal to $2/3$ and as such need to be repeated several times in order to achieve a negligible soundness error. In 2011, two 5-round code-based PoK reducing the soundness error close to $1/2$ (hence leading to smaller signature sizes) were proposed. The first one (CVE) relies on the SD problem over $\mathbb{F}_q$ [16] while the second one (AGS) relies on the QCGD problem over $\mathbb{F}_2$ namely the quasi-cyclic variant of the GD problem [2]. A zero-knowledge issue impacting GD based protocols (Véron and AGS) was identified in [29] and fixed in [12]. In addition, the AGS protocol has been improved by the BGKS proposal by using an optimization specifically tailored to the QCSD problem [15]. Furthermore, it has been shown recently (see related work section below for additional details) that one can design a protocol achieving an arbitrarily small soundness error in [26] and [23]. Some of the aforementioned protocols have been adapted to the rank metric setting, see [9, 17, 25].

Table 1 PoK for SD/GD or QCSD/QCGD problems in Hamming metric

Soundness	SD/GD over $\mathbb{F}_2$	QC-SD/GD over $\mathbb{F}_2$	SD/GD over $\mathbb{F}_q$
2/3	Stern [40]		
	Véron [44], [12]		
1/2	PoK 1 (Sect. 3)	AGS [2], [12]	CVE [16]
		BGKS [15]	
		PoK 1 (Sect. 3)	
1/N	FJR [23]	FJR [23]	GPS [26]
	PoK 2 (Sect. 4.1)	PoK 2 (Sect. 4.1)	
	PoK 3 (Sect. 4.2)	PoK 3 (Sect. 4.2)	

Recently, Katz, Kolesnikov and Wang [31], designed a signature scheme based on PoK using the MPC-in-the-head paradigm [28] in the preprocessing model. An important highlight of this design is that it allows to achieve smaller soundness errors which can result in shorter signatures. While this benefit comes at the cost of slightly involved protocols and performance overhead, with careful analysis and parameter selection it is possible to design signature schemes with acceptable performance and shorter sizes using this framework. Beullens generalized the work of [31] by introducing the notion of PoK with trusted helper and designing new PoK for the Multivariate Quadratic (MQ) problem, Permuted Kernel Problem (PKP) and Short Integer Solution (SIS) problem [13]. In this work, we propose new code-based PoK with trusted helper for the decoding problem, and later construct signature schemes from these PoK by using the Fiat–Shamir transformation.

Contributions We introduce three new PoK with trusted helper for the decoding problem over $\mathbb{F}_2$. The first one (denoted PoK 1) is a PoK for the SD problem achieving a soundness error equal to $1/2$ without any extra assumption such as using quasi-cyclic variants of the problem or working over $\mathbb{F}_q$. The second one (denoted PoK 2) is a PoK for the GD problem over $\mathbb{F}_2$ achieving an arbitrarily small soundness error. The third one (denoted PoK 3) is a variant of PoK 2 using some ideas from [23]. Our proofs (as well as that in [23]) can leverage the quasi-cyclic variants of the SD or GD problems to improve their performances although this is not mandatory. Table 1 compares our new PoK to existing ones with respect to their soundness error along with their respective underlying security assumptions. The soundness error is directly linked to the resulting signature size (once the Fiat–Shamir heuristic have been applied) hence the smaller the soundness error is, the more compact the signature can be. Regarding security assumptions, the SD problem over $\mathbb{F}_2$ has been arguably more studied than its counterpart over $\mathbb{F}_q$ hence can be considered as a slightly more conservative assumption. The QCSD and QCGD constitute structured variants of the initial SD and GD problems and as such are less conservative than the latter although they are believed to be hard by the community. Similarly to our PoK, protocols over $\mathbb{F}_q$ could also use quasi-cyclicity to improve their performances which is not depicted in Table 1.

In addition, we explain how to transform our PoK with trusted helper into 3-round PoK without helper or 5-round PoK without helper. These two transformations offer different trade-offs between communication cost and security. Using 5-round PoK leads to smaller signature sizes however the security proof of the Fiat–Shamir heuristic is less tight in this case. In practice, this means that one needs to take into account attacks such as the one from [30]. We consider both transformations for our PoK 1 and denote the resulting PoK without helper by 3-round PoK 1 and 5-round PoK 1 respectively. For the PoK 2 and PoK 3, we only consider the first transformation thus leading to PoK without helper denoted 3-round PoK 2 and 3-round PoK 3 respectively. Furthermore, we present several optimizations for these PoK and describe how to convert them into signature schemes. Our first signature is built from our 3-round PoK 1 and is denoted Sig 1 (3-round). It features the most conservative design possible as it relies on the SD problem over $\mathbb{F}_2$ along with an underlying 3-round structure. Our second signature is built from our 5-round PoK 1 and is denoted Sig 1 (5-round). Both signatures can be instantiated using the plain SD problem or its quasi-cyclic variant QCSD. Sig 1 (3-round) and Sig 1 (5-round) both outperform the Stern [40], Véron [44] and AGS [2] schemes with respect to signature size for comparable settings at the cost of a small performance overhead. In addition, they achieve similar performances to [15] while relying on more conservative security assumptions. Sig 2 and Sig 3 are constructed from 3-round PoK 2 and 3-round PoK 3 respectively and feature even smaller signature sizes however at the cost of a bigger performance overhead. Sig 2 outperforms the recent proposal from [26] but is outperformed by the proposal from [23]. Finally, Sig 3 closes this performance gap by mixing PoK 2 with the shared permutation idea from [23].

Related work Gueron, Persichetti and Santini have recently proposed a new code-based signature built from a PoK with trusted helper for the SD problem over $\mathbb{F}_q$ that achieve an arbitrarily small soundness error [26]. Recently in an independent and concurrent work, Feneuil, Joux and Rivain have proposed a code-based signature based on a PoK for the SD problem over $\mathbb{F}_2$ that achieves an arbitrarily small soundness error [23]. These works present some similarities with our PoK 2 and its associated signature Sig 2. A PoK for the SD problem requires to prove two statements: (i) there exists a value $\mathbf{x}$ such that $\mathbf{H}\mathbf{x}^\top = \mathbf{y}^\top$ and (ii) the weight of $\mathbf{x}$ is small. To achieve an arbitrarily small soundness error, one needs to prove both statements simultaneously which is challenging to do while preserving the zero-knowledge property of the underlying proof. Indeed, one generally proves the first property by masking $\mathbf{x}$ using $\mathbf{u} + \mathbf{x}$ with a uniform random value $\mathbf{u}$ and proves the second property by masking $\mathbf{x}$ using $\pi[\mathbf{x}]$ for some random permutation π while reconciling the two parts of the proof thanks to a third value such as $\pi[\mathbf{u} + \mathbf{x}]$. The authors of [26] solve this issue by *revealing the permutation and later cancelling it* in their proof. As such, their proposal reveals π rather than $\pi[\mathbf{x}]$ contrary to the existing protocols. The authors of [23] solve the aforementioned issue by introducing what they called a *shared permutation* namely by masking the permutation during the computation of some permuted vectors. Doing so, they are able to compute a value related to $\pi[\mathbf{u} + \mathbf{x}]$ from $\mathbf{u} + \mathbf{x}$ without revealing anything on π. Our PoK 2 relies on another approach by *introducing several permutations and revealing all of them but one* in order to prove the knowledge of the solution of a permuted decoding problem instance.

We briefly discuss the main differences between these three approaches. The PoK from [26] relies on the SD problem over $\mathbb{F}_q$ while our PoK 2 relies on the SD problem over $\mathbb{F}_2$. As the SD problem over $\mathbb{F}_2$ has been arguably more studied than its counterpart over $\mathbb{F}_q$, it can be considered as a more conservative assumption. Furthermore, using the protocol from [26], one has to send the permutation π (which is fixed hence not replaceable by a seed) to prove the weight of $\mathbf{x}$ while our protocol only requires to send $\pi[\mathbf{x}]$ (which is a small weight vector hence can be compressed). As sending a permutation of a vector of size n over $\mathbb{F}_q$ is costly, the communication cost associated with the GPS proposal is bigger than the communication cost of our PoK 2. In practice, this means that for comparable parameters, our Sig 2 outperforms the signature from [26].

The PoK from [23] and our PoK 2 are more closely related as they are both based on the SD problem over $\mathbb{F}_2$ and both achieve an arbitrarily small soundness error equal to $1/N$.

As such, they are equivalent from a theoretical point of view. Nonetheless, the optimized version of the FJR protocol outperforms the optimized version of our PoK 2 in practice. This is explained by the fact that some optimizations related to commitment compression bring a better improvement for the FJR protocol than for our PoK 2. As a result, we also introduce PoK 3 which mixes PoK 2 with the shared permutation setting of [23]. Doing so, one can consider that our PoK 3 is a dual version (Véron-like) of the protocol from [23].

Paper organization We start by describing some preliminaries related to code-based cryptography and PoK in Sect. 2. We present our new PoK with trusted helper in Sects. 3 and 4 respectively. Then, we explain how to remove the trusted helper from the aforementioned protocols in Sect. 5. Several optimizations reducing the bandwidth cost of these PoK are described in Sect. 6. We explain how to transform our PoK into signature schemes in Sect. 7. Parameters for these new signatures are provided in Sect. 8 along with a comparison to existing code-based signatures. To finish, we discuss some generalizations and variants of our PoK in Sect. 9 followed by the conclusion in Sect. 10.

2 Preliminaries

Notations Hereafter, vectors (respectively matrices) are represented using bold lower-case (respectively upper-case) letters. Also, the vectors are assumed to be row vectors by default, and we denote the column vectors by the transpose of row vectors (such as $\mathbf{x}^T$). The Hamming weight (number of non-zero coordinates) of a vector $\mathbf{x}$ is denoted by $w_H(\mathbf{x})$. For an integer $n > 0$, we use $\mathcal{S}_n$ to denote the symmetric group of all permutations of n elements. For a finite set S, $x \xleftarrow{\$} S$ denotes that x is sampled uniformly at random from S while $x \xleftarrow{\$,\theta} S$ denotes that x is sampled uniformly at random from S using the seed θ. In addition, we use the acronym PPT as an abbreviation for the term "probabilistic polynomial time". We also call a function *negligible* and denote it by $\mathsf{negl}(\cdot)$ if for all sufficiently large $\lambda \in \mathbb{N}$, $\mathsf{negl}(\lambda) < \lambda^{-c}$, for all constants $c > 0$.

2.1 Code-based cryptography

We start by defining binary linear codes and quasi-cyclic codes. Then, we describe the syndrome decoding (SD) and general decoding (GD) problems which are hard problems commonly used in code-based cryptography. These problems are equivalent and differ only in the way used to represent the underlying code namely using a parity-check matrix in the former and using a generator matrix in the latter. The SD problem has been proven NP-complete in [10]. In addition, we also introduce the quasi-cyclic problems QCSD and QCGD which are structured variants of the SD and GD problems.

Definition 1 (Binary Linear Code) Let n and k be positive integers such that $k < n$. A binary linear $\mathcal{C}$ code (denoted $[n, k]$) is a k-dimensional subspace of $\mathbb{F}_2^n$. $\mathcal{C}$ can be represented in two equivalent ways: by a generator matrix $\mathbf{G} \in \mathbb{F}_2^{k\times n}$ such that $\mathcal{C} = \{\mathbf{mG} \mid \mathbf{m} \in \mathbb{F}_2^k\}$ or by a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$ such that $\mathcal{C} = \{\mathbf{x} \in \mathbb{F}_2^n \mid \mathbf{Hx}^\top = 0\}$.

Definition 2 (Systematic Binary Quasi-Cyclic Code) A systematic binary quasi-cyclic code of index ℓ and rate $1/\ell$ is a $[n = \ell k, k]$ code that can be represented by a $k \times \ell k = k \times n$ generator matrix $\mathbf{G} \in \mathcal{QC}(\mathbb{F}_2^{k\times n})$ of the form:

$$\mathbf{G} = \left[\mathbf{I}_k \ \mathbf{A}_0 \ \cdots \ \mathbf{A}_{\ell-2}\right]$$

where $\mathbf{A}_0, \ldots, \mathbf{A}_{\ell-2}$ are circulant $k \times k$ matrices. Alternatively, it can be represented by an $(\ell-1)k \times \ell k = (n-k) \times n$ parity check matrix $\mathbf{H} \in \mathcal{QC}(\mathbb{F}_2^{(n-k)\times k})$ of the form:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}_k & \cdots & 0 & \mathbf{B}_0 \\ & \ddots & & \vdots \\ 0 & \cdots & \mathbf{I}_k & \mathbf{B}_{\ell-2} \end{bmatrix}$$

where $\mathbf{B}_0, \ldots, \mathbf{B}_{\ell-2}$ are circulant $k \times k$ matrices.

Definition 3 (SD problem) Given positive integers n, k, w, a random parity-check matrix $\mathbf{H} \xleftarrow{\$} \mathbb{F}_2^{(n-k)\times n}$ and a syndrome $\mathbf{y} \in \mathbb{F}_2^{n-k}$, the syndrome decoding problem $\mathsf{SD}(n, k, w)$ asks to find $\mathbf{x} \in \mathbb{F}_2^n$ such that $\mathbf{H}\mathbf{x}^\top = \mathbf{y}^\top$ and $w_H(\mathbf{x}) = w$.

Definition 4 (GD problem) Given positive integers n, k, w, a random generator matrix $\mathbf{G} \xleftarrow{\$} \mathbb{F}_2^{k\times n}$ and a vector $\mathbf{y} \in \mathbb{F}_2^n$, the general decoding problem $\mathsf{GD}(n, k, w)$ asks to find $(\mathbf{x}, \mathbf{e}) \in \mathbb{F}_2^k \times \mathbb{F}_2^n$ such that $\mathbf{x}\mathbf{G} + \mathbf{e} = \mathbf{y}$ and $w_H(\mathbf{e}) = w$.

Definition 5 (ℓ-QCSD problem) Given positive integers n, k, w, with $n = \ell k$ for some ℓ, a random parity-check matrix of a quasi-cyclic code $\mathbf{H} \xleftarrow{\$} \mathcal{QC}(\mathbb{F}_2^{(n-k)\times n})$ and a syndrome $\mathbf{y} \in \mathbb{F}_2^{(n-k)}$, the syndrome decoding problem ℓ-$\mathsf{QCSD}(n, k, w)$ asks to find $\mathbf{x} \in \mathbb{F}_2^n$, such that $\mathbf{H}\mathbf{x}^\top = \mathbf{y}^\top$ and $w_H(\mathbf{x}) = w$.

Definition 6 (ℓ-QCGD problem) Given positive integers n, k, w, with $n = \ell k$ for some ℓ, a random generator matrix of a quasi-cyclic code $\mathbf{G} \xleftarrow{\$} \mathcal{QC}(\mathbb{F}_2^{k\times n})$ and a vector $\mathbf{y} \in \mathbb{F}_2^n$, the general decoding problem ℓ-$\mathsf{QCGD}(n, k, w)$ asks to find $(\mathbf{x}, \mathbf{e}) \in \mathbb{F}_2^k \times \mathbb{F}_2^n$ such that $\mathbf{x}\mathbf{G} + \mathbf{e} = \mathbf{y}$ and $w_H(\mathbf{e}) = w$.

2.2 Commitment schemes

We now introduce commitment schemes as they are building blocks commonly used to construct proofs of knowledge. We require such schemes to be both hiding and binding. The former property ensures that the commitment does not leak any information on the committed message while the latter ensures that adversaries cannot change their committed messages once the commitment is sent. We now present the formal definition of the commitment schemes.

Definition 7 (Commitment Scheme) A (non-interactive) commitment scheme with underlying message space $\mathcal{M}$ is a tuple of algorithms (Keygen, Com, Open) such that:

- Keygen: Takes the security parameter λ as input and outputs a pair of keys. $(\mathsf{gk}, \mathsf{ck}) \longleftarrow \mathsf{Keygen}(1^\lambda)$. Here gk is called the *setup key* and it serves as an implicit input to the Com and Open algorithms, whereas ck is called the *commitment key* and it is given to the sender. Note that the commitment key ck can be set to an empty string ε, if only the setup key gk is sufficient for committing to the messages.
- Com: Takes a message $m \in \mathcal{M}$ and commitment key ck as input and outputs a commitment c and opening d. Formally, $(c, d) \longleftarrow \mathsf{Com}(\mathsf{ck}, m)$.
- Open: Takes a commitment c, opening d, message $m \in \mathcal{M}$ as input and outputs a bit $b \in \{0, 1\}$ indicating whether the commitment c is a valid commitment of m. Formally, $b := \mathsf{Open}(c, d, m)$.

The commitment scheme is *perfectly correct* if $\forall \lambda \in \mathbb{N}$, $\forall m \in \mathcal{M}$ and for all valid key pairs $(\mathsf{gk}, \mathsf{ck})$,

$$\Pr[\mathsf{Open}(\mathsf{Com}(\mathsf{ck}, m), m) = 1] = 1.$$

The commitment scheme satisfies two security properties guaranteeing security from a malicious sender (prover) and from a malicious receiver (verifier):

- *Hiding:* It is computationally hard for an efficient adversary $\mathcal{A}$ to generate two distinct messages $m_0, m_1 \in \mathcal{M}$, such that $\mathcal{A}$ can distinguish between their respective commitments. Formally, for any PPT adversary $\mathcal{A}$ it should hold that,

$$\Pr\left[b = b' \;\middle|\; \begin{array}{c} (\mathsf{gk}, \mathsf{ck}) \longleftarrow \mathsf{Keygen}(1^\lambda),\ (m_0, m_1) \longleftarrow \mathcal{A}(\mathsf{gk}, \mathsf{ck}) \\ b \xleftarrow{\$} \{0,1\}, (c, d) \longleftarrow \mathsf{Com}(m_b, \mathsf{ck}), b' \longleftarrow \mathcal{A}(c) \end{array}\right] = \frac{1}{2} + \mathsf{negl}(\lambda).$$

- *Binding:* It is computationally hard for an efficient adversary to generate a triple (c, d, d') such that *both* (c, d) and (c, d') are valid commitment/opening pairs for some $m, m' \in \mathcal{M}$ respectively, where $m \neq m'$. Formally, for any PPT adversary it should hold that,

$$\Pr\left[\begin{array}{l} m \neq m' \bigwedge \\ m, m' \in \mathcal{M} \end{array} \;\middle|\; \begin{array}{c} (\mathsf{gk}, \mathsf{ck}) \longleftarrow \mathsf{Keygen}(1^\lambda),\ (c, d, d') \longleftarrow \mathcal{A}(\mathsf{gk}, \mathsf{ck}) \\ 1 := \mathsf{Open}(c, d, m),\ 1 := \mathsf{Open}(c, d', m') \end{array}\right] \leq \mathsf{negl}(\lambda).$$

In this work, we assume that the commitment scheme is implemented using a collision-resistant hash function H modelled as random oracle. To commit to a message m, we first sample a random value $r \leftarrow \{0,1\}^\lambda$ and compute the commitment as $c := H(r, m)$. The hiding follows since H is modelled as random oracle and the binding follows from the collision-resistance of H. The random value r serves as the opening d. The verifier can simply re-compute $H(r, m)$ on receiving r as opening and check if $H(r, m)$ equals c.

2.3 Proofs of knowledge with helper

Following the work of Katz, Kolesnikov and Wang [31], Beullens introduced the notion of sigma protocols with helper in [13]. Given a relation $R = (x, w)$, these Honest-Verifier Zero-Knowledge Proofs of Knowledge (HVZK PoK) allow a prover to convince an honest verifier (namely a verifier that follows the protocol as described) that it knows a witness w for the statement x without revealing anything on w. In our context, the relation $R = (x, w)$ is defined by an instance of the SD problem such that $x = (\mathbf{H}, \mathbf{y})$ and $w = \mathbf{x}$ namely the prover convinces the verifier that it knows a solution to an SD instance without revealing anything on its solution. Alternatively, when the GD form of the problem is considered, one has $x = (\mathbf{G}, \mathbf{y})$ and $w = (\mathbf{x}, \mathbf{e})$.

Definition 8 (Sigma Protocol with Helper [13]) A protocol is a Sigma Protocol with helper for relation R with challenge space $\mathcal{C}$ if it follows the form of Fig. 1 and satisfies:

- **Completeness** If all parties (Helper, Prover and Verifier) follow the protocol on input $(x, w) \in R$, then the verifier always accepts.
- **Special soundness** From an adversary $\mathcal{A}$ that outputs two valid transcripts $(x, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$ and $(x, \mathsf{aux}, \mathsf{com}, \alpha', \mathsf{rsp}')$ with $\alpha \neq \alpha'$ and where $\mathsf{aux} = \mathsf{Setup}(\theta)$ for some seed value θ (not necessarily known to the extractor), there exists an extractor Ext that efficiently extracts a witness w such that $(x, w) \in R$ with probability $1 - \mathsf{negl}(\lambda)$.
- **Special honest-verifier zero-knowledge** There exists a PPT simulator Sim that on input x, a random seed value θ and a random challenge α outputs a transcript $(x, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$

$\underline{\mathsf{Helper}(x)}$

$\theta \xleftarrow{\$} \{0,1\}^{\lambda}$

$\mathsf{aux} \longleftarrow \mathsf{Setup}(\theta)$

Send θ to the Prover and aux to the $\mathsf{Verifier}$

$\underline{\mathsf{Prover}(x, w, \theta)}$		$\underline{\mathsf{Verifier}(x, \mathsf{aux})}$
$(\mathsf{com},) \longleftarrow P_1(x, w, \theta)$		
	$\xrightarrow{\quad \mathsf{com} \quad}$	
		$\alpha \xleftarrow{\$} \mathcal{C}$
	$\xleftarrow{\quad \alpha \quad}$	
$\mathsf{rsp} = P_2(x, w, \theta, , \alpha)$		
	$\xrightarrow{\quad \mathsf{rsp} \quad}$	
		return $V(x, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$

Fig. 1 Structure of an HVZK PoK with Trusted Helper [13]

with $\mathsf{aux} = \mathsf{Setup}(\theta)$ that is computationally indistinguishable from the probability distribution of transcript of honest executions of the protocol on input (x, w) for some witness w such that $(x, w) \in R$, conditioned on the auxiliary information being equal to aux and the challenge being equal to α.

2.4 Signature schemes

In this section, we define signatures based on the Fiat–Shamir transform [24] and then present the security definitions associated with these schemes.

Definition 9 (Digital Signature) A digital signature scheme SIG is a tuple of algorithms $\mathsf{SIG} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$,

- Gen takes security parameter λ as input and outputs the key pair $(\mathsf{pk}, \mathsf{sk})$.
- Sign takes a message m along with the secret (signing) key sk as input and produces signature σ as output.
- The verification algorithm Verify takes the public (verification) key pk, message m', and signature σ as input and returns accept or reject.

A signature scheme SIG is said to have correctness error ε if for all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^{\lambda})$, and all messages $m \in \mathcal{M}$, it holds true that

$$\mathsf{Pr}\left[\mathsf{Verify}(\mathsf{pk}, \mathsf{Sign}(\mathsf{sk}, m), m) = \mathsf{reject}\right] \leq \varepsilon.$$

Definition 10 A binary relation R with instance generator IG is called *hard* if for any (quantum) adversary $\mathcal{A}$, it holds that

$$\mathsf{Pr}\left[(x, \tilde{w}) \in R \mid (x, w) \leftarrow \mathsf{IG},\ \tilde{w} \leftarrow \mathcal{A}(x)\right]$$

is negligible, for any IG that always outputs a valid pair $(x, w) \in R$.

Definition 11 (Fiat–Shamir Signature) A Fiat–Shamir signature scheme based on a public-coin interactive proof system (or Σ-protocol) $\Pi = (\mathsf{Prover}, \mathsf{Verifier})$ for a hard relation R with instance generator IG, denoted by $\mathsf{SIG}[\Pi]$ is a tuple of algorithms $\mathsf{SIG}[\Pi] = (\mathsf{GenFS}, \mathsf{SignFS}, \mathsf{VerifyFS})$,

- GenFS samples $(x, w) \leftarrow \mathsf{IG}$ then outputs $\mathsf{sk} := (x, w)$ and $\mathsf{pk} := x$.
- $\mathsf{SignFS}^H(\mathsf{pk}, \mathsf{sk}, m)$ outputs (m, σ) where $\sigma \leftarrow \mathsf{Prover}^H(x, w, m)$.
- $\mathsf{VerifyFS}^H(\mathsf{pk}, \sigma, \tilde{m})$ runs $\mathsf{Verifier}^H(x, \sigma, \tilde{m})$ and returns its output.

Hereafter, we assume that the algorithms have oracle access to hash function H which is modelled as (quantum) random oracle.

Definition 12 (Strong Existentially Unforgeable Signatures under Chosen Message Attack (sEUF-CMA)) A signature scheme possesses *strong existential unforgeability under chosen message attack* (sEUF-CMA) if for all (quantum) polynomial-time algorithms $\mathcal{A}$ and for uniformly random H, it holds that

$$\Pr\left[\mathsf{Verify}^H(\mathsf{pk}, \sigma, m) \bigwedge (m, \sigma) \notin \mathbf{Sign} - \mathbf{q} \mid (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen},\ (m, \sigma) \leftarrow \mathcal{A}^{H, \mathbf{Sign}}(\mathsf{pk})\right]$$

is negligible. Here, **Sign** is a classical oracle which on input m returns $\mathsf{Sign}^H(\mathsf{pk}, \mathsf{sk}, m)$ and $\mathbf{Sign} - \mathbf{q}$ is the list of all (q) queries made to **Sign**.

3 PoK 1: Stern protocol improvement

The first PoK for the SD problem over $\mathbb{F}_2$ was introduced by Stern in 1993 [40]. This 3-round protocol features a soundness error equal to $2/3$ and as such needs to be repeated several times in order to achieve a negligible soundness error. Over the years, 5-round code-based PoK reducing the soundness error to $1/2$ (hence providing smaller communication costs) have been proposed. Such protocols either rely on the SD problem over $\mathbb{F}_q$ [16] or leverage the structured QCGD and QCSD problems over $\mathbb{F}_2$ [2, 15]. Hereafter, we introduce a PoK for the SD problem over $\mathbb{F}_2$ with soundness error equal to $1/2$. Our new protocol (denoted PoK 1) can be either seen as (i) a modification of the initial Stern protocol leveraging the MPC-in-the-head paradigm along with several optimizations from [15] or as (ii) the BGKS protocol [15] in which the quasi-cyclicity is replaced by the use of the MPC-in-head technique.

The initial Stern protocol permits to prove the knowledge of $\mathbf{x}$ such that $\mathbf{y}^\top = \mathbf{H}\mathbf{x}^\top$ and $w_H(\mathbf{x}) = w$. Within the protocol, one proves the knowledge of $\mathbf{x}$ using $\mathbf{x} + \mathbf{u}$ for some random value $\mathbf{u}$ and prove that $w_H(\mathbf{x}) = w$ using $\pi[\mathbf{x}]$ for some random permutation π. To this end, the prover starts by generating three commitments related to π, $\mathbf{x}$ and $\mathbf{u}$. Next, the verifier samples a random challenge from $\{0, 1, 2\}$ and the prover outputs a response that is specific to the received challenge. Amongst these three possible responses, one can be computed without knowing the secret $\mathbf{x}$ hence could be verified using the MPC-in-the-head paradigm. Doing so, one can reduce the challenge space to $\{0, 1\}$ thus achieving a soundness error equal to $1/2$. Our PoK 1 follows this approach and is depicted as a PoK with helper in Fig. 2.

We explain in Sect. 5 how to remove the helper from Fig. 2 in order to get both a 3-round HVZK PoK and a 5-round HVZK PoK. Our 3-round PoK 1 features a very conservative design (SD assumption over $\mathbb{F}_2$ only, tighter Fiat–Shamir transformation proof thanks to the 3-round structure) therefore is comparable to the Stern [40] and Véron [44] proposals which provide the same security guarantees. Our 3-round PoK 1 benefits from a smaller signature

$\underline{\mathsf{Helper}(\mathbf{H}, \mathbf{y})}$

$\theta \xleftarrow{\$} \{0,1\}^\lambda,\ \pi \xleftarrow{\$,\theta} \mathcal{S}_n,\ \mathbf{u} \xleftarrow{\$,\theta} \mathbb{F}_2^n$

$r_0 \xleftarrow{\$,\theta} \{0,1\}^\lambda,\ \mathsf{com}_0 = \mathsf{Com}\big(r_0,\ \pi \,||\, \mathbf{H}\mathbf{u}^\top\big)$

$r_1 \xleftarrow{\$,\theta} \{0,1\}^\lambda,\ \mathsf{com}_1 = \mathsf{Com}\big(r_1,\ \pi[\mathbf{u}]\big)$

$\mathsf{aux} = (\mathsf{com}_0, \mathsf{com}_1)$

Send θ to the Prover and aux to the $\mathsf{Verifier}$

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{H}, \mathbf{y}, \theta)}$		$\underline{\mathsf{Verifier}(\mathbf{H}, \mathbf{y}, \mathsf{aux})}$
Compute r_0, r_1, π and $\mathbf{u}$ from θ		
$r \xleftarrow{\$} \{0,1\}^\lambda,\ \mathsf{com} = \mathsf{Com}\big(r,\ \pi[\mathbf{u}+\mathbf{x}]\big)$		
	$\xrightarrow{\mathsf{com}}$	
	$\xleftarrow{\alpha}$	$\alpha \xleftarrow{\$} \{0,1\}$
if $\alpha = 0$ **then** $z_1 = \pi,\ \mathbf{z}_2 = \mathbf{u}+\mathbf{x}$ $\mathsf{rsp} = (r_0,\ r,\ z_1,\ \mathbf{z}_2)$ **end**		
if $\alpha = 1$ **then** $\mathbf{z}_3 = \pi[\mathbf{u}],\ \mathbf{z}_4 = \pi[\mathbf{x}]$ $\mathsf{rsp} = (r_1,\ r,\ \mathbf{z}_3,\ \mathbf{z}_4)$ **end**		
	$\xrightarrow{\mathsf{rsp}}$	**if** $\alpha = 0$ **then** $b_1 \longleftarrow \big(\mathsf{com}_0 = \mathsf{Com}\big(r_0,\ z_1 \,||\, \mathbf{H}\mathbf{z}_2^\top - \mathbf{y}^\top\big)\big)$ $b_2 \longleftarrow \big(\mathsf{com} = \mathsf{Com}\big(r,\ z_1[\mathbf{z}_2]\big)\big)$ **return** $b_1 \wedge b_2$ **end**
		if $\alpha = 1$ **then** $b_1 \longleftarrow \big(\mathsf{com}_1 = \mathsf{Com}\big(r_1,\ \mathbf{z}_3\big)\big)$ $b_2 \longleftarrow \big(\mathsf{com} = \mathsf{Com}\big(r,\ \mathbf{z}_3 + \mathbf{z}_4\big)\big)$ $b_3 \longleftarrow \big(w_H(\mathbf{z}_4) = \omega\big)$ **return** $b_1 \wedge b_2 \wedge b_3$ **end**

Fig. 2 ZK PoK with Helper for the SD problem over $\mathbb{F}_2$

size than the Stern and Véron protocols at the cost of a small performance overhead due to the use of the MPC-in-the-head. When coupled with quasi-cyclicity, our 5-round PoK 1 is comparable to the AGS [2] and BGKS [15] protocols (QCSD assumption over $\mathbb{F}_2$, 5-round structure) while being more conservative security-wise as it relies on the QCSD problem directly rather than the DiffSD problem contrary to the AGS and BGKS protocols (see [15], Definition 11 for a description of the DiffSD problem). Similarly to the 3-round case, our 5-round PoK 1 features a smaller signature size than the AGS and BGKS protocols at the cost of a small performance overhead.

Theorem 1 (Proof of knowledge with helper) *If the commitment used is binding and hiding, then the protocol depicted in Fig. 2 is a proof of knowledge with helper for the SD problem with challenge space $\mathcal{C}$ such that $|\mathcal{C}| = 2$.*

Proof We need to prove that the protocol in Fig. 2 satisfies the properties of correctness, special soundness, and special honest-verifier zero-knowledge.

Correctness The correctness follows straightforwardly from the protocol description once the commitments are verified.

Special soundness Given an adversary $\mathcal{A}$ that outputs with non negligible probability two valid transcripts $(\mathbf{H}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$ and $(\mathbf{H}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha', \mathsf{rsp}')$ with $\alpha \neq \alpha'$ and where $\mathsf{aux} = \mathsf{Setup}(\theta)$ for some random seed θ, one can easily build a knowledge extractor Ext that returns a solution to the SD instance defined by $(\mathbf{H}, \mathbf{y})$:

1. Compute and output $z_1^{-1}[\mathbf{z}_4]$.

As $\alpha \neq \alpha'$, the extractor Ext has access to both transcripts $(z_1, \mathbf{z}_2)$ and $(\mathbf{z}_3, \mathbf{z}_4)$ therefore it can output $z_1^{-1}[\mathbf{z}_4]$. We now explain why the extractor's output is a solution to the considered SD problem instance. Using the binding property of com_0 and com_1, one has $z_1 = \pi$ and $\mathbf{H}\mathbf{z}_2^\top - \mathbf{y}^\top = \mathbf{H}\mathbf{u}^\top$ as well as $z_3 = \pi[\mathbf{u}]$. In addition, from the binding property of com, one has $z_1[\mathbf{z}_2] = \mathbf{z}_3 + \mathbf{z}_4$ thus $\mathbf{z}_2 = \mathbf{u} + \pi^{-1}[\mathbf{z}_4]$. Using this expression within $\mathbf{H}\mathbf{z}_2^\top - \mathbf{y}^\top = \mathbf{H}\mathbf{u}^\top$, one can deduce that $\mathbf{H}(\pi^{-1}[\mathbf{z}_4])^\top = \mathbf{y}^\top$. Given that $w_H\big(\mathbf{z}_4\big) = \omega$, one also has $w_H\big(\pi^{-1}[\mathbf{z}_4]\big) = \omega$ thus $z_1^{-1}[\mathbf{z}_4]$ is a solution to the considered SD problem instance. Finally, Ext runs in polynomial time which completes the proof.

Special honest-verifier zero-knowledge We start by explaining why valid transcripts do not leak anything on the secret. A valid transcript contains either $(\pi, \mathbf{u} + \mathbf{x})$ or $(\pi[\mathbf{u}], \pi[\mathbf{x}])$ namely the secret $\mathbf{x}$ is masked either by a random value $\mathbf{u}$ or by some random permutation π. We now explain how to build a PPT simulator Sim that given $(\mathbf{H}, \mathbf{y})$, a random seed θ and a random challenge α outputs a transcript $(\mathbf{H}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$ such that $\mathsf{aux} = \mathsf{Setup}(\theta)$ that is indistinguishable from the probability distribution of transcripts of honest executions of the protocol:

1. Compute $(r_0, r_1, \mathbf{u}, \pi)$ from θ
2. If $\alpha = 0$, compute $\tilde{\mathbf{x}}$ such that $\mathbf{H}\tilde{\mathbf{x}} = \mathbf{y}$ (without constraint on the weight of $\mathbf{x}$)

   If $\alpha = 1$, compute $\tilde{\mathbf{x}} \xleftarrow{\$} \mathcal{S}_\omega(\mathbb{F}_2^n)$
3. Compute $r \xleftarrow{\$} \{0, 1\}^\lambda$, $\tilde{\mathsf{com}} = \mathsf{Com}\big(r, \pi[\mathbf{u} + \tilde{\mathbf{x}}]\big)$
4. If $\alpha = 0$, compute $z_1 = \pi$, $\tilde{\mathbf{z}}_2 = \mathbf{u} + \tilde{\mathbf{x}}$ and $\tilde{\mathsf{rsp}} = (r_0, r, z_1, \tilde{\mathbf{z}}_2)$

   If $\alpha = 1$, compute $\mathbf{z}_3 = \pi[\mathbf{u}]$, $\tilde{\mathbf{z}}_4 = \pi[\tilde{\mathbf{x}}]$ and $\tilde{\mathsf{rsp}} = (r_1, r, \mathbf{z}_3, \tilde{\mathbf{z}}_4)$
5. Output $(\mathbf{H}, \mathbf{y}, \mathsf{aux}, \tilde{\mathsf{com}}, \alpha, \tilde{\mathsf{rsp}})$

The transcript generated by the simulator Sim is $(\mathbf{H}, \mathbf{y}, \mathsf{aux}, \tilde{\mathsf{com}}, \alpha, \tilde{\mathsf{rsp}})$ where $\mathsf{aux} \longleftarrow \mathsf{Setup}(\theta)$. One need to check that $\tilde{\mathsf{com}}$ and $\tilde{\mathsf{rsp}}$ are indistinguishable in the simulation and during the real execution. If the commitment used is hiding, then com and $\tilde{\mathsf{com}}$ are indistinguishable in the simulation and during the real execution. When $\alpha = 0$, one cannot distinguish between $\tilde{\mathbf{z}}_2$ and $\mathbf{z}_2$ as $\mathbf{u}$ is sampled uniformly at random. When $\alpha = 1$, one cannot distinguish between $\tilde{\mathbf{z}}_4$ and $\mathbf{z}_4$ as $\pi[\tilde{\mathbf{x}}]$ follows the same probability distribution as $\pi[\mathbf{x}]$, since π is a random permutation. As a consequence, rsp and $\tilde{\mathsf{rsp}}$ are indistinguishable in the simulation and during the real execution. Finally, Sim runs in polynomial time which completes the proof.

4 PoK 2 & 3: arbitrarily small soundness error

In the previous section, we have leveraged the MPC-in-the-head technique in order to design a PoK for the SD problem over $\mathbb{F}_2$ achieving a soundness error of $1/2$. Hereafter, we present two PoK for the GD problem over $\mathbb{F}_2$ achieving an arbitrarily small soundness error equal to $1/N$ for some parameter N.

4.1 Reducing soundness using several permutations

We start by highlighting a particularity of PoK for the SD problem namely that it requires to prove two statements: (i) there exists a value $\mathbf{x}$ such that $\mathbf{H}\mathbf{x}^\top = \mathbf{y}^\top$ and (ii) the weight of $\mathbf{x}$ is small. As a consequence, it is very natural to design these proofs with two parts checking respectively each one of the aforementioned properties (see [2, 15, 16] as well as our construction from Sect. 3) which leads to a soundness error equal to $1/2$. In order to reduce the soundness error even further, one needs to merge the two parts of the proof together which turn out to be challenging to do while preserving the zero-knowledge property of the underlying proof. Indeed, one generally (see for instance Fig. 2) proves the first property by masking $\mathbf{x}$ using $\mathbf{u}+\mathbf{x}$ for some random value $\mathbf{u}$ and prove the second property by masking $\mathbf{x}$ using $\pi[\mathbf{x}]$ for some random permutation π. The verifier can then convince itself by checking some third value such as $\pi[\mathbf{u}+\mathbf{x}]$ which binds the two parts of the proof together. To enforce that the same permutation π is used in both $\pi[\mathbf{x}]$ and $\pi[\mathbf{u}+\mathbf{x}]$, we can commit to the permutation π and reveal it later such that at any given point (during or after the execution of the protocol) the verifier either knows π or $\pi[\mathbf{x}]$ but not both.

Our second PoK (hereafter denoted PoK 2) solves this issue by *introducing several permutations and revealing of all them but one* in order to prove the knowledge of the solution of a permuted syndrome decoding problem instance. In particular, we need to ensure the protocol guarantees soundness (i.e. one can extract the permutations used in the protocol, whenever more than one valid transcripts are given), and preserves the zero-knowledge (i.e. all of the permutations cannot be retrieved from any given (single) valid transcript). Our PoK relies on the GD problem namely the SD problem defined with a generator matrix instead of a parity-check matrix. Given a GD instance $(\mathbf{G}, \mathbf{y})$, we consider N permuted instances $(\pi_i[\mathbf{G}], \pi_i[\mathbf{y}])_{i\in[1,N]}$ satisfying $\pi_i[\mathbf{x}\mathbf{G}] + \pi_i[\mathbf{e}] = \pi_i[\mathbf{y}]$. Here, the solution to the GD problem $(\mathbf{x}, \mathbf{e})$ is the secret witness. By adding random values $\mathbf{u}$ and $\mathbf{v}_i$, one get an equivalent equation namely $\pi_i[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_i + \pi_i[\mathbf{e}] = \pi_i[\mathbf{y}+\mathbf{u}\mathbf{G}] + \mathbf{v}_i$. Using the random mask $\mathbf{v}$ is necessary as failing to do so (like in the initial Véron protocol) leads to a zero-knowledge issue that was identified in [29] and then fixed in [12]. To summarize, by adding random values $\mathbf{u}$ and $\mathbf{v}_i$, one get an equivalent equation namely $\pi_i[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_i + \pi_i[\mathbf{e}] = \pi_i[\mathbf{y}+\mathbf{u}\mathbf{G}] + \mathbf{v}_i$. This can be used for the verification while preserving the zero-knowledge. We now explain how our PoK 2 achieve an arbitrarily small soundness error. As shown in Fig. 3, the helper can compute a commitment of $\pi_i[\mathbf{y}+\mathbf{u}\mathbf{G}] + \mathbf{v}_i$ as this value does not involve any secret information. Thus, one can design a PoK for the GD problem by revealing both $\pi_i[\mathbf{e}]$ and $\pi_i[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_i$. We now explain how to use the cut-and-choose technique on the N permutations of the considered GD instance in order to ensure that the latter value $\pi_i[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_i$ has been correctly computed. Given some challenge $\alpha \in [1, N]$, the prover can reveal (i) the masked secret $\mathbf{u}+\mathbf{x}$, (ii) all the permutations and random masks $\mathbf{v}_i$ except for the instance α (denoted by $(\pi_i, \mathbf{v}_i)_{i\in[1,N]\setminus\alpha}$) as well as (iii) the value to be checked $\pi_\alpha[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_\alpha$, in the instance α. The verifier can then recompute $\pi_i[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_i$ for all $i \in [1, N] \setminus \alpha$, using the public value $\mathbf{G}$. This enforces that $\pi_\alpha[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_\alpha$ has been correctly generated

$\underline{\mathsf{Helper}(\mathbf{G}, \mathbf{y})}$

$\theta \xleftarrow{\$} \{0,1\}^\lambda$
for $i \in [1, N]$ **do**
  $\theta_i \xleftarrow{\$,\theta} \{0,1\}^\lambda, \ \pi_i \xleftarrow{\$,\theta_i} \mathcal{S}_n$
  $\mathbf{u}_i \xleftarrow{\$,\theta_i} \mathbb{F}_2^k, \ \mathbf{v}_i \xleftarrow{\$,\theta_i} \mathbb{F}_2^n$
end
$\mathbf{u} = \sum_{i \in [1,N]} \mathbf{u}_i$
for $i \in [1, N]$ **do**
  $r_i \xleftarrow{\$,\theta_i} \{0,1\}^\lambda, \ \theta_{i^*} = (\theta_j)_{j \in [1,N] \setminus i}$
  $\mathsf{com}_i = \mathsf{Com}(r_i, \ \pi_i[\mathbf{y} + \mathbf{u}\mathbf{G}] + \mathbf{v}_i \,||\, \theta_{i^*})$
end
$\mathsf{aux} = (\mathsf{com}_i)_{i \in [1,N]}$
Send θ to the Prover and aux to the Verifier

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{e}, \mathbf{G}, \mathbf{y}, \theta)}$

Compute $(\pi_i, \mathbf{u}_i, \mathbf{v}_i, \theta_{i^*})_{i \in [1,N]}$ and $\mathbf{u}$ from θ
for $i \in [1, N]$ **do**
  $\mathbf{s}_i = \pi_i[(\mathbf{u} + \mathbf{x})\mathbf{G}] + \mathbf{v}_i$
end
$r \xleftarrow{\$} \{0,1\}^\lambda, \ \mathbf{s} = (\mathbf{s}_i)_{i \in [1,N]}$
$\mathsf{com} = \mathsf{Com}(r, \ \mathbf{u} + \mathbf{x} \,||\, \mathbf{s})$

$\xrightarrow{\mathsf{com}}$

$\underline{\mathsf{Verifier}(\mathbf{G}, \mathbf{y}, \mathsf{aux})}$

$\alpha \xleftarrow{\$} [1, N]$

$\xleftarrow{\alpha}$

$\mathbf{z}_1 = \mathbf{u}_\alpha + \mathbf{x}, \ \mathbf{z}_2 = \pi_\alpha[\mathbf{e}]$
$\mathbf{z}_3 = \mathbf{s}_\alpha, \ z_4 = \theta_{\alpha^*}$
$\mathsf{rsp} = (r_\alpha, r, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, z_4)$

$\xrightarrow{\mathsf{rsp}}$

Compute $(\bar{\pi}_i, \bar{\mathbf{u}}_i, \bar{\mathbf{v}}_i)_{i \in [1,N] \setminus \alpha}$ from z_4
$\bar{\mathbf{z}}_1 = \mathbf{z}_1 + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i$
for $i \in [1, N] \setminus \alpha$ **do**
  $\bar{\mathbf{s}}_i = \bar{\pi}_i[\bar{\mathbf{z}}_1 \mathbf{G}] + \bar{\mathbf{v}}_i$
end
$\bar{\mathbf{s}} = (\bar{\mathbf{s}}_1, \cdots, \bar{\mathbf{s}}_{\alpha-1}, \mathbf{z}_3, \bar{\mathbf{s}}_{\alpha+1}, \cdots, \bar{\mathbf{s}}_N)$

$b_1 \longleftarrow (\mathsf{com} = \mathsf{Com}(r, \ \bar{\mathbf{z}}_1 \,||\, \bar{\mathbf{s}}))$
$b_2 \longleftarrow (\mathsf{com}_\alpha = \mathsf{Com}(r_\alpha, \ \mathbf{z}_3 + \mathbf{z}_2 \,||\, z_4))$
$b_3 \longleftarrow (w_H(\mathbf{z}_2) = \omega)$
return $b_1 \wedge b_2 \wedge b_3$

Fig. 3 ZK PoK with Helper for the GD problem over $\mathbb{F}_2$

except with arbitrarily small probability $1/N$. As a technicality, our proof also requires $\mathbf{u}$ to be extractable for the soundness hence we define it as $\mathbf{u} = \sum_{i \in [1,N]} \mathbf{u}_i$. The resulting protocol is described in Fig. 3. We explain in Sect. 5 how to remove the helper in order to construct a 3-round HVZK PoK. Our PoK 2 achieves an arbitrarily small soundness error and therefore leads to small signatures at the cost of a significant overhead on performances.

Theorem 2 (Proof of knowledge with helper) *If the commitment scheme is computationally binding and computationally hiding, then the protocol depicted in Fig. 3 is a proof of* *knowledge with helper for the* GD *problem with challenge space* $\mathcal{C}$ *such that* $|\mathcal{C}| = N$*, with computational soundness error* $1/N$ *and honest-verifier computational zero-knowledge.* [1]

Proof We prove the correctness, special soundness and special honest-verifier zero-knowledge properties below.

Correctness We prove the correctness by showing that the input to the com and com_α used by the verifier in the final steps of Fig. 3 are same as the input provided by the prover while generating com and com_α. Once the inputs are shown to be identical, the correctness follows from the correctness of the commitment scheme. We begin by considering the inputs to com, we need to show that $\bar{\mathbf{z}}_1 = \mathbf{u} + \mathbf{x}$ and $\bar{\mathbf{s}} = \mathbf{s}$. Note that the verifier computes $\bar{\mathbf{z}}_1$ as $\mathbf{z}_1 + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i$. Here, $\mathbf{z}_1$ is computed by the prover as $(\mathbf{u}_\alpha + \mathbf{x})$, and the second term is sum of all $\mathbf{u}_i$ except $\mathbf{u}_\alpha$. Therefore, it is easy to see that $\bar{\mathbf{z}}_1 = (\mathbf{u}_\alpha + \mathbf{x}) + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i = \mathbf{u} + \mathbf{x}$. Next, note that the verifier has access to all seeds θ_i except θ_α from the z_4 sent by the prover in the response rsp. The verifier can therefore compute all the permutation π_i and the random masks $\mathbf{v}_i$ except π_α and $\mathbf{v}_\alpha$. As shown in previous step, the verifier also knows the value $\bar{\mathbf{z}}_1 = \mathbf{u} + \mathbf{x}$. The verifier can therefore compute $\bar{\mathbf{s}}_i = \pi_i[\bar{\mathbf{z}}_1 \mathbf{G}] + \mathbf{v}_i$ for all $i \in [1, N] \setminus \alpha$ since $\mathbf{G}$ is public which is same as $\bar{\mathbf{s}}_i = \pi_i[(\mathbf{u} + \mathbf{x})\mathbf{G}] + \mathbf{v}_i$ for all $i \in [1, N] \setminus \alpha$, but this is exactly how the $\mathbf{s}_i$ values are computed by the prover. Therefore, we have shown that $\bar{\mathbf{s}}_i = \mathbf{s}_i$ for all $i \neq \alpha$. However, $\bar{\mathbf{s}}_\alpha = \mathbf{z}_3 = \mathbf{s}_\alpha$ since $\mathbf{z}_3$ is computed by the prover. Therefore we have shown that $\bar{\mathbf{s}} = \mathbf{s}$. This concludes the part related the commitment com, since we have shown that both the inputs are identical. We now show the same for com_α. Here, we need to show that (i) $\mathbf{z}_3 + \mathbf{z}_2 = \pi_\alpha[\mathbf{y} + \mathbf{u}\mathbf{G}] + \mathbf{v}_\alpha$ and, (ii) $z_4 = \theta_{\alpha^*}$ where θ_{α^*} denotes all the seeds θ_i except for $i = \alpha$. It is easy to verify and has been discussed earlier that z_4 computed by the prover as θ_{α^*} and sent as part of the response to the verifier. Thus, $z_4 = \theta_{\alpha^*}$ by just inspecting the prover's response. Recall, that $\mathbf{z}_3 = \mathbf{s}_\alpha$ and $\mathbf{s}_\alpha = \pi_\alpha[(\mathbf{u} + \mathbf{x})\mathbf{G}] + \mathbf{v}_\alpha$. Adding $\mathbf{z}_2$ to both sides and substituting $\mathbf{z}_2 = \pi_\alpha[\mathbf{e}]$, one get $\mathbf{z}_3 + \mathbf{z}_2 = \pi_\alpha[(\mathbf{u}+\mathbf{x})\mathbf{G}] + \mathbf{v}_\alpha + \pi_\alpha[\mathbf{e}] = \pi_\alpha[\mathbf{x}\mathbf{G} + \mathbf{e} + \mathbf{u}\mathbf{G}] + \mathbf{v}_\alpha = \pi_\alpha[\mathbf{y} + \mathbf{u}\mathbf{G}] + \mathbf{v}_\alpha$. Hence we have shown that both the inputs to com_α are also identical. As mentioned earlier, since the inputs to com and com_α are identical to their counterparts computed by the prover, the correctness of the protocol follows from the correctness of the commitment scheme.

Special soundness In order to prove the special soundness, we need to build an efficient knowledge extractor Ext which returns a solution of the GD instance defined by $(\mathbf{G}, \mathbf{y})$ with high probability, when provided with two valid transcripts $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$ and $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha', \mathsf{rsp}')$ with $\alpha \neq \alpha'$ generated by a PPT adversary (malicious prover) $\mathcal{A}$, where $\mathsf{aux} = \mathsf{Setup}(\theta)$ for some random seed θ. The knowledge extractor Ext computes the solution as:

1. Compute $(\pi_i, \mathbf{u}_i)_{i \in [1,N]}$ from z_4 and z_4'
2. Output $(\mathbf{z}_1 - \mathbf{u}_\alpha, \pi_\alpha^{-1}[\mathbf{z}_2])$

We now show that this can be computed efficiently by Ext, and then prove that the output is indeed a solution to the given GD problem. Recall from Fig. 3, that the prover's response is of the form $\mathsf{rsp} = (r_\alpha, r, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, z_4)$ and $\mathsf{rsp}' = (r_{\alpha'}, r', \mathbf{z}_1', \mathbf{z}_2', \mathbf{z}_3', z_4')$. Here, z_4 denotes all the seeds θ_i for $i \neq \alpha$ and z_4' denotes all the seeds θ_i for $i \neq \alpha'$. Since, $\alpha \neq \alpha'$, the Ext has access to all the seeds θ_i for $i \in [1, N]$. The extractor therefore can efficiently compute

[1] Proof of Knowledge systems with computational soundness are also called Arguments of Knowledge. Our PoK achieves computational ZK since the random masks $(\mathbf{u}, \mathbf{v})$ added to hide the secrets are generated from seeds with the help of pseudorandom objects such as XOF.

all the permutations π_i and masks $\mathbf{u}_i$ for $i \in [1, N]$, including π_α and $\mathbf{u}_\alpha$. Using these the extractor can efficiently compute and then output $(\mathbf{z}_1 - \mathbf{u}_\alpha, \pi_\alpha^{-1}[\mathbf{z}_2])$.

Let $\tilde{\ } = (\tilde{\theta}_i)_{i\in[1,N]}$ denote the seeds used to generate the commitments $(\mathsf{com}_i)_{i\in[1,N]}$ comprising the value aux. Note that if there exists an index $j \in [1, N]$, such that $\theta_j \neq \tilde{\theta}_j$ or $\theta'_j \neq \tilde{\theta}_j$, then that particular θ_j or θ'_j can be used to break the binding property of the commitment scheme as any commitment com_i in aux where $i \neq j$, can be opened as valid commitment for two distinct messages where one contains $\tilde{\theta}_j$ and the other contains θ_j. Thus, the binding property of the commitment scheme ensures that the extracted mask $\mathbf{u}_\alpha$ and the permutation π_α are same as those committed in the common information aux and the first commitment com. We now explain why the extractor's output is a solution to the considered GD problem instance. Note that $\mathbf{z}_1$ is computed as $\mathbf{z}_1 = \mathbf{u}_\alpha + \mathbf{x}$, therefore the extractor correctly recovers the secret $\mathbf{x}$ by computing $\mathbf{z}_1 - \mathbf{u}_\alpha$. Also, $\mathbf{z}_2 = \pi_\alpha[\mathbf{e}]$, which can be inverted by Ext after learning π_α as $\mathbf{e} = \pi_\alpha^{-1}[\mathbf{z}_2]$. This proves that extracted solution $(\mathbf{z}_1 - \mathbf{u}_\alpha, \pi_\alpha^{-1}[\mathbf{z}_2]) = (\mathbf{x}, \mathbf{e})$ is the correct solution to the GD problem.

Special honest-verifier zero-knowledge We start by explaining why valid transcripts do not leak anything on the secret $(\mathbf{x}, \mathbf{e})$. A valid transcript contains $(\mathbf{u}_\alpha + \mathbf{x}, \pi_\alpha[\mathbf{e}], \mathbf{s}_\alpha, (\pi_i, \mathbf{u}_i, \mathbf{v}_i)_{i\in[1,N]\setminus\alpha})$ namely the secret $\mathbf{x}$ is masked by a random value $\mathbf{u}_\alpha$ and the secret $\mathbf{e}$ is masked by a random permutation π_α. From the protocol, one can compute $\mathbf{u}_\alpha + \mathbf{x}$ and $\mathbf{u} + \mathbf{x}$ however this does not leak anything on $\mathbf{x}$ as $\mathbf{u}_\alpha$ and $\mathbf{u}$ are both unknown. The main difficulty concerns the permutation π_α as the protocol requires $\pi_\alpha[(\mathbf{u}+\mathbf{x})\mathbf{G}]$ to be computed while both $(\mathbf{u}+\mathbf{x})$ and $\mathbf{G}$ are known. To overcome this issue, the protocol actually computes $\pi_\alpha[(\mathbf{u}+\mathbf{x})\mathbf{G}]+\mathbf{v}_\alpha$ for some random value $\mathbf{v}_\alpha$ hence the transcript does not leak anything on $\mathbf{e}$. Formally, one can build a PPT simulator Sim that given the public values $(\mathbf{G}, \mathbf{y})$, a random seed θ and a random challenge α outputs a transcript $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$ such that $\mathsf{aux} = \mathsf{Setup}(\theta)$ that is computationally indistinguishable from the transcript of honest executions of the protocol:

1. Compute $(\pi_i, \mathbf{u}_i, \mathbf{v}_i, \theta_{i^*})_{i\in[1,N]}$ and $\mathbf{u}$ from θ
2. Compute $\tilde{\mathbf{x}} \xleftarrow{\$} \mathbb{F}_2^k$, $\tilde{\mathbf{e}} \xleftarrow{\$} \mathcal{S}_\omega(\mathbb{F}_2^n)$ and $\tilde{\mathbf{y}} = \tilde{\mathbf{x}}\mathbf{G} + \tilde{\mathbf{e}}$
3. Compute $\tilde{\mathbf{s}}_i = \pi_i[(\mathbf{u} + \tilde{\mathbf{x}})\mathbf{G}] + \mathbf{v}_i$ for all $i \in [1, N] \setminus \alpha$
4. Compute $\tilde{\mathbf{s}}_\alpha = \pi_\alpha[(\mathbf{u} + \tilde{\mathbf{x}})\mathbf{G}] + \mathbf{v}_\alpha + \pi_\alpha[\mathbf{y} - \tilde{\mathbf{y}}]$
5. Compute $r \xleftarrow{\$} \{0,1\}^\lambda$, $\tilde{\mathbf{s}} = (\tilde{\mathbf{s}}_i)_{i\in[1,N]}$ and $\tilde{\mathsf{com}} = \mathsf{Com}\big(r,\ (\mathbf{u} + \tilde{\mathbf{x}}) \,||\, \tilde{\mathbf{s}}\big)$
6. Compute $\tilde{\mathbf{z}}_1 = \mathbf{u}_\alpha + \tilde{\mathbf{x}}$, $\tilde{\mathbf{z}}_2 = \pi_\alpha[\tilde{\mathbf{e}}]$, $\tilde{\mathbf{z}}_3 = \tilde{\mathbf{s}}_\alpha$, $z_4 = \theta_{\alpha^*}$
7. Compute $\tilde{\mathsf{rsp}} = (r_\alpha, r, \tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2, \tilde{\mathbf{z}}_3, z_4)$ and output $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \tilde{\mathsf{com}}, \alpha, \tilde{\mathsf{rsp}})$

The transcript generated by the simulator Sim is $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \tilde{\mathsf{com}}, \alpha, \tilde{\mathsf{rsp}})$ where $\mathsf{aux} \longleftarrow \mathsf{Setup}(\theta)$. We now show that $\tilde{\mathsf{com}}$ and $\tilde{\mathsf{rsp}}$ are indistinguishable in the simulation and during the real execution. If the commitment used is hiding, then com and $\tilde{\mathsf{com}}$ are indistinguishable in the simulation and during the real execution. Since $\tilde{\mathbf{x}}$ (in the simulation) and $\mathbf{x}$ (in the real execution) are masked by a random mask $\mathbf{u}_\alpha$ which is unknown to the verifier, $\tilde{\mathbf{z}}_1$ and $\mathbf{z}_1$ are computationally indistinguishable. Similarly, $\tilde{\mathbf{e}}$ and $\mathbf{e}$ have same hamming weight ω, and are masked by random permutation π_α which is never known to the verifier. Thus, making $\tilde{\mathbf{z}}_2$ and $\mathbf{z}_2$ computationally indistinguishable. In addition, as the mask $\mathbf{v}_\alpha$ is sampled uniformly at random and is unknown to the verifier, it cannot distinguish between $\tilde{\mathbf{z}}_3$ and $\mathbf{z}_3$. Finally, $z_4 = \theta_{\alpha^*}$ is identical in both cases. As a consequence, rsp and $\tilde{\mathsf{rsp}}$ are computationally indistinguishable in the simulation and during the real execution. Finally, Sim runs in polynomial time which completes the proof.

4.2 Reducing soundness using a shared permutation

The PoK 2 presented in the previous section achieves an arbitrarily small soundness error equal to $1/N$. As such, it is theoretically equivalent to the proposal from [23]. Nonetheless, the optimized version of the FJR protocol outperforms the optimized version of our PoK 2 in practice. This is explained by the fact that some optimizations bring a better improvement for the FJR protocol than for PoK 2. We defer the interested reader to the paragraph "Commitment compression" in Sect. 6 for additional details on this topic. In this section, we show how one can adapt our PoK 2 to the shared permutation setting used in [23] to achieve similar performances. The resulting protocol is denoted PoK 3 and can be seen as a dual version of the FJR protocol based on the GD problem rather than the SD one.

Theorem 3 (Proof of knowledge with helper) *If the commitment is binding and hiding, then the protocol depicted in Fig. 4 is a proof of knowledge with helper for the* GD *problem with challenge space* $\mathcal{C}$ *such that* $|\mathcal{C}| = N$*, with computational soundness error* $1/N$ *and honest-verifier computational zero-knowledge.*

Proof One need to prove the correctness, special soundness and special honest-verifier zero-knowledge properties to complete the proof.

Correctness The proof for correctness follows the same arguments as for the proof of correctness of Theorem 2, with the only difference being the $\mathbf{s}_i$ values are computed by composing the permutations π_i in a nested manner.

Special soundness Given an adversary $\mathcal{A}$ that outputs two valid transcripts $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$ and $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha', \mathsf{rsp}')$ with $\alpha \neq \alpha'$ and where $\mathsf{aux} = \mathsf{Setup}(\theta)$ for some random seed θ, one can build a knowledge extractor Ext that returns a solution of the GD instance defined by $(\mathbf{G}, \mathbf{y})$ with high probability as follows:

1. Compute $(\pi_i, \mathbf{u}_i)_{i \in [1,N]}$ from z_4 and z'_4
2. Compute $\pi = \pi_N \circ \cdots \circ \pi_1$
3. Output $(\mathbf{z}_1 - \mathbf{u}_\alpha, \pi^{-1}[\mathbf{z}_2])$

The proof of soundness showing that the extractor Ext is efficient and returns a valid solution for the GD instance follows the same ideas as for the proof of soundness of Theorem 2, with the only difference being computing the permutation π as composition of the permutations π_i after extracting π_α.

Special honest-verifier zero-knowledge The proof of zero-knowledge follows the same arguments as for the proof of zero-knowledge of Theorem 2. We provide the description of the PPT simulator Sim which generates the indistinguishable transcript using only the public information for the completeness below. Formally, one can build a PPT simulator Sim that given the public values $(\mathbf{G}, \mathbf{y})$, a random seed θ and a random challenge α outputs a transcript $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \mathsf{com}, \alpha, \mathsf{rsp})$ such that $\mathsf{aux} = \mathsf{Setup}(\theta)$ that is computationally indistinguishable from the probability distribution of transcripts of honest executions of the protocol:

$\underline{\mathsf{Helper}(\mathbf{G}, \mathbf{y})}$

$\theta \xleftarrow{\$} \{0,1\}^\lambda$

for $i \in [1, N]$ **do**

  $\theta_i \xleftarrow{\$,\theta} \{0,1\}^\lambda,\ \pi_i \xleftarrow{\$,\theta_i} \mathcal{S}_n$

  $\mathbf{u}_i \xleftarrow{\$,\theta_i} \mathbb{F}_2^k,\ \mathbf{v}_i \xleftarrow{\$,\theta_i} \mathbb{F}_2^n$

end

$\pi = \pi_N \circ \cdots \circ \pi_1$

$\mathbf{u} = \sum_{i\in[1,N]} \mathbf{u}_i$

$\mathbf{v} = \mathbf{v}_N + \sum_{i\in[1,N-1]} \pi_N \circ \cdots \circ \pi_{i+1}[\mathbf{v}_i]$

for $i \in [1, N]$ **do**

  $r_i \xleftarrow{\$,\theta_i} \{0,1\}^\lambda,\ \theta_{i^*} = (\theta_j)_{j\in[1,N]\setminus i}$

  $\mathsf{com}_i = \mathsf{Com}\big(r_i,\ \pi[\mathbf{y} + \mathbf{uG}] + \mathbf{v} \,||\, \theta_{i^*}\big)$

end

$\mathsf{aux} = (\mathsf{com}_i)_{i\in[1,N]}$

Send θ to the Prover and aux to the Verifier

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{e}, \mathbf{G}, \mathbf{y}, \theta)}$

Compute $(\pi_i, \mathbf{u}_i, \mathbf{v}_i, \theta_{i^*})_{i\in[1,N]}$ and $\mathbf{u}, \pi$ from θ

$\mathbf{s}_0 = (\mathbf{u} + \mathbf{x})\mathbf{G}$

for $i \in [1, N]$ **do**

  $\mathbf{s}_i = \pi_i[\mathbf{s}_{i-1}] + \mathbf{v}_i$

end

$r \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{s} = (\mathbf{s}_i)_{i\in[1,N]}$

$\mathsf{com} = \mathsf{Com}\big(r,\ \mathbf{u} + \mathbf{x} \,||\, \mathbf{s}\big)$

$\xrightarrow{\mathsf{com}}$

$\underline{\mathsf{Verifier}(\mathbf{G}, \mathbf{y}, \mathsf{aux})}$

$\alpha \xleftarrow{\$} [1, N]$

$\xleftarrow{\alpha}$

$\mathbf{z}_1 = \mathbf{u}_\alpha + \mathbf{x},\ \mathbf{z}_2 = \pi[\mathbf{e}]$

$\mathbf{z}_3 = \mathbf{s}_\alpha,\ z_4 = \theta_{\alpha^*}$

$\mathsf{rsp} = (r_\alpha, r, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, z_4)$

$\xrightarrow{\mathsf{rsp}}$

Compute $(\bar{\pi}_i, \bar{\mathbf{u}}_i, \bar{\mathbf{v}}_i)_{i\in[1,N]\setminus\alpha}$ from z_4

$\bar{\mathbf{z}}_1 = \mathbf{z}_1 + \sum_{i\in[1,N]\setminus\alpha} \bar{\mathbf{u}}_i$

$\bar{\mathbf{s}}_0 = \bar{\mathbf{z}}_1\mathbf{G},\ \bar{\mathbf{s}}_\alpha = \mathbf{z}_3$

for $i \in [1, N] \setminus \alpha$ **do**

  $\bar{\mathbf{s}}_i = \bar{\pi}_i[\bar{\mathbf{s}}_{i-1}] + \bar{\mathbf{v}}_i$

end

$\bar{\mathbf{s}} = (\bar{\mathbf{s}}_i)_{i\in[1,N]}$

$b_1 \longleftarrow \big(\mathsf{com} = \mathsf{Com}\big(r,\ \bar{\mathbf{z}}_1 \,||\, \bar{\mathbf{s}}\big)\big)$

$b_2 \longleftarrow \big(\mathsf{com}_\alpha = \mathsf{Com}\big(r_\alpha,\ \bar{\mathbf{s}}_N + \mathbf{z}_2 \,||\, z_4\big)\big)$

$b_3 \longleftarrow \big(w_H\big(\mathbf{z}_2\big) = \omega\big)$

return $b_1 \wedge b_2 \wedge b_3$

Fig. 4 ZK PoK with Helper for the GD problem over $\mathbb{F}_2$

Prover(x, w)		Verifier(x)
for $k \in [1, M]$ **do**		
$\theta^{(k)} \xleftarrow{\$} \{0,1\}^{\lambda}$		
$\mathsf{aux}^{(k)} \longleftarrow \mathsf{Setup}(\theta^{(k)})$		
$(\mathsf{com}^{(k)}, {}^{(k)}) \longleftarrow P_1(x, w, \theta^{(k)})$		
end		
	$\xrightarrow{(\mathsf{aux}^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M]}}$	
		$\kappa \xleftarrow{\$} [1, M]$
		$\alpha \xleftarrow{\$} \mathcal{C}$
	$\xleftarrow{(\kappa, \alpha)}$	
$\mathsf{rsp} = P_2(x, w, \theta^{(\kappa)}, {}^{(\kappa)}, \alpha)$		
	$\xrightarrow{(\theta^{(k)})_{k \in [1,M] \setminus \kappa},\ \mathsf{rsp}}$	
		for $k \in [1, M] \setminus \kappa$ **do**
		Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$
		if $(\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})$ **return** 0
		end
		return $V(x, \mathsf{aux}^{(\kappa)}, \mathsf{com}^{(\kappa)}, \alpha, \mathsf{rsp})$

Fig. 5 3-round HVZK PoK from HVZK PoK with Trusted Helper where P_1 and P_2 are defined in Fig. 1 [13]

1. Compute $(\pi_i, \mathbf{u}_i, \mathbf{v}_i, \theta_{i^*})_{i \in [1,N]}$ and $\mathbf{u}, \pi$ from θ
2. Compute $\tilde{\mathbf{x}} \xleftarrow{\$} \mathbb{F}_2^k$, $\tilde{\mathbf{e}} \xleftarrow{\$} \mathcal{S}_\omega(\mathbb{F}_2^n)$ and $\tilde{\mathbf{y}} = \tilde{\mathbf{x}}\mathbf{G} + \tilde{\mathbf{e}}$
3. Compute $\tilde{\mathbf{s}}_0 = (\mathbf{u} + \tilde{\mathbf{x}})\mathbf{G}$ and $\tilde{\mathbf{s}}_i = \pi_i[\tilde{\mathbf{s}}_{i-1}] + \mathbf{v}_i$ for all $i \in [1, \alpha - 1]$
4. Compute $\tilde{\mathbf{s}}_\alpha = \pi_\alpha[(\mathbf{u} + \tilde{\mathbf{x}})\mathbf{G}] + \mathbf{v}_\alpha + \pi_{\alpha+1}^{-1} \circ \cdots \circ \pi_N^{-1} \circ \pi[\mathbf{y} - \tilde{\mathbf{y}}]$
5. Compute $\tilde{\mathbf{s}}_i = \pi_i[\tilde{\mathbf{s}}_{i-1}] + \mathbf{v}_i$ for all $i \in [\alpha + 1, N]$
6. Compute $r \xleftarrow{\$} \{0, 1\}^\lambda$, $\tilde{\mathbf{s}} = (\tilde{\mathbf{s}}_i)_{i \in [1,N]}$ and $\tilde{\mathsf{com}} = \mathsf{Com}\big(r,\ (\mathbf{u} + \tilde{\mathbf{x}})\,||\,\tilde{\mathbf{s}}\big)$
7. Compute $\tilde{\mathbf{z}}_1 = \mathbf{u}_\alpha + \tilde{\mathbf{x}}$, $\tilde{\mathbf{z}}_2 = \pi[\tilde{\mathbf{e}}]$, $\tilde{\mathbf{z}}_3 = \tilde{\mathbf{s}}_\alpha$, $z_4 = \theta_{\alpha^*}$
8. Compute $\tilde{\mathsf{rsp}} = (r_\alpha, r, \tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2, \tilde{\mathbf{z}}_3, z_4)$ and output $(\mathbf{G}, \mathbf{y}, \mathsf{aux}, \tilde{\mathsf{com}}, \alpha, \tilde{\mathsf{rsp}})$

5 PoK without trusted helper

PoK with helper can be transformed into either 3-round HVZK PoK without helper or 5-round HVZK PoK without helper using the cut-and-choose paradigm as explained in [13, 31]. When the 5-round transformation is used, one must take into account the attack from [30] that specifically exploits the fact that the proof has a 5-round structure. Choosing to use the 3-round or the 5-round transformation leads to different communication costs depending on the underlying proof and therefore must be decided on a case by case basis. Hereafter, we present both transformations and discuss which ones to consider for the proof of knowledge introduced in Sects. 3 and 4 respectively.

The main idea is to let the prover run the setup phase multiple times with many independent seeds $\mathsf{Setup}(\theta^{(k)})$ for $k \in [1, M]$ and then share the auxiliary information $\mathsf{aux}^{(k)}$ for all the instances with the verifier. The verifier then picks an arbitrary instance κ, and the prover sends all the seeds $\theta^{(k)}$ for $k \neq \kappa$ to the verifier. The verifier can then verify that the received auxiliary information $\mathsf{aux}^{(k)}$ has been honestly computed by running the setup algorithm itself, if this check does not pass then the verifier rejects. Otherwise, the prover and the verifier proceed with the protocol for PoK using the seed $\theta^{(\kappa)}$. If the soundness error of the PoK with the helper is $\frac{1}{N}$, and the protocol without helper runs the setup M times with independent seeds, then the soundness error of the PoK without helper is $\max(\frac{1}{M}, \frac{1}{N})$.

Theorem 4 (3-round proof of knowledge [13]) *If the commitment used is binding and hiding, then the protocol depicted in Fig. 5 is a 3-round honest-verifier zero-knowledge proof of knowledge with challenge space $\mathcal{C}_1 \times \mathcal{C}_2$ such that $|\mathcal{C}_1| = M$ and $|\mathcal{C}_2| = N$ and soundness error equal to* $\max(\frac{1}{M}, \frac{1}{N})$.

Proof This is a direct application of Theorem 3 from [13] that permits to build a 3-round PoK without helper from a PoK with helper.

Theorem 5 (5-round proof of knowledge) *If the commitment used is binding and hiding, then the protocol depicted in Fig. 6 is a 5-round honest-verifier zero-knowledge proof of knowledge with challenge space $\mathcal{C}_1 \times \mathcal{C}_2$ such that $|\mathcal{C}_1| = M$ and $|\mathcal{C}_2| = N$ and soundness error equal to* $\max(\frac{1}{M}, \frac{1}{N})$.

Proof One can straightforwardly adapt the proof of Theorem 3 from [13] to the 5-round setting.

Removing the helper from PoK 1 For our first PoK (see Sect. 3), we will consider both the 3-round and 5-round transformations. We defer the reader to Appendices A and D for the description of the 3-round PoK 1 and 5-round PoK 1 protocols which are obtained after respectively applying the 3-round and 5-round transformations to our fist proof of knowledge. The 3-round PoK 1 protocol is a slightly more conservative choice while the 5-round PoK 1 leads to a slightly smaller signature (see Sect. 8).

Removing the helper from PoK 2 and PoK 3 For our PoK 2 and PoK 3 (see Sect. 4.1), we will only consider the 3-round transformation as it leads to smaller signatures when taking account the attack from [30]. We defer the reader to Appendices G and J for the description of the protocols.

6 Communication cost and optimizations

In this section, we present optimizations that permits to reduce the communication cost of the aforementioned proofs of knowledge. Several optimizations are related to the use of the MPC-in-the-head paradigm and were first introduced in [31]. We also consider code-based cryptography related optimizations that were introduced in [2] and [15]. Finally, we discuss a performance oriented optimization relying on the use of structured matrices. The optimized versions of our 3-round PoK 1, 5-round PoK 1, 3-round PoK 2 and 3-round PoK 3 are described in Appendices B, E, H and K respectively.

Protocol repetition [13, 31]**.** The 3-round or 5-round PoK constructed by removing the helper (see Sect. 5) have a soundness error equal to $\max(\frac{1}{M}, \frac{1}{N})$. In order to obtain a negligible soundness error with respect to security parameter λ, one can compute τ parallel executions of the protocol where $\tau = \frac{\lambda}{\log_2(\min(M,N))}$. In that case, one needs to execute $\tau \cdot M$ setup steps (the Helper part in Fig. 1) followed by τ executions of the protocol (the Prover part

$\underline{\mathsf{Prover}(x, w)}$		$\underline{\mathsf{Verifier}(x)}$
for $k \in [1, M]$ **do**		
$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^{\lambda}$		
$\quad \mathsf{aux}^{(k)} \longleftarrow \mathsf{Setup}(\theta^{(k)})$		
end		
	$\xrightarrow{(\mathsf{aux}^{(k)})_{k \in [1,M]}}$	
		$\kappa \xleftarrow{\$} [1, M]$
	$\xleftarrow{\kappa}$	
$(\mathsf{com}^{(\kappa)},\ ^{(\kappa)}) \longleftarrow P_1(x,\ w,\ \theta^{(\kappa)})$		
	$\xrightarrow{(\theta^{(k)})_{k \in [1,M] \setminus \kappa},\ \mathsf{com}^{(\kappa)}}$	
		for $k \in [1, M] \setminus \kappa$ **do**
		$\quad$ Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$
		$\quad$ **if** $(\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})$ **return** 0
		end
		$\alpha \xleftarrow{\$} \mathcal{C}$
	$\xleftarrow{\alpha}$	
$\mathsf{rsp} = P_2(x,\ w,\ \theta^{(\kappa)},\ ^{(\kappa)},\ \alpha)$		
	$\xrightarrow{\mathsf{rsp}}$	
		return $V(x, \mathsf{aux}^{(\kappa)},\ \mathsf{com}^{(\kappa)},\ \alpha,\ \mathsf{rsp})$

Fig. 6 5-round HVZK PoK from HVZK PoK with Trusted Helper where P_1 and P_2 are defined in Fig. 1 [13]

in Fig. 1). The key idea of the *beating parallel repetition* optimization from [31] is to let the verifier choose τ out of M (instead of only 1 out of M) setups to execute, which means the setup phase is repeated only M times (instead of $\tau \cdot M$ times). However, this comes at the cost of increasing the number executions (higher τ) in the protocol as explained below. Suppose, a malicious prover computes $e \leq \tau$ setup steps incorrectly. It can only convince the verifier if the latter chooses to execute all these e setups (hence never verifying any of them) which happens with probability $\binom{M-e}{\tau-e} \cdot \binom{M}{\tau}^{-1}$. Moreover, the malicious prover needs to be accepted for the remaining $\tau - e$ executions with honest setups which can happen with probability $\leq (\frac{1}{N})^{\tau-e}$. Therefore, using this optimization, one needs to execute M setup steps followed by τ executions of the protocol and the soundness error is given by $\max\limits_{0 \leq e \leq \tau} \binom{M-e}{\tau-e} \cdot \binom{M}{\tau}^{-1} \cdot N^{-(\tau-e)}$.

Seed compression [31] For all the aforementioned PoK, one has to send the seeds $(\theta^{(k)})_{k \in [1,M] \setminus K}$ used to recompute the auxiliary information $(\mathsf{aux}^{(k)})_{k \in [1,M] \setminus K}$ (with $K \xleftarrow{\$} \{K \subset \{1, \cdots, M\},\ |K| = \tau\}$) associated with the setups that are not going to be executed. As explained in [31], one can use Merkle trees in order to reduce the cost of sending these seeds. To this end, the prover samples a root seed and generates a binary tree of depth $\log(M)$ where each node is a seed derived from its parent node. Doing so, the prover can send to the verifier the nodes that allows to recompute all the leaves of the tree except the τ seeds that should not be revealed thus reducing the cost associated with the $(\theta^{(k)})_{k \in [1,M] \setminus K}$ seeds from $(M - \tau) \cdot \lambda$ to $\tau \log_2(\frac{M}{\tau}) \cdot \lambda$ where λ is the security parameter. A crucial observation is that this optimization works well only when τ is small with respect to M. In particular

Table 2 Example of parameters trade-off for $\lambda = 128$

	Parallel repetition			[31] Repetition		
N	2	16	32	2	16	32
M	2	16	32	256	272	389
τ	128	32	26	128	35	28
# Setup	**256**	**512**	**832**	**256**	**272**	**389**
# Execution	**128**	**32**	**26**	**128**	**35**	**28**

The optimization greatly reduces the Setup's number for only a small increase of the Execution's number From here onward, the reader is advised to keep in mind that during the protocol, the signer needs to send information (seeds) corresponding to (i) $M - \tau$ instances of the underlying PoK which are generated during the setup phase but are only used to verify that the setup was run honestly and (ii) information associated with τ executions of the underlying PoK

for $\tau = M/2$, using a Merkle tree provides no benefit. Hence, in the case of PoK 1 where $\tau = M/2$, we employ a variant of this optimization by considering $M/2$ binary trees of depth 1 instead of single binary tree of depth $\log(M)$ which reduces the expected cost of sending the seeds to $3/4 \cdot (M - \tau) \cdot \lambda$ from $(M - \tau) \cdot \lambda$.

Commitment compression [2, 31] When considering 3-round PoK, the prover's first message contains the commitments $(\mathsf{aux}^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]}$ where $\mathsf{aux}^{(k)} = (\mathsf{com}_i^{(k)})_{i\in[1,N]}$. In order to reduce the cost associated with these commitments, one can instead send a unique commitment $h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)} \mid\mid \mathsf{com}^{(k)})_{k\in[1,M]}\big)$ as suggested in [2]. Doing so, the prover has to give the verifier all the commitments that the latter cannot recompute itself in order to allow the verifier to check the commitment h. The situation is similar for 5-round PoK with $h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)})_{k\in[1,M]}\big)$ and $h' = \mathsf{Com}\big(r',\ (\mathsf{com}^{(\kappa)})_{\kappa\in K}\big)$. In this case, the verifier can recompute h' by itself hence only the commitments contained in h have to be considered. To reduce the cost associated with these commitments, one can once again leverage Merkle trees as explained in [31]. Indeed, the prover will generate a Merkle tree of its commitments (from bottom to top contrary to the previous case) and send the root to the verifier as well as all the nodes of the tree that permits to recompute the root from the commitments that the verifier can obtain by itself.

In our PoK 1 (3-round), $(\mathsf{aux}^{(k)})_{k\in[1,M]}$ contains $2M$ commitments while $(\mathsf{com}^{(k)})_{k\in[1,M]}$ contains M commitments. For the $M-\tau$ instances that are not executed, these commitments can be recomputed by the verifier from the $(\theta^{(k)})_{k\in[1,M]\setminus K}$ seeds. For the τ instances that are executed, one commitment from $(\mathsf{aux}^{(\kappa)})_{\kappa\in K}$ can be recomputed by the verifier while the other one can be given in the prover's response rsp. In addition, all the commitments from $(\mathsf{com}^{(k)})_{k\in[1,M]\setminus K}$ need to be given to the verifier. Given that $\tau = M/2$, this reduces the cost of sending the $3M$ commitments of our PoK 1 (3-round) to $(1+3/4\cdot(M-\tau)+7/8\cdot\tau)\cdot|\mathsf{com}|$. For PoK 1 (5-round), the cost is reduced from $3M$ commitments to $(2 + 7/8 \cdot \tau) \cdot |\mathsf{com}|$.

For PoK 2 and PoK 3, sending the $(M - \tau)$ commitments $(\mathsf{com}^{(k)})_{k\in[1,M]\setminus K}$ can be done using Merkle trees hence cost $\tau \log_2(\frac{M}{\tau}) \cdot |\mathsf{com}|$. For the $M - \tau$ instances that are not executed, the $N(M - \tau)$ commitments from $(\mathsf{aux}^{(k)})_{k\in[1,M]\setminus K}$ can be recomputed by the verifier from the $(\theta^{(k)})_{k\in[1,M]\setminus K}$ seeds. Interestingly, for the τ instances that are executed, the situation differs between PoK 2 and PoK 3 which explains why the optimized version of PoK 3 outperforms the optimized version of PoK 2 although both non optimized versions are equivalent. In the case of PoK 2, only one commitment from $(\mathsf{aux}^{(\kappa)})_{\kappa\in K}$ can be recomputed by the verifier while the other $(N - 1)$ have to be given to it as it cannot recompute them due to the presence of the value $\mathbf{u}$. Hence, sending these commitments using Merkle trees cost

$\log(N) \cdot |\mathsf{com}|$. In contrast, in the [23] setting, $(N-1)$ commitments from $(\mathsf{aux}^{(\kappa)})_{\kappa \in K}$ can be recomputed by the verifier hence the cost associated with sending the commitments from $(\mathsf{aux}^{(\kappa)})_{\kappa \in K}$ is only $|\mathsf{com}|$.

Small weight vector compression [2] One can leverage the small weight of some vectors such as $\pi[\mathbf{x}]$ in PoK 1 or $\pi[\mathbf{e}]$ in PoK 2 and PoK 3 by using a compression algorithm before sending them. Hence, the cost of sending small weight vectors is reduced from n to $n/2$ approximately.

Additional vector compression [15] This optimization is specific to PoK 1 in which the prover has to send a permutation of a random vector $\pi[\mathbf{u}]$. Instead of doing this, one can sample a random value $\mathbf{v} \xleftarrow{\$, \psi} \mathbb{F}_2^n$ from some random seed ψ and compute the value $\mathbf{u} = \pi^{-1}[\mathbf{v}]$. When the prover has to send $\pi[\mathbf{u}]$, it can send $\mathbf{v}$ instead which can be substituted by the seed ψ. Doing so, one reduce the cost of sending such vectors from n to λ.
Improved performances from structured matrices In all aforementioned PoK, a matrix vector multiplication must be computed during each setup. In order to improve the performance of these protocols, one may choose to use structured matrices featuring an efficient matrix vector multiplication. For example, one can use quasi-cyclic matrices as their matrix vector multiplication can be performed efficiently by polynomial multiplication. In this case, the security of the protocol relies on the quasi-cyclic variants QCSD or QCGD of the syndrome decoding problem.

7 Signature schemes

In this section, we explain how to transform our interactive HVZK PoKs without helper as detailed in Sect. 5 into digital signatures using the strong Fiat–Shamir heuristic [11, 24]. We also discuss the security of the resulting signature schemes in both the random oracle model (ROM) and the quantum random oracle model (QROM).
The keystone idea of the Fiat–Shamir heuristic [24] is to "emulate" the random challenge sampling from the verifier by a call to a hash function modelled as a random oracle, thereby turning an interactive protocol into non-interactive protocol with access to random oracle. Figures 7 and 8 explain how to apply the Fiat–Shamir heuristic in the context of PoK with trusted helper. We next present the signatures obtained by applying the Fiat–Shamir transform to HVZK PoK schemes from Sects. 3 and 4. Starting with our PoK 1 with Helper (Sect. 3, Fig. 2), PoK 2 with Helper (Sect. 4.1, Fig. 3) and PoK 3 with Helper (Sect. 4.2, Fig. 4), one can remove the helper using the constructions from Sect. 5. Doing so, one get our non-optimized 3-round PoK 1 (Appendix A), 5-round PoK 1 (Appendix D), 3-round PoK 2 (Appendix G) and 3-round PoK 3 (Appendix J). Hereafter, we assume that these protocols provide a negligible soundness error which in practice implies to perform a parallel repetition of the PoK. By applying the results from Sect. 6, we obtain the optimized versions of our 3-round PoK 1 (Appendix B), 5-round PoK 1 (Appendix E), 3-round PoK 2 (Appendix H) and 3-round PoK 3 (Appendix K). The optimized versions of our PoK include protocol repetitions hence achieve a negligible soundness error. Then, using the Fiat–Shamir transformation presented in Figs. 7 and 8, we construct four signatures denoted Sig 1 (3-round), Sig 1 (5-round), Sig 2 (3-round) and Sig 3 (3-round) that can be found in Appendices C, F, I and L respectively. Similarly to the signatures constructed in [13] and [26], our security theorems apply to the non-optimized versions of our protocols while the optimized versions are the ones considered in practice.

$\underline{\mathsf{Keygen}(x, w)}$

$\mathsf{pk} = x,\ \mathsf{sk} = w$

return $(\mathsf{pk}, \mathsf{sk})$

$\underline{\mathsf{Sign}(\mathsf{pk}, \mathsf{sk}, m)}$

for $k \in [1, M]$ **do**

$\quad \theta^{(k)} \xleftarrow{\$} \{0, 1\}^{\lambda}$

$\quad \mathsf{aux}^{(k)} \longleftarrow \mathsf{Setup}(\theta^{(k)})$

$\quad (\mathsf{com}^{(k)},\ ^{(k)}) \longleftarrow P_1(\mathsf{pk},\ \mathsf{sk},\ \theta^{(k)})$

end

$\mathsf{com} = (\mathsf{aux}^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M]}$

$(\kappa,\ \alpha) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, \mathsf{com})$

$\mathsf{rsp}' = P_2(\mathsf{pk},\ \mathsf{sk},\ \theta^{(\kappa)},\ ^{(\kappa)},\ \alpha)$

$\mathsf{rsp} = ((\theta^{(k)})_{k \in [1,M] \setminus \kappa},\ \mathsf{rsp}')$

return $\sigma = (\mathsf{com}, \mathsf{rsp})$

$\underline{\mathsf{Verify}(\mathsf{pk}, \sigma, m)}$

$(\kappa,\ \alpha) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, \mathsf{com})$

for $k \in [1, M] \setminus \kappa$ **do**

$\quad$ Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

$\quad$ **if** $(\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})$ **return** 0

end

return $V(\mathsf{pk},\ \mathsf{aux}^{(\kappa)},\ \mathsf{com}^{(\kappa)},\ \alpha,\ \mathsf{rsp}')$

Fig. 7 Signature from Fiat–Shamir Heuristic applied to Fig. 5

Signatures built from the Fiat–Shamir heuristic have been proven existentially unforgeable in the ROM whenever the underlying HVZK PoK achieves a negligible soundness error, see [1, 36, 37]. These security guarantees can be extended to the QROM model following the work of [21, 22, 32, 33, 41–43, 45]. Similarly to the signatures constructed in [13] and [26], our security theorems apply to the non-optimized versions of our protocols while the optimized versions are the ones considered in practice. On a high level, this line of research has shown that the (multi-round) Fiat–Shamir heuristic preserves the soundness and the other proof of knowledge properties in the QROM setting and that transforming such protocols into signature schemes provides (strong) existential unforgeability guarantees. Before stating the security properties of the schemes proposed in this work, we define the "computationally unique responses" property which is required for the proof.

Definition 13 (Computationally Unique Responses) A $(2n + 1)$ round public-coin interactive PoK is said to have *computationally unique responses* if given a partial transcript $(\mathsf{com},\ \mathsf{ch}_1,\ \mathsf{rsp}_1,\ \ldots,\ \mathsf{ch}_i)$ it is computationally infeasible to find two accepting conversations which share the first $(2i)$ messages as above but differ in at least one position. That

$\underline{\mathsf{Keygen}(x, w)}$

$\mathsf{pk} = x,\ \mathsf{sk} = w$

$\textbf{return}\ (\mathsf{pk}, \mathsf{sk})$

$\underline{\mathsf{Sign}(\mathsf{pk}, \mathsf{sk}, m)}$

$\textbf{for}\ k \in [1, M]\ \textbf{do}$

$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

$\quad \mathsf{aux}^{(k)} \longleftarrow \mathsf{Setup}(\theta^{(k)})$

$\textbf{end}$

$\mathsf{com}_1 = (\mathsf{aux}^{(k)})_{k\in[1,M]}$

$\kappa \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, \mathsf{com}_1)$

$(\mathsf{com}^{(\kappa)},\ ^{(\kappa)}) \longleftarrow P_1(\mathsf{pk},\ \mathsf{sk},\ \theta^{(\kappa)})$

$\mathsf{com}_2 = \mathsf{com}^{(\kappa)}$

$\alpha \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, \mathsf{com}_1 \,||\, \mathsf{com}_2)$

$\mathsf{rsp}' = P_2(\mathsf{pk},\ \mathsf{sk},\ \theta^{(\kappa)},\ ^{(\kappa)},\ \alpha)$

$\mathsf{rsp} = ((\theta^{(k)})_{k\in[1,M]\setminus\kappa},\ \mathsf{rsp}')$

$\textbf{return}\ \sigma = (\mathsf{com}_1,\ \mathsf{com}_2,\ \mathsf{rsp})$

$\underline{\mathsf{Verify}(\mathsf{pk}, \sigma, m)}$

$\kappa \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, \mathsf{com}_1)$

$\alpha \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, \mathsf{com}_1 \,||\, \mathsf{com}_2)$

$\textbf{for}\ k \in [1, M] \setminus \kappa\ \textbf{do}$

$\quad \text{Compute } \bar{\mathsf{aux}}^{(k)} \text{ from } \theta^{(k)}$

$\quad \textbf{if}\ (\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})\ \textbf{return}\ 0$

$\textbf{end}$

$\textbf{return}\ V(\mathsf{pk},\ \mathsf{aux}^{(\kappa)},\ \mathsf{com}^{(\kappa)},\ \alpha,\ \mathsf{rsp}')$

Fig. 8 Signature from Fiat–Shamir Heuristic applied to Fig. 6

is,

$$\Pr\left[V(\mathsf{trans}_1) = \mathsf{accept} \bigwedge V(\mathsf{trans}_2) = \mathsf{accept} \,\middle|\, (\mathsf{trans}_1, \mathsf{trans}_2) \leftarrow \mathcal{A}\right]$$

is negligible for computationally bounded (quantum) adversary $\mathcal{A}$, where, $\mathsf{trans}_b = (\mathsf{com},\ \mathsf{ch}_1,\ \mathsf{rsp}_1,\ \ldots,\ \mathsf{ch}_i,\ \mathsf{rsp}_i^b, \mathsf{ch}_{i+1}^b,\ \mathsf{rsp}_{i+1}^b,\ \ldots,\ \mathsf{ch}_n^b,\ \mathsf{rsp}_n^b,\ )$ for $b \in \{0, 1\}$ such that $\mathsf{rsp}_i^0 \neq \mathsf{rsp}_i^1$.

Theorem 6 *If the non-optimized variants of Sig 1 (3-round), Sig 1 (5-round), Sig 2 and Sig 3 signature schemes are instantiated with a collapsing hash function as commitment scheme, then the signature schemes are strong existential unforgeable under chosen message attack (sUF-CMA) in the QROM.*

Proof The proof of Theorem 6 is similar to those analyzing the security of (multi-round) Fiat–Shamir transformation of PoK in the QROM [21, 22]. Here, we note that the first message is a commitment generated using a collapsing hash function and hence is unpredictable. Also, the second message in case of Sig 1 (5-round) is computed as a function of the committed values in first message. Similar is the case for the final responses, which additionally includes some opening information. Due to the binding property of the commitment, the second message and the response for each of the schemes is computationally unique. As shown earlier, the schemes are also HVZK. This suffices to prove the strong existential unforgeable under chosen message attack (sUF-CMA) property of the schemes following Theorems 23, 28, Corollarys 24, 29, and 30 from [21].

8 Parameters and comparison

8.1 Parameters choice

The system parameters (n, k, w, M, N, τ) are chosen such that all known attacks cost more than 2^λ elementary operations for a given security parameter λ. The parameters (n, k, w) are related to the difficulty of solving the underlying decoding problems while (M, N, τ) are related to the soundness of the PoK. Resulting parameters are given in Table 3.

Decoding attack We consider decoding problems instantiated with binary $[n, k]$ codes and secrets of small weight w. Parameters are chosen according to the BJMM generic attack [8] along with estimates from [27]. For the PoK leveraging quasi-cyclicity, we take into account the DOOM attack from [39] which reduces the complexity by a factor $\sqrt{n}$.

Soundness error When considering the *beating parallel repetition* optimization (see Sect. 6), one has to choose (M, N, τ) such that the soundness error $\max_{0 \le e \le \tau} \binom{M-e}{\tau-e} \cdot \binom{M}{\tau}^{-1} \cdot N^{-(\tau-e)}$ is negligible with respect to λ. In practice, this offer a trade-off between performances and signature sizes as one can increase M and N (degrading running time) in order to reduce τ (improving signature size). We illustrate this trade-off by providing several parameter sets in Table 3.

Attack against 5-round protocols An attack exploiting the structure of 5-round PoK has been identified in [30]. The main idea is to split the attacker work in two phases: (i) initially it tries to guess the first challenge for several repetitions and then (ii) to guess the second challenge for the remaining repetitions. Our schemes feature the *capability for early abort* described in [30] therefore the cost of the attack is equal to $|\mathcal{C}_1|^{\tau^*} + |\mathcal{C}_2|^{\tau-\tau^*}$ where $\mathcal{C}_1$ and $\mathcal{C}_2$ are the challenge spaces and $\tau^* \in [0, \tau]$ is chosen by the adversary to minimize the attack's cost.

8.2 Resulting key and signature sizes

We now explain how to compute the sizes of the signatures Sig 1 (3-round), Sig 1 (5-round), Sig 2 and Sig 3 (see Appendices C, F, I and L). Hereafter, we consider commitments instantiated from hash functions such that $|\mathsf{com}| = 2\lambda$ bits. Resulting average sizes are given in Table 3. In practice, the size of these signatures vary depending on the challenges received.

Key pair The key pair of the signature built from PoK 1 is defined by $\mathsf{sk} = (\mathbf{x})$ and $\mathsf{pk} = (\mathbf{H}, \mathbf{y}^\top = \mathbf{H}\mathbf{x}^\top)$ while the key pair of the signature built from PoK 2 and PoK 3 is given by $\mathsf{sk} = (\mathbf{x}, \mathbf{e})$ and $\mathsf{pk} = (\mathbf{G}, \mathbf{y} = \mathbf{x}\mathbf{G} + \mathbf{e})$. Both the secret values ($\mathbf{x}$ and $(\mathbf{x}, \mathbf{e})$) and the public

Table 3 Parameters and sizes for Sig 1, Sig 2 and Sig 3 ($\lambda = 128$)

	n	k	w	M	N	τ	pk	σ
Sig 1 (3-round) [QCSD]	1238	619	137	256	2	128	0.1 kB	25.2 kB
Sig 1 (3-round) [SD]	1190	595	132				0.1 kB	24.6 kB
Sig 1 (5-round) [QCSD]	1238	619	137	256	2	143	0.1 kB	24.6 kB
Sig 1 (5-round) [SD]	1190	595	132				0.1 kB	24.0 kB
Sig 2 (3-round) [QCGD]	1238	619	137	272	16	35	0.2 kB	22.6 kB
				389	32	28		20.8 kB
				631	64	23		19.3 kB
Sig 3 (3-round) [QCGD]	1238	619	137	272	16	35	0.2 kB	19.3 kB
				389	32	28		17.2 kB
				631	64	23		15.6 kB

matrices ($\mathbf{H}$ and $\mathbf{G}$) can be generated from seeds. Therefore, sk has size λ bits in both cases while pk is $(n-k)+\lambda$ bits long for PoK 1 and $n+\lambda$ bits long for PoK 2 and PoK 3.

Signature from PoK 1 In our Sig 1 (3-round), the signer has to send $(h, \xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]\setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa\in K})$. In the 5-round variant, the signer send $(h, h', \xi, (\theta^{(k)})_{k\in[1,M]\setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa\in K})$. Both h and h' are commitments of size $|\mathsf{com}|$ bits while ξ is a λ bits long seed. The $M-\tau$ values $\theta^{(k)}$ are the seeds corresponding to the setups that are checked by the verifier without being executed. They can be sent using the aforementioned Merkle tree based optimization (see Sect. 6). In the case of our Sig 1 parameters where $\tau = M/2$, the cost associated with the $\theta^{(k)}$ seeds is equal to $3/4\cdot(M-\tau)\cdot\lambda$ bits (following the same argument as the one used for the seed compression in the response optimization). The $(M-\tau)$ commitments $\mathsf{com}^{(k)}$ used in the 3-round variant can be sent in a similar way using $3/4\cdot(M-\tau)\cdot|\mathsf{com}|$ bits. The response $\mathsf{rsp}^{(\kappa)}$ differs with respect to the value of the challenge α. It either contains $(\phi^{(\kappa)}, \mathbf{u}^{(\kappa)}+\mathbf{x}, \mathsf{com}_1^{(\kappa)})$ when $\alpha = 0$ or $(\psi^{(\kappa)}, \pi^{(\kappa)}[\mathbf{x}], \mathsf{com}_0^{(\kappa)})$ when $\alpha = 1$. The values $\phi^{(\kappa)}$ and $\psi^{(\kappa)}$ are seeds, the value $\mathbf{u}^{(\kappa)}+\mathbf{x}$ is a vector of size n and $\pi^{(\kappa)}[\mathbf{x}]$ can be sent using $n/2$ bits thanks to the small weight vector compression optimization. The cost of sending $\mathsf{com}_0^{(\kappa)}$ and $\mathsf{com}_1^{(\kappa)}$ can be reduced from $2|\mathsf{com}|$ to $7/8\cdot 2|\mathsf{com}|$ using binary trees of length 1 although this is less efficient than for $(\mathsf{com}^{(k)})_{k\in[1,M]\setminus K}$ as one has to take into account the fact that some instances are not going to be executed. Thus, the cost associated with the τ responses $\mathsf{rsp}^{(\kappa)}$ used in the PoK 1 is equal to $\tau/2\cdot(3n/2+2\lambda+7/8\cdot 2|\mathsf{com}|)$. Overall, the signature constructed from 3-round PoK 1 has a size equal to $(1+3/4\cdot(M-\tau))\cdot(\lambda+|\mathsf{com}|)+\tau/2\cdot(3n/2+2\lambda+7/8\cdot 2|\mathsf{com}|)\approx \tau\cdot(0.75n+5\lambda)$ bits. Similarly, the signature constructed from 5-round PoK 1 has a size equal to $(2|\mathsf{com}|+\lambda)+3/4\cdot(M-\tau)\cdot\lambda+\tau/2\cdot(3n/2+2\lambda+7/8\cdot 2|\mathsf{com}|)\approx\tau\cdot(0.75n+3.5\lambda)$ bits. We have provided parameters suitable for the SD problem and its quasi-cyclic variant QCSD in Table 3.

Signature from PoK 2 In our Sig 2, the signer has to send $(h, \xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]\setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa\in K})$. The value h is commitment of size $|\mathsf{com}|$ bits while ξ is a λ bits long seed. The $(M-\tau)$ values $\theta^{(k)}$ and $\mathsf{com}^{(k)}$ are respectively the seeds and commitments corresponding to the setups that are checked by the verifier without being executed. By leveraging the Merkle tree based optimization, their cost is respectively equal to $\tau\log_2(M/\tau)\cdot\lambda$ bits and $\tau\log_2(M/\tau)\cdot|\mathsf{com}|$ bits. The prover's response contains $(\mathbf{u}_\alpha^{(\kappa)}+\mathbf{x}, \pi_\alpha^{(\kappa)}[\mathbf{e}], \mathbf{s}_\alpha^{(\kappa)}, \theta_{\alpha*}^{(\kappa)}, \mathsf{aux}_{\alpha*}^{(\kappa)})$.

Table 4 Comparison between code-based signatures built from PoK ($\lambda = 128$)

	Performance			Size		Security Assumption
	μ	ν	Cost	pk	σ	
Stern	219	2	438	0.1 kB	37.6 kB	SD (or QCSD) over $\mathbb{F}_2$
Véron	219	2	438	0.2 kB	31.2 kB	SD (or QCSD) over $\mathbb{F}_2$
CVE	156	2	312	0.2 kB	32.6 KB	SD (or QCSD) over $\mathbb{F}_q$
AGS	151	2	302	0.2 kB	30.5 kB	QCSD/DiffSD over $\mathbb{F}_2$
	141	2	282	3.1 kB	28.5 kB	
BGKS	151	2	302	0.1 kB	24.1 kB	QCSD/DiffSD over $\mathbb{F}_2$
	141	2	282	1.7 kB	22.5 kB	
GPS	512	128	65 536	0.1 kB	24.6 kB	SD (or QCSD) over $\mathbb{F}_q$
	1024	256	262 144	0.1 kB	22.2 kB	
	2048	512	1 048 576	0.1 kB	20.2 kB	
	4096	1024	4 194 304	0.1 kB	19.5 kB	
FJR	187	8	1496	0.1 kB	24.4 kB	SD (or QCSD) over $\mathbb{F}_2$
	389	32	12 448	0.1 kB	17.6 kB	
Sig 1 (3-round)	256	2	512	0.1 kB	25.2 kB	QCSD over $\mathbb{F}_2$
	256	2	512	0.1 kB	24.6 kB	SD over $\mathbb{F}_2$
Sig 1 (5-round)	256	2	512	0.1 kB	24.6 kB	QCSD over $\mathbb{F}_2$
	256	2	512	0.1 kB	24.0 kB	SD over $\mathbb{F}_2$
Sig 2	272	16	4352	0.2 kB	22.6 kB	QCSD (or SD) over $\mathbb{F}_2$
	389	32	12 448	0.2 kB	20.8 kB	
	631	64	40 384	0.2 kB	19.3 kB	
Sig 3	272	16	4352	0.2 kB	19.3 kB	QCSD (or SD) over $\mathbb{F}_2$
	389	32	12 448	0.2 kB	17.2 kB	
	631	64	40 384	0.2 kB	15.6 kB	

The vectors $\mathbf{u}_\alpha^{(\kappa)} + \mathbf{x}$ and $\mathbf{s}_\alpha^{(\kappa)}$ are of size $k = n/2$ and n respectively. In addition, $\pi_\alpha^{(\kappa)}[\mathbf{e}]$ can be sent using $n/2$ bits thanks to the small weight vector compression optimization. The values $\theta_{\alpha^*}^{(\kappa)}$ and $\mathsf{aux}_{\alpha^*}^{(\kappa)}$ contains respectively $N-1$ seeds and commitments that can be sent using $\lambda \cdot \log_2(N)$ bits and $|\mathsf{com}| \cdot \log_2(N)$ bits. Hence the cost associated with the τ responses is equal to $\tau \cdot (2n + (\lambda + |\mathsf{com}|) \cdot \log_2(N))$ bits. Overall, the signature constructed from 3-round PoK 2 has a size equal to $(|\mathsf{com}| + \lambda) \cdot (1 + \tau \log_2(M/\tau)) + \tau \cdot (2n + (\lambda + |\mathsf{com}|) \cdot \log_2(N)) \approx \tau \cdot (2n + 3\lambda \cdot [\log_2(M/\tau) + \log_2(N)])$ bits.

Signature from PoK 3 The case of Sig 3 is very similar to the one of Sig 2 except that it benefits more from the commitment compression optimization as explained in Sect. 6. As a result, the cost associated with the τ responses is equal to $\tau \cdot (2n + \lambda \cdot \log_2(N) + |\mathsf{com}|)$ instead of $\tau \cdot (2n + (\lambda + |\mathsf{com}|) \cdot \log_2(N))$. Hence, the signature constructed from PoK 3 is of size $(|\mathsf{com}| + \lambda) \cdot (1 + \tau \log_2(M/\tau)) + \tau \cdot (2n + \lambda \cdot \log_2(N) + |\mathsf{com}|) \approx \tau \cdot (2n + \lambda \cdot [3 \log_2(M/\tau) + \log_2(N) + 2])$ bits.

8.3 Comparison to other code-based signatures

We first compare our new signatures to existing code-based signatures based on proofs on knowledge for the syndrome decoding. Next, we extend this comparison to any code-based signatures.

Comparison between code-based signatures using PoK We compare the Stern [40], Véron [12, 44], AGS [2, 12], BGKS [15], CVE [16], GPS [26] and FJR [23] schemes to our new proposals according to their security, sizes and performances. To provide a meaningful comparison, we have updated the parameters of the old schemes so that they achieve a security level $\lambda = 128$ and have applied recent optimizations to the old schemes whenever relevant. We have used the parameters $n = 1190$, $k = 595$ and $w = 132$ for the schemes based on the SD problem (without quasi-cyclicity), $n = 1238$, $k = 619$ and $w = 137$ for the schemes based on the QCSD problem and $n = 226$, $q = 256$, $k = 113$ and $w = 86$ for the schemes based on the SD problem over $\mathbb{F}_q$. For recent GPS and FJR proposals, we have used the parameters proposed by their respective authors in [26] and [23]. As the parameters used in [23] differ from the ones used here, this induces small differences between FJR and Sig 3 although both schemes can achieve similar results. As several of these schemes can be based either on the SD or QCSD problem, we have indicated the case considered in the comparison (which matches the initial design of the scheme) and have indicated the other possible case using parenthesis whenever relevant. Since there are no implementations available for most of these schemes yet (to the best of our knowledge), we provide an estimate of their relative performances. For all these schemes, the first step (every operations executed by the prover before it outputs its first commitment) can be seen as repeating μ times the computation of ν operations whose cost is arbitrarily denoted as one `cost_unit`. This first step hence costs $\mu \cdot \nu \cdot$ `cost_unit`. Using our PoK 2 for illustrative purposes, one can see that $\mu = M$, $\nu = N$ and the `cost_unit` encompasses all the operations required to compute $\mathsf{com}_i^{(k)}$ for a given $k \in [1, M]$ and $i \in [1, N]$. We believe this constitutes a good estimate of the relative performances between these schemes as this step will likely dominate their overall cost. Indeed, the `cost_unit` generally contains the most costly operations (matrix / vector multiplication, randomness sampling and hash computation) and is repeated $\mu \cdot \nu$ time in order to obtain a negligible soundness error. We define our cost estimate as $\mu \cdot \nu$ thus assuming that the `cost_unit` is similar for each scheme. This introduces an approximation in our comparison which could only be solved by providing and benchmarking actual implementations of the aforementioned schemes. In particular, this approximation hides the performance difference between using plain matrices and structured ones which is not negligible in practice. As such, one should compare the schemes whose `cost_unit` includes a matrix / vector multiplication (the ones based on the plain SD problem) separately from the schemes whose `cost_unit` features an efficient one thanks to structured matrices (the ones based on the QCSD problem). Moreover, the GPS scheme does not include such a multiplication hence its real `cost_unit` is likely to be smaller than the one of other schemes which means that the proposed estimate might overestimate its real cost. Results are displayed in Table 4. One can see that our new constructions provide various trade-offs between security assumptions, performances and sizes for code-based signatures built from PoK. In particular, Sig 1 brings improvement with respect to Stern, Véron, AGS and BGKS protocols at the cost of a very small overhead. If one is willing to accept a greater performance

Table 5 Comparison between code-based signatures ($\lambda = 128$)

	pk	σ	pk + σ	Security Assumption
Wave	3.2 MB	0.93 kB	3.3 MB	Syndrome decoding over $\mathbb{F}_3$ (large weight)
				Generalized $(U, U+V)$-codes indistinguishability
LESS	9.8 kB	15.2 kB	25.0 kB	Linear Code Equivalence
	206 kB	5.3 kB	212 kB	Permutation Code Equivalence
	11.6 kB	10.4 kB	22.0 kB	
Durandal	15.3 kB	4.1 kB	19.4 kB	Rank syndrome decoding over $\mathbb{F}_{2^m}$ (small weight)
	18.6 kB	5.1 kB	23.7 kB	Product spaces subspaces indistinguishability
Sig 1	0.1 kB	24.6 kB	24.7 kB	Syndrome decoding over $\mathbb{F}_2$ (small weight)
	0.1 kB	24.0 kB	24.1 kB	
Sig 2	0.2 kB	22.6 kB	22.8 kB	Syndrome decoding over $\mathbb{F}_2$ (small weight)
	0.2 kB	20.8 kB	21.0 kB	
	0.2 kB	19.3 kB	19.5 kB	
Sig 3	0.2 kB	19.3 kB	19.5 kB	Syndrome decoding over $\mathbb{F}_2$ (small weight)
	0.2 kB	17.2 kB	17.4 kB	
	0.2 kB	15.6 kB	15.8 kB	

overhead, then PoK 3 (and PoK 2 to a lesser extent) permits to achieve even smaller signature sizes.

Comparison to other signatures We provide a comparison with existing code-based signatures (including Wave [20], LESS [7, 14] and Durandal [6]) in Table 5. The LESS scheme relies on the code equivalence problem while Wave relies on both the SD problem (with large weight) and the indistinguishability of generalized $(U, U+V)$-codes. We also include Durandal in our comparison even if it is a scheme based on the rank metric. As such, it would be better to compare it with the rank-metric variants of our schemes. Such variants are straightforward and are discussed in Sect. 9.

9 Generalization and variants

In this section, we briefly discuss the generalization of the proposed signatures to additional metrics as well as several possible variants.

Generalization to additional metrics Following the work of [15], we define a Full Domain Linear Isometry (FDLI) set as a set of linear isometries I which has the property that given a random element $\phi \in I$, the image by ϕ of a random word $\mathbf{x}$ of weight ω is a random word $\mathbf{y}$ of weight ω. One can adapt our protocols to other metrics by (i) redefining the weight of vectors and by (ii) replacing the permutation $\pi \in \mathcal{S}_n$ by a random element ϕ of a FDLI set I. For instance, our protocols can be adapted easily to the rank metric setting using the FDLI set for rank metric described in [25] along with the rank weight and vectors over $\mathbb{F}_{q^m}^n$ in place of the Hamming weight and vectors over $\mathbb{F}_2^n$. In this specific case, for a vector $\mathbf{x} \in \mathbb{F}_{q^m}^n$, one can derive a matrix $\mathbf{X} \in \mathbb{F}_q^{m \times n}$ by writing the coordinates of $\mathbf{x}$ using a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$. Doing so, the set of isometries of $\mathbf{x}$ is given by $\mathbf{PXQ}$ where $\mathbf{P} \in \mathsf{GL}_m(\mathbb{F}_q)$ and $\mathbf{Q} \in \mathsf{GL}_n(\mathbb{F}_q)$. This allows to further reduce the size of our signatures as illustrated in Table 6.

Table 6 Parameters and sizes in the rank metric setting ($\lambda = 128$)

	q	m	n	k	w	M	N	τ	pk	σ
Sig 1 (3-round)	2	31	32	16	9	256	2	128	0.1 kB	22.8 kB
Sig 1 (5-round)									0.1 kB	19.7 kB
Sig 2 (3-round)	2	31	32	16	9	230	8	45	0.2 kB	20.8 kB
						207	16	39		19.1 kB
Sig 3 (3-round)	2	31	32	16	9	230	8	45	0.2 kB	17.4 kB
						389	32	28		15.5 kB

Variants with others structured matrices We have explained in Sect. 6 how one can use structured matrices in order to improve the performances of our proofs of knowledge. To this end, we suggest the use of quasi-cyclic matrices as they are commonly used in code-based cryptography. One should note that this performance improvement could be achieved with any set of matrices that benefit from an efficient matrix / vector product. Therefore, our protocols can be adapted to work with other kind of structured matrices such as Toeplitz matrices which is a generalization of quasi-cyclic matrices.

10 Conclusion

In this paper, we highlighted various trade-offs between the computational performance and signature sizes for PoK-based signatures constructed from code-based cryptography. Specifically, we proposed three new schemes and different trade-offs, in terms of performance overhead, signature sizes, and hardness of the underlying computational assumptions. Our first PoK scheme achieves the soundness error of 1/2, while the second and third PoK schemes can achieve the arbitrarily small soundness error of 1/N. We also presented extensive comparisons of our proposals with many existing schemes in order to highlight various trade-offs. In future we aim to provide the implementation of our proposed schemes to facilitate the comparison of the trade-offs delivered by different schemes. We hope that our work contributes in study of this important topic which can expedite the transition to the post-quantum cryptographic solutions in practice, and encourage future research in this direction.

Appendix

A PoK 1 (3-round, without optimization)

See Fig. 9.

Prover$(\mathbf{x}, \mathbf{H}, \mathbf{y})$ **Verifier**$(\mathbf{H}, \mathbf{y})$

for $k \in [1, M]$ **do**

$\theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

$\pi^{(k)} \xleftarrow{\$,\theta^{(k)}} \mathcal{S}_n,\ \mathbf{u}^{(k)} \xleftarrow{\$,\theta^{(k)}} \mathbb{F}_2^n$

$r_0 \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda,\ r_1 \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda$

$\mathsf{com}_0^{(k)} = \mathsf{Com}\big(r_0^{(k)},\ \pi^{(k)} \,||\, \mathbf{H}(\mathbf{u}^{(k)})^\top\big)$

$\mathsf{com}_1^{(k)} = \mathsf{Com}\big(r_1^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)}]\big)$

$\mathsf{aux}^{(k)} = (\mathsf{com}_0^{(k)}, \mathsf{com}_1^{(k)})$

$r^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

$\mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)} + \mathbf{x}]\big)$

end

$\xrightarrow{(\mathsf{aux}^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]}}$

$\kappa \xleftarrow{\$} [1, M]$

$\alpha \xleftarrow{\$} \{0,1\}$

$\xleftarrow{(\kappa, \alpha)}$

if $\alpha = 0$ **then**

$z_1 = \pi^{(\kappa)},\ \mathbf{z}_2 = \mathbf{u}^{(\kappa)} + \mathbf{x}$

$\mathsf{rsp} = (r_0^{(\kappa)},\ r^{(\kappa)},\ z_1,\ \mathbf{z}_2)$

end

if $\alpha = 1$ **then**

$\mathbf{z}_3 = \pi^{(\kappa)}[\mathbf{u}^{(\kappa)}],\ \mathbf{z}_4 = \pi^{(\kappa)}[\mathbf{x}]$

$\mathsf{rsp} = (r_1^{(\kappa)},\ r^{(\kappa)},\ \mathbf{z}_3,\ \mathbf{z}_4)$

end

$\xrightarrow{(\theta^{(k)})_{k\in[1,M]\setminus\kappa},\ \mathsf{rsp}}$

for $k \in [1, M] \setminus \kappa$ **do**

Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

if $(\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})$ **return** 0

end

if $\alpha = 0$ **then**

$b_1 \longleftarrow \big(\mathsf{com}_0^{(\kappa)} = \mathsf{Com}\big(r_0^{(\kappa)},\ z_1 \,||\, \mathbf{H}\mathbf{z}_2^\top - \mathbf{y}^\top\big)\big)$

$b_2 \longleftarrow \big(\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)},\ z_1[\mathbf{z}_2]\big)\big)$

return $b_1 \wedge b_2$

end

if $\alpha = 1$ **then**

$b_1 \longleftarrow \big(\mathsf{com}_1^{(\kappa)} = \mathsf{Com}\big(r_1^{(\kappa)},\ \mathbf{z}_3\big)\big)$

$b_2 \longleftarrow \big(\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)},\ \mathbf{z}_3 + \mathbf{z}_4\big)\big)$

$b_3 \longleftarrow \big(w_H(\mathbf{z}_4) = \omega\big)$

return $b_1 \wedge b_2 \wedge b_3$

end

Fig. 9 3-round HVZK PoK for the SD problem (without optimization)

B PoK 1 (3-round, with optimizations)

See Fig. 10.

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{H}, \mathbf{y})}$		$\underline{\mathsf{Verifier}(\mathbf{H}, \mathbf{y})}$
$\xi \xleftarrow{\$} \{0,1\}^\lambda$		
for $k \in [1, M]$ **do**		
$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda,\ \phi^{(k)} \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda,\ \psi^{(k)} \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda$		
$\quad \pi^{(k)} \xleftarrow{\$, \phi^{(k)}} \mathcal{S}_n,\ \mathbf{v}^{(k)} \xleftarrow{\$, \psi^{(k)}} \mathbb{F}_2^n,\ \mathbf{u}^{(k)} = (\pi^{(k)})^{-1}[\mathbf{v}^{(k)}]$		
$\quad r_0^{(k)} \xleftarrow{\$, \phi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_0^{(k)} = \mathsf{Com}\big(r_0^{(k)},\ \pi^{(k)} \mid\mid \mathbf{H}(\mathbf{u}^{(k)})^\top\big)$		
$\quad r_1^{(k)} \xleftarrow{\$, \psi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_1^{(k)} = \mathsf{Com}\big(r_1^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)}]\big)$		
$\quad \mathsf{aux}^{(k)} = (\mathsf{com}_0^{(k)}, \mathsf{com}_1^{(k)})$		
$\quad r^{(k)} \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ \mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)} + \mathbf{x}]\big)$		
end		
$r \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)} \mid\mid \mathsf{com}^{(k)})_{k \in [1,M]}\big)$	$\xrightarrow{\quad h \quad}$	$K \xleftarrow{\$} \{K \subset \{1, \cdots, M\},\ \lvert K \rvert = \tau\}$
	$\xleftarrow{\quad (K, A) \quad}$	$A \xleftarrow{\$} \{0,1\}^\tau$
for $(\kappa, \alpha) \in K \times A$ **do**		
$\quad$ **if** $\alpha = 0$ **then**		
$\qquad z_1^{(\kappa)} = \phi^{(\kappa)},\ \mathbf{z}_2^{(\kappa)} = \mathbf{u}^{(\kappa)} + \mathbf{x}$		
$\qquad \mathsf{rsp}^{(\kappa)} = (z_1^{(\kappa)},\ \mathbf{z}_2^{(\kappa)},\ \mathsf{com}_1^{(\kappa)})$		
$\quad$ **end**		
$\quad$ **if** $\alpha = 1$ **then**		
$\qquad z_3^{(\kappa)} = \psi^{(\kappa)},\ \mathbf{z}_4^{(\kappa)} = \pi^{(\kappa)}[\mathbf{x}]$		
$\qquad \mathsf{rsp}^{(\kappa)} = (z_3^{(\kappa)},\ \mathbf{z}_4^{(\kappa)},\ \mathsf{com}_0^{(\kappa)})$		
$\quad$ **end**		
end		
$\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M] \setminus K},\ (\mathsf{rsp}^{(\kappa)})_{\kappa \in K})$	$\xrightarrow{\quad \mathsf{rsp} \quad}$	Compute $\bar{r}, (\bar{r}^{(\kappa)})_{\kappa \in K}$ from ξ
		for $k \in [1, M] \setminus K$ **do**
		$\quad$ Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$
		end
		for $(\kappa, \alpha) \in K \times A$ **do**
		$\quad$ **if** $\alpha = 0$ **then**
		$\qquad \bar{z}_1^{(\kappa)} \xleftarrow{\$, z_1^{(\kappa)}} \mathcal{S}_n,\ \bar{r}_0^{(\kappa)} \xleftarrow{\$, z_1^{(\kappa)}} \{0,1\}^\lambda$
		$\qquad \mathsf{com}_0^{(\kappa)} = \mathsf{Com}\big(\bar{r}_0^{(\kappa)},\ \bar{z}_1^{(\kappa)} \mid\mid \mathbf{H}(\mathbf{z}_2^{(\kappa)})^\top - \mathbf{y}^\top\big)$
		$\qquad \mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)},\ \bar{z}_1^{(\kappa)}[\mathbf{z}_2^{(\kappa)}]\big)$
		$\quad$ **end**
		$\quad$ **if** $\alpha = 1$ **then**
		$\qquad \bar{\mathbf{z}}_3^{(\kappa)} \xleftarrow{\$, z_3^{(\kappa)}} \mathbb{F}_2^n,\ \bar{r}_1^{(\kappa)} \xleftarrow{\$, z_3^{(\kappa)}} \{0,1\}^\lambda$
		$\qquad \mathsf{com}_1^{(\kappa)} = \mathsf{Com}\big(\bar{r}_1^{(\kappa)},\ \bar{\mathbf{z}}_3\big)$
		$\qquad \mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)},\ \bar{\mathbf{z}}_3 + \mathbf{z}_4\big)$
		$\qquad b_1^{(\kappa)} \longleftarrow \big(w_H(\mathbf{z}_4^{(\kappa)}) = \omega\big)$
		$\quad$ **end**
		end
		$b_1 = \bigwedge_{\kappa \in K} b_1^{(\kappa)}$
		$b_2 \longleftarrow \big(h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)} \mid\mid \mathsf{com}^{(k)})_{k \in [1,M]}\big)\big)$
		return $b_1 \wedge b_2$

Fig. 10 3-round HVZK PoK for the SD problem (with optimizations)

C Sig 1 (3-round)

See Figs. 11 and 12.

$\underline{\mathsf{Keygen}(\lambda)}$

$\rho_1 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{x} \xleftarrow{\$,\rho_1} \mathbb{F}_2^n \text{ such that } w_H(\mathbf{x}) = \omega$

$\rho_2 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{H} \xleftarrow{\$,\rho_2} \mathbb{F}_2^{(n-k)\times n},\ \mathbf{y}^\top = \mathbf{H}\mathbf{x}^\top$

$\mathbf{return}\ (\mathsf{sk}, \mathsf{pk}) = (\rho_1, (\rho_2, \mathbf{y}))$

$\underline{\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}, m)}$

$\xi \xleftarrow{\$} \{0,1\}^\lambda$

$\mathbf{for}\ k \in [1, M]\ \mathbf{do}$

$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda,\ \phi^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda,\ \psi^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda$

$\quad \pi^{(k)} \xleftarrow{\$,\phi^{(k)}} \mathcal{S}_n,\ \mathbf{v}^{(k)} \xleftarrow{\$,\psi^{(k)}} \mathbb{F}_2^n,\ \mathbf{u}^{(k)} = (\pi^{(k)})^{-1}[\mathbf{v}^{(k)}]$

$\quad r_0^{(k)} \xleftarrow{\$,\phi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_0^{(k)} = \mathsf{Com}\big(r_0^{(k)},\ \pi^{(k)} \,||\, \mathbf{H}(\mathbf{u}^{(k)})^\top\big)$

$\quad r_1^{(k)} \xleftarrow{\$,\psi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_1^{(k)} = \mathsf{Com}\big(r_1^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)}]\big)$

$\quad \mathsf{aux}^{(k)} = (\mathsf{com}_0^{(k)}, \mathsf{com}_1^{(k)})$

$\quad r^{(k)} \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ \mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)} + \mathbf{x}]\big)$

$\mathbf{end}$

$r \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k\in[1,M]}\big)$

$(K,\ A) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

$\mathbf{for}\ (\kappa, \alpha) \in K \times A\ \mathbf{do}$

$\quad \mathbf{if}\ \alpha = 0\ \mathbf{then}$

$\quad\quad z_1^{(\kappa)} = \phi^{(\kappa)},\ \mathbf{z}_2^{(\kappa)} = \mathbf{u}^{(\kappa)} + \mathbf{x}$

$\quad\quad \mathsf{rsp}^{(\kappa)} = (z_1^{(\kappa)},\ \mathbf{z}_2^{(\kappa)},\ \mathsf{com}_1^{(\kappa)})$

$\quad \mathbf{end}$

$\quad \mathbf{if}\ \alpha = 1\ \mathbf{then}$

$\quad\quad z_3^{(\kappa)} = \psi^{(\kappa)},\ \mathbf{z}_4^{(\kappa)} = \pi^{(\kappa)}[\mathbf{x}]$

$\quad\quad \mathsf{rsp}^{(\kappa)} = (z_3^{(\kappa)},\ \mathbf{z}_4^{(\kappa)},\ \mathsf{com}_0^{(\kappa)})$

$\quad \mathbf{end}$

$\mathbf{end}$

$\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]\setminus K},\ (\mathsf{rsp}^{(\kappa)})_{\kappa\in K})$

$\mathbf{return}\ \sigma = (h,\ \mathsf{rsp})$

Fig. 11 Keygen and Sign algorithms for Sig 1 (3-round)

$\underline{\mathsf{Verify}(\mathsf{pk}, \sigma, m)}$

Parse σ as $\sigma = (h, \mathsf{rsp})$ and rsp as $\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]\setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa\in K})$

$(K, A) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

Compute $\bar{r}, (\bar{r}^{(\kappa)})_{\kappa\in K}$ from ξ

for $k \in [1, M] \setminus K$ **do**

  Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

end

for $(\kappa, \alpha) \in K \times A$ **do**

  if $\alpha = 0$ **then**

    $\bar{z}_1^{(\kappa)} \overset{\$, z_1^{(\kappa)}}{\longleftarrow} \mathcal{S}_n,\ \bar{r}_0^{(\kappa)} \overset{\$, z_1^{(\kappa)}}{\longleftarrow} \{0,1\}^\lambda$

    $\mathsf{com}_0^{(\kappa)} = \mathsf{Com}\big(\bar{r}_0^{(\kappa)}, \bar{z}_1^{(\kappa)} \,||\, \mathbf{H}(\mathbf{z}_2^{(\kappa)})^\top - \mathbf{y}^\top\big)$

    $\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)}, \bar{z}_1^{(\kappa)}[\mathbf{z}_2^{(\kappa)}]\big)$

  end

  if $\alpha = 1$ **then**

    $\bar{\mathbf{z}}_3^{(\kappa)} \overset{\$, z_3^{(\kappa)}}{\longleftarrow} \mathbb{F}_2^n,\ \bar{r}_1^{(\kappa)} \overset{\$, z_3^{(\kappa)}}{\longleftarrow} \{0,1\}^\lambda$

    $\mathsf{com}_1^{(\kappa)} = \mathsf{Com}\big(\bar{r}_1^{(\kappa)}, \bar{\mathbf{z}}_3\big)$

    $\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)}, \bar{\mathbf{z}}_3 + \mathbf{z}_4\big)$

    $b_1^{(\kappa)} \longleftarrow \big(w_H\big(\mathbf{z}_4^{(\kappa)}\big) = \omega\big)$

  end

end

$b_1 = \bigwedge_{\kappa\in K} b_1^{(\kappa)}$

$b_2 \longleftarrow \big(h = \mathsf{Com}\big(r, (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k\in[1,M]}\big)\big)$

return $b_1 \wedge b_2$

Fig. 12 Verify algorithm for Sig 1 (3-round)

D PoK 1 (5-round, without optimization)

See Fig. 13.

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{H}, \mathbf{y})}$ $\quad\quad$ $\underline{\mathsf{Verifier}(\mathbf{H}, \mathbf{y})}$

for $k \in [1, M]$ **do**

$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

$\quad \pi^{(k)} \xleftarrow{\$, \theta^{(k)}} \mathcal{S}_n, \ \mathbf{u}^{(k)} \xleftarrow{\$, \theta^{(k)}} \mathbb{F}_2^n$

$\quad r_0 \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda, \ r_1 \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda$

$\quad \mathsf{com}_0^{(k)} = \mathsf{Com}\big(r_0^{(k)}, \ \pi^{(k)} \,||\, \mathbf{H}(\mathbf{u}^{(k)})^\top\big)$

$\quad \mathsf{com}_1^{(k)} = \mathsf{Com}\big(r_1^{(k)}, \ \pi^{(k)}[\mathbf{u}^{(k)}]\big)$

$\quad \mathsf{aux}^{(k)} = (\mathsf{com}_0^{(k)}, \mathsf{com}_1^{(k)})$

end

$\xrightarrow{(\mathsf{aux}^{(k)})_{k \in [1,M]}}$

$\xleftarrow{\kappa}$ $\quad\quad$ $\kappa \xleftarrow{\$} [1, M]$

$r^{(\kappa)} \xleftarrow{\$} \{0,1\}^\lambda$

$\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)}, \ \pi^{(\kappa)}[\mathbf{u}^{(\kappa)} + \mathbf{x}]\big)$

$\xrightarrow{(\theta^{(k)})_{k \in [1,M] \setminus \kappa}, \ \mathsf{com}^{(\kappa)}}$

for $k \in [1, M] \setminus \kappa$ **do**

$\quad$ Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

$\quad$ **if** $(\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})$ **return** 0

end

$\xleftarrow{\alpha}$ $\quad\quad$ $\alpha \xleftarrow{\$} \{0,1\}$

if $\alpha = 0$ **then**

$\quad z_1 = \pi^{(\kappa)}, \ \mathbf{z}_2 = \mathbf{u}^{(\kappa)} + \mathbf{x}$

$\quad \mathsf{rsp} = (r_0^{(\kappa)}, r^{(\kappa)}, z_1, \mathbf{z}_2)$

end

if $\alpha = 1$ **then**

$\quad \mathbf{z}_3 = \pi^{(\kappa)}[\mathbf{u}^{(\kappa)}], \ \mathbf{z}_4 = \pi^{(\kappa)}[\mathbf{x}]$

$\quad \mathsf{rsp} = (r_1^{(\kappa)}, r^{(\kappa)}, \mathbf{z}_3, \mathbf{z}_4)$

end

$\xrightarrow{\mathsf{rsp}}$

if $\alpha = 0$ **then**

$\quad b_1 \longleftarrow \big(\mathsf{com}_0^{(\kappa)} = \mathsf{Com}\big(r_0^{(\kappa)}, \ z_1 \,||\, \mathbf{H}\mathbf{z}_2^\top - \mathbf{y}^\top\big)\big)$

$\quad b_2 \longleftarrow \big(\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)}, \ z_1[\mathbf{z}_2]\big)\big)$

$\quad$ **return** $b_1 \wedge b_2$

end

if $\alpha = 1$ **then**

$\quad b_1 \longleftarrow \big(\mathsf{com}_1^{(\kappa)} = \mathsf{Com}\big(r_1^{(\kappa)}, \ \mathbf{z}_3\big)\big)$

$\quad b_2 \longleftarrow \big(\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)}, \ \mathbf{z}_3 + \mathbf{z}_4\big)\big)$

$\quad b_3 \longleftarrow \big(w_H(\mathbf{z}_4) = \omega\big)$

$\quad$ **return** $b_1 \wedge b_2 \wedge b_3$

end

Fig. 13 5-round HVZK PoK for the SD problem (without optimization)

E PoK 1 (5-round, with optimizations)

See Fig. 14.

Prover$(\mathbf{x}, \mathbf{H}, \mathbf{y})$

$\xi \xleftarrow{\$} \{0,1\}^\lambda$

for $k \in [1, M]$ **do**

$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda,\ \phi^{(k)} \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda,\ \psi^{(k)} \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda$

$\quad \pi^{(k)} \xleftarrow{\$, \phi^{(k)}} \mathcal{S}_n,\ \mathbf{v}^{(k)} \xleftarrow{\$, \psi^{(k)}} \mathbb{F}_2^n,\ \mathbf{u}^{(k)} = (\pi^{(k)})^{-1}[\mathbf{v}^{(k)}]$

$\quad r_0^{(k)} \xleftarrow{\$, \phi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_0^{(k)} = \mathsf{Com}\big(r_0^{(k)},\ \pi^{(k)} \,||\, \mathbf{H}(\mathbf{u}^{(k)})^\top\big)$

$\quad r_1^{(k)} \xleftarrow{\$, \psi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_1^{(k)} = \mathsf{Com}\big(r_1^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)}]\big)$

$\quad \mathsf{aux}^{(k)} = (\mathsf{com}_0^{(k)}, \mathsf{com}_1^{(k)})$

end

$r \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)})_{k \in [1,M]}\big)$

$\xrightarrow{\quad h \quad}$

Verifier$(\mathbf{H}, \mathbf{y})$: $K \xleftarrow{\$} \{K \subset \{1, \cdots, M\},\ |K| = \tau\}$

$\xleftarrow{\quad K \quad}$

for $\kappa \in K$ **do**

$\quad r^{(\kappa)} \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ \mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)},\ \pi^{(\kappa)}[\mathbf{u}^{(\kappa)} + \mathbf{x}]\big)$

end

$r' \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ h' = \mathsf{Com}\big(r',\ (\mathsf{com}^{(\kappa)})_{\kappa \in K}\big)$

$\xrightarrow{\quad h' \quad}$

Verifier: $A \xleftarrow{\$} \{0,1\}^\tau$

$\xleftarrow{\quad A \quad}$

for $(\kappa, \alpha) \in K \times A$ **do**

$\quad$ **if** $\alpha = 0$ **then**

$\quad\quad z_1^{(\kappa)} = \phi^{(\kappa)},\ \mathbf{z}_2^{(\kappa)} = \mathbf{u}^{(\kappa)} + \mathbf{x},\ \mathsf{rsp}^{(\kappa)} = (z_1^{(\kappa)}, \mathbf{z}_2^{(\kappa)}, \mathsf{com}_1^{(\kappa)})$

$\quad$ **end**

$\quad$ **if** $\alpha = 1$ **then**

$\quad\quad z_3^{(\kappa)} = \psi^{(\kappa)},\ \mathbf{z}_4^{(\kappa)} = \pi^{(\kappa)}[\mathbf{x}],\ \mathsf{rsp}^{(\kappa)} = (z_3^{(\kappa)}, \mathbf{z}_4^{(\kappa)}, \mathsf{com}_0^{(\kappa)})$

$\quad$ **end**

end

$\mathsf{rsp} = \big(\xi, (\theta^{(k)})_{k \in [1,M] \setminus K},\ (\mathsf{rsp}^{(\kappa)})_{\kappa \in K}\big)$

$\xrightarrow{\quad \mathsf{rsp} \quad}$

Verifier:

Compute $\bar{r}, (\bar{r}^{(\kappa)})_{\kappa \in K}$ and $\bar{r'}$ from ξ

for $k \in [1, M] \setminus K$ **do**

$\quad$ Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

end

for $(\kappa, \alpha) \in K \times A$ **do**

$\quad$ **if** $\alpha = 0$ **then**

$\quad\quad \bar{z}_1^{(\kappa)} \xleftarrow{\$, z_1^{(\kappa)}} \mathcal{S}_n,\ \bar{r}_0^{(\kappa)} \xleftarrow{\$, z_1^{(\kappa)}} \{0,1\}^\lambda$

$\quad\quad \mathsf{com}_0^{(\kappa)} = \mathsf{Com}\big(\bar{r}_0^{(\kappa)},\ \bar{z}_1^{(\kappa)} \,||\, \mathbf{H}(\mathbf{z}_2^{(\kappa)})^\top - \mathbf{y}^\top\big)$

$\quad\quad \mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)},\ \bar{z}_1^{(\kappa)}[\mathbf{z}_2^{(\kappa)}]\big)$

$\quad$ **end**

$\quad$ **if** $\alpha = 1$ **then**

$\quad\quad \bar{\mathbf{z}}_3^{(\kappa)} \xleftarrow{\$, z_3^{(\kappa)}} \mathbb{F}_2^n,\ \bar{r}_1^{(\kappa)} \xleftarrow{\$, z_3^{(\kappa)}} \{0,1\}^\lambda$

$\quad\quad \mathsf{com}_1^{(\kappa)} = \mathsf{Com}\big(\bar{r}_1^{(\kappa)},\ \bar{\mathbf{z}}_3\big)$

$\quad\quad \mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)},\ \bar{\mathbf{z}}_3 + \mathbf{z}_4\big)$

$\quad\quad b_1^{(\kappa)} \longleftarrow \big(w_H(\mathbf{z}_4^{(\kappa)}) = \omega\big)$

$\quad$ **end**

end

$b_1 = \bigwedge_{\kappa \in K} b_1^{(\kappa)}$

$b_2 \longleftarrow \big(h = \mathsf{Com}(\bar{r},\ (\mathsf{aux}^{(k)})_{k \in [1,M]})\big)$

$b_3 \longleftarrow \big(h' = \mathsf{Com}(\bar{r'},\ (\mathsf{com}^{(\kappa)})_{\kappa \in K})\big)$

return $b_1 \wedge b_2 \wedge b_3$

Fig. 14 5-round HVZK PoK for the SD problem (with optimizations)

F Sig 1 (5-round)

See Figs. 15 and 16.

$\underline{\mathsf{Keygen}(\lambda)}$

$\rho_1 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{x} \xleftarrow{\$,\rho_1} \mathbb{F}_2^n$ such that $w_H(\mathbf{x}) = \omega$

$\rho_2 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{H} \xleftarrow{\$,\rho_2} \mathbb{F}_2^{(n-k)\times n},\ \mathbf{y}^\top = \mathbf{H}\mathbf{x}^\top$

return $(\mathsf{sk},\mathsf{pk}) = (\rho_1, (\rho_2, \mathbf{y}))$

$\underline{\mathsf{Sign}(\mathsf{sk},\mathsf{pk},m)}$

$\xi \xleftarrow{\$} \{0,1\}^\lambda$

for $k \in [1, M]$ **do**

$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda,\ \phi^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda,\ \psi^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda$

$\quad \pi^{(k)} \xleftarrow{\$,\phi^{(k)}} \mathcal{S}_n,\ \mathbf{v}^{(k)} \xleftarrow{\$,\psi^{(k)}} \mathbb{F}_2^n,\ \mathbf{u}^{(k)} = (\pi^{(k)})^{-1}[\mathbf{v}^{(k)}]$

$\quad r_0^{(k)} \xleftarrow{\$,\phi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_0^{(k)} = \mathsf{Com}\big(r_0^{(k)},\ \pi^{(k)} \,||\, \mathbf{H}(\mathbf{u}^{(k)})^\top\big)$

$\quad r_1^{(k)} \xleftarrow{\$,\psi^{(k)}} \{0,1\}^\lambda,\ \mathsf{com}_1^{(k)} = \mathsf{Com}\big(r_1^{(k)},\ \pi^{(k)}[\mathbf{u}^{(k)}]\big)$

$\quad \mathsf{aux}^{(k)} = (\mathsf{com}_0^{(k)}, \mathsf{com}_1^{(k)})$

end

$r \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)})_{k\in[1,M]}\big)$

$K \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

for $\kappa \in K$ **do**

$\quad r^{(\kappa)} \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ \mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)},\ \pi^{(\kappa)}[\mathbf{u}^{(\kappa)} + \mathbf{x}]\big)$

end

$r' \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ h' = \mathsf{Com}\big(r',\ (\mathsf{com}^{(\kappa)})_{\kappa\in K}\big)$

$A \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h \,||\, h')$

for $(\kappa, \alpha) \in K \times A$ **do**

$\quad$ **if** $\alpha = 0$ **then**

$\quad\quad z_1^{(\kappa)} = \phi^{(\kappa)},\ \mathbf{z}_2^{(\kappa)} = \mathbf{u}^{(\kappa)} + \mathbf{x},\ \mathsf{rsp}^{(\kappa)} = (z_1^{(\kappa)},\ \mathbf{z}_2^{(\kappa)},\ \mathsf{com}_1^{(\kappa)})$

$\quad$ **end**

$\quad$ **if** $\alpha = 1$ **then**

$\quad\quad z_3^{(\kappa)} = \psi^{(\kappa)},\ \mathbf{z}_4^{(\kappa)} = \pi^{(\kappa)}[\mathbf{x}],\ \mathsf{rsp}^{(\kappa)} = (z_3^{(\kappa)},\ \mathbf{z}_4^{(\kappa)},\ \mathsf{com}_0^{(\kappa)})$

$\quad$ **end**

end

$\mathsf{rsp} = \big(\xi, (\theta^{(k)})_{k\in[1,M]\setminus K},\ (\mathsf{rsp}^{(\kappa)})_{\kappa\in K}\big)$

return $\sigma = (h,\ h',\ \mathsf{rsp})$

Fig. 15 Keygen and Sign algorithms for Sig 1 (5-round)

$\underline{\mathsf{Verify}(\mathsf{pk}, \sigma, m)}$

Parse σ as $\sigma = (h, h', \mathsf{rsp})$ and rsp as $\mathsf{rsp} = \big(\xi, (\theta^{(k)})_{k \in [1,M] \setminus K},\ (\mathsf{rsp}^{(\kappa)})_{\kappa \in K}\big)$

$K \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

$A \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h \,||\, h')$

Compute $\bar{r}, (\bar{r}^{(\kappa)})_{\kappa \in K}$ and $\bar{r'}$ from ξ

for $k \in [1, M] \setminus K$ **do**

  Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

end

for $(\kappa, \alpha) \in K \times A$ **do**

  if $\alpha = 0$ **then**

    $\bar{z}_1^{(\kappa)} \xleftarrow{\$, z_1^{(\kappa)}} \mathcal{S}_n,\ \bar{r}_0^{(\kappa)} \xleftarrow{\$, z_1^{(\kappa)}} \{0,1\}^\lambda$

    $\mathsf{com}_0^{(\kappa)} = \mathsf{Com}\big(\bar{r}_0^{(\kappa)},\ \bar{z}_1^{(\kappa)} \,||\, \mathbf{H}(\mathbf{z}_2^{(\kappa)})^\top - \mathbf{y}^\top\big)$

    $\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)},\ \bar{z}_1^{(\kappa)}[\mathbf{z}_2^{(\kappa)}]\big)$

  end

  if $\alpha = 1$ **then**

    $\bar{\mathbf{z}}_3^{(\kappa)} \xleftarrow{\$, z_3^{(\kappa)}} \mathbb{F}_2^n,\ \bar{r}_1^{(\kappa)} \xleftarrow{\$, z_3^{(\kappa)}} \{0,1\}^\lambda$

    $\mathsf{com}_1^{(\kappa)} = \mathsf{Com}\big(\bar{r}_1^{(\kappa)},\ \bar{\mathbf{z}}_3\big)$

    $\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)},\ \bar{\mathbf{z}}_3 + \mathbf{z}_4\big)$

    $b_1^{(\kappa)} \longleftarrow \big(w_H\big(\mathbf{z}_4^{(\kappa)}\big) = \omega\big)$

  end

end

$b_1 = \bigwedge_{\kappa \in K} b_1^{(\kappa)}$

$b_2 \longleftarrow \big(h = \mathsf{Com}\big(\bar{r},\ (\mathsf{aux}^{(k)})_{k \in [1,M]}\big)\big)$

$b_3 \longleftarrow \big(h' = \mathsf{Com}\big(\bar{r'},\ (\mathsf{com}^{(\kappa)})_{\kappa \in K}\big)\big)$

return $b_1 \wedge b_2 \wedge b_3$

Fig. 16 Verify algorithm for Sig 1 (5-round)

G PoK 2 (3-round, without optimization)

See Fig. 17.

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{e}, \mathbf{G}, \mathbf{y})}$ $\qquad$ $\underline{\mathsf{Verifier}(\mathbf{G}, \mathbf{y})}$

for $k \in [1, M]$ **do**
$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$
$\quad$ **for** $i \in [1, N]$ **do**
$\quad\quad \theta_i^{(k)} \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda, \ \pi_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathcal{S}_n$
$\quad\quad \mathbf{u}_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathbb{F}_2^k, \ \mathbf{v}_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathbb{F}_2^n$
$\quad$ **end**
$\quad \mathbf{u}^{(k)} = \sum_{i \in [1,N]} \mathbf{u}_i^{(k)}$
$\quad$ **for** $i \in [1, N]$ **do**
$\quad\quad r_i^{(k)} \xleftarrow{\$, \theta_i} \{0,1\}^\lambda, \ \theta_{i^*}^{(k)} = (\theta_j^{(k)})_{j \in [1,N] \setminus i}$
$\quad\quad \mathsf{com}_i^{(k)} = \mathsf{Com}\big(r_i^{(k)}, \ \pi_i^{(k)}[\mathbf{y} + \mathbf{u}^{(k)}\mathbf{G}] + \mathbf{v}_i^{(k)} \,||\, \theta_{i^*}^{(k)}\big)$
$\quad$ **end**
$\quad \mathsf{aux}^{(k)} = (\mathsf{com}_i^{(k)})_{i \in [1,N]}$

$\quad$ **for** $i \in [1, N]$ **do**
$\quad\quad \mathbf{s}_i^{(k)} = \pi_i^{(k)}[(\mathbf{u}^{(k)} + \mathbf{x})\mathbf{G}] + \mathbf{v}_i^{(k)}$
$\quad$ **end**
$\quad r^{(k)} \xleftarrow{\$} \{0,1\}^\lambda, \ \mathbf{s}^{(k)} = (\mathbf{s}_i^{(k)})_{i \in [1,N]}$
$\quad \mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)}, \ \mathbf{u}^{(k)} + \mathbf{x} \,||\, \mathbf{s}^{(k)}\big)$
end

$\xrightarrow{(\mathsf{aux}^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M]}}$

Verifier: $\kappa \xleftarrow{\$} [1, M]$
$\alpha \xleftarrow{\$} \{0, 1\}$

$\xleftarrow{(\kappa, \alpha)}$

Prover: $\mathbf{z}_1 = \mathbf{u}_\alpha^{(\kappa)} + \mathbf{x}, \ \mathbf{z}_2 = \pi_\alpha^{(\kappa)}[\mathbf{e}], \ \mathbf{z}_3 = \mathbf{s}_\alpha^{(\kappa)}, \ z_4 = \theta_{\alpha^*}^{(\kappa)}$
$\mathsf{rsp} = (r_\alpha^{(\kappa)}, r^{(\kappa)}, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, z_4)$

$\xrightarrow{(\theta^{(k)})_{k \in [1,M] \setminus \kappa}, \ \mathsf{rsp}}$

Verifier:
for $k \in [1, M] \setminus \kappa$ **do**
$\quad$ Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$
$\quad$ **if** $(\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})$ **return** 0
end

Compute $(\bar{\pi}_i^{(\kappa)}, \bar{\mathbf{u}}_i^{(\kappa)}, \bar{\mathbf{v}}_i^{(\kappa)})_{i \in [1,N] \setminus \alpha}$ from z_4
$\bar{\mathbf{z}}_1 = \mathbf{z}_1 + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i^{(\kappa)}$
for $i \in [1, N] \setminus \alpha$ **do**
$\quad \bar{\mathbf{s}}_i^{(\kappa)} = \bar{\pi}_i^{(\kappa)}[\bar{\mathbf{z}}_1^{(\kappa)}\mathbf{G}] + \bar{\mathbf{v}}_i^{(\kappa)}$
end
$\bar{\mathbf{s}}^{(\kappa)} = (\bar{\mathbf{s}}_1^{(\kappa)}, \cdots, \bar{\mathbf{s}}_{\alpha-1}^{(\kappa)}, \mathbf{z}_3, \bar{\mathbf{s}}_{\alpha+1}^{(\kappa)}, \cdots, \bar{\mathbf{s}}_N^{(\kappa)})$

$b_1 \longleftarrow \big(\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)}, \ \bar{\mathbf{z}}_1 \,||\, \bar{\mathbf{s}}^{(\kappa)}\big)\big)$
$b_2 \longleftarrow \big(\mathsf{com}_\alpha^{(\kappa)} = \mathsf{Com}\big(r_\alpha^{(\kappa)}, \ \mathbf{z}_3 + \mathbf{z}_2 \,||\, z_4\big)\big)$
$b_3 \longleftarrow \big(w_H(\mathbf{z}_2) = \omega\big)$
return $b_1 \wedge b_2 \wedge b_3$

Fig. 17 3-round ZK PoK for the GD problem (without optimization)

H PoK 2 (3-round, with optimizations)

See Fig. 18.

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{e}, \mathbf{G}, \mathbf{y})}$

$\xi \xleftarrow{\$} \{0,1\}^\lambda$

for $k \in [1, M]$ **do**

  $\theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

  for $i \in [1, N]$ **do**

    $\theta_i^{(k)} \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda, \ \pi_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathcal{S}_n$

    $\mathbf{u}_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathbb{F}_2^k, \ \mathbf{v}_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathbb{F}_2^n$

  end

  $\mathbf{u}^{(k)} = \sum_{i \in [1,N]} \mathbf{u}_i^{(k)}$

  for $i \in [1, N]$ **do**

    $r_i^{(k)} \xleftarrow{\$, \xi} \{0,1\}^\lambda, \ \theta_{i^*}^{(k)} = (\theta_j^{(k)})_{j \in [1,N] \setminus i}$

    $\mathsf{com}_i^{(k)} = \mathsf{Com}\big(r_i^{(k)}, \ \pi_i^{(k)}[\mathbf{y} + \mathbf{u}^{(k)}\mathbf{G}] + \mathbf{v}_i^{(k)} \,||\, \theta_{i^*}^{(k)}\big)$

    $\mathbf{s}_i^{(k)} = \pi_i^{(k)}[(\mathbf{u}^{(k)} + \mathbf{x})\mathbf{G}] + \mathbf{v}_i^{(k)}$

  end

  $\mathsf{aux}^{(k)} = (\mathsf{com}_i^{(k)})_{i \in [1,N]}$

  $\mathbf{s}^{(k)} = (\mathbf{s}_i^{(k)})_{i \in [1,N]}$

  $r^{(k)} \xleftarrow{\$, \xi} \{0,1\}^\lambda, \ \mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)}, \ \mathbf{u}^{(k)} + \mathbf{x} \,||\, \mathbf{s}^{(k)}\big)$

end

$r \xleftarrow{\$, \xi} \{0,1\}^\lambda, \ h = \mathsf{Com}\big(r, \ (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k \in [1,M]}\big)$

$\xrightarrow{\quad h \quad}$

$\underline{\mathsf{Verifier}(\mathbf{G}, \mathbf{y})}$

$K \xleftarrow{\$} \{K \subset \{1, \cdots, M\}, |K| = \tau\}$

$A \xleftarrow{\$} \{0,1\}^\tau$

$\xleftarrow{\quad (K, A) \quad}$

for $(\kappa, \alpha) \in K \times A$ **do**

  $\mathbf{z}_1^{(\kappa)} = \mathbf{u}_\alpha^{(\kappa)} + \mathbf{x}, \ \mathbf{z}_2^{(\kappa)} = \pi_\alpha^{(\kappa)}[\mathbf{e}], \ \mathbf{z}_3^{(\kappa)} = \mathbf{s}_\alpha^{(\kappa)}$

  $z_4^{(\kappa)} = \theta_{\alpha^*}^{(\kappa)}, \ \mathsf{aux}_{\alpha^*}^{(\kappa)} = (\mathsf{com}_i^{(\kappa)})_{i \in [1,N] \setminus \alpha}$

  $\mathsf{rsp}^{(\kappa)} = (\mathbf{z}_1^{(\kappa)}, \mathbf{z}_2^{(\kappa)}, \mathbf{z}_3^{(\kappa)}, z_4^{(\kappa)}, \mathsf{aux}_{\alpha^*}^{(\kappa)})$

end

$\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M] \setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa \in K})$

$\xrightarrow{\quad \mathsf{rsp} \quad}$

Compute $\bar{r}, (\bar{r}_\alpha^{(\kappa)}, \bar{r}^{(\kappa)})_{(\kappa,\alpha) \in K \times A}$ from ξ

for $k \in [1, M] \setminus K$ **do**

  Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

end

for $(\kappa, \alpha) \in K \times A$ **do**

  Compute $(\bar{\pi}_i^{(\kappa)}, \bar{\mathbf{u}}_i^{(\kappa)}, \bar{\mathbf{v}}_i^{(\kappa)})_{i \in [1,N] \setminus \alpha}$ from $z_4^{(\kappa)}$

  $\bar{\mathbf{z}}_1^{(\kappa)} = \mathbf{z}_1^{(\kappa)} + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i^{(\kappa)}$

  for $i \in [1, N] \setminus \alpha$ **do**

    $\bar{\mathbf{s}}_i^{(\kappa)} = \bar{\pi}_i^{(\kappa)}[\bar{\mathbf{z}}_1^{(\kappa)}\mathbf{G}] + \bar{\mathbf{v}}_i^{(\kappa)}$

  end

  $\bar{\mathbf{s}}^{(\kappa)} = (\bar{\mathbf{s}}_1^{(\kappa)}, \cdots, \bar{\mathbf{s}}_{\alpha-1}^{(\kappa)}, \mathbf{z}_3^{(\kappa)}, \bar{\mathbf{s}}_{\alpha+1}^{(\kappa)}, \cdots, \bar{\mathbf{s}}_N^{(\kappa)})$

  $\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)}, \ \bar{\mathbf{z}}_1^{(\kappa)} \,||\, \bar{\mathbf{s}}^{(\kappa)}\big)$

  $\mathsf{com}_\alpha^{(\kappa)} = \mathsf{Com}\big(\bar{r}_\alpha^{(\kappa)}, \ \mathbf{z}_3^{(\kappa)} + \mathbf{z}_2^{(\kappa)} \,||\, z_4^{(\kappa)}\big)$

  $b_1^{(\kappa)} \longleftarrow \big(w_H(\mathbf{z}_2^{(\kappa)}) = \omega\big)$

end

$b_1 = \bigwedge_{\kappa \in K} b_1^{(\kappa)}$

$b_2 \longleftarrow \big(h = \mathsf{Com}(\bar{r}, \ (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k \in [1,M]})\big)$

return $b_1 \wedge b_2$

Fig. 18 ZK PoK for the GD problem over $\mathbb{F}_2$ (with optimizations)

I Sig 2

See Figs. 19 and 20.

$\underline{\mathsf{Keygen}(\lambda)}$

$\rho_1 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{x} \xleftarrow{\$,\rho_1} \mathbb{F}_2^k,\ \mathbf{e} \xleftarrow{\$,\rho_1} \mathbb{F}_2^n$ such that $w_H(\mathbf{e}) = \omega$

$\rho_2 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{G} \xleftarrow{\$,\rho_2} \mathbb{F}_2^{k\times n},\ \mathbf{y} = \mathbf{x}\mathbf{G} + \mathbf{e}$

return $(\mathsf{sk}, \mathsf{pk}) = (\rho_1, (\rho_2, \mathbf{y}))$

$\underline{\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}, m)}$

$\xi \xleftarrow{\$} \{0,1\}^\lambda$

for $k \in [1, M]$ **do**

  $\theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

  for $i \in [1, N]$ **do**

    $\theta_i^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda,\ \pi_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathcal{S}_n$

    $\mathbf{u}_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathbb{F}_2^k,\ \mathbf{v}_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathbb{F}_2^n$

  end

  $\mathbf{u}^{(k)} = \sum_{i\in[1,N]} \mathbf{u}_i^{(k)}$

  for $i \in [1, N]$ **do**

    $r_i^{(k)} \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ \theta_{i^*}^{(k)} = (\theta_j^{(k)})_{j\in[1,N]\backslash i}$

    $\mathsf{com}_i^{(k)} = \mathsf{Com}\big(r_i^{(k)},\ \pi_i^{(k)}[\mathbf{y} + \mathbf{u}^{(k)}\mathbf{G}] + \mathbf{v}_i^{(k)} \,||\, \theta_{i^*}^{(k)}\big)$

    $\mathbf{s}_i^{(k)} = \pi_i^{(k)}[(\mathbf{u}^{(k)} + \mathbf{x})\mathbf{G}] + \mathbf{v}_i^{(k)}$

  end

  $\mathsf{aux}^{(k)} = (\mathsf{com}_i^{(k)})_{i\in[1,N]}$

  $\mathbf{s}^{(k)} = (\mathbf{s}_i^{(k)})_{i\in[1,N]}$

  $r^{(k)} \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ \mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)},\ \mathbf{u}^{(k)} + \mathbf{x} \,||\, \mathbf{s}^{(k)}\big)$

end

$r \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ h = \mathsf{Com}\big(r,\ (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k\in[1,M]}\big)$

$(K, A) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

for $(\kappa, \alpha) \in K \times A$ **do**

  $\mathbf{z}_1^{(\kappa)} = \mathbf{u}_\alpha^{(\kappa)} + \mathbf{x},\ \mathbf{z}_2^{(\kappa)} = \pi_\alpha^{(\kappa)}[\mathbf{e}],\ \mathbf{z}_3^{(\kappa)} = \mathbf{s}_\alpha^{(\kappa)}$

  $z_4^{(\kappa)} = \theta_{\alpha^*}^{(\kappa)},\ \mathsf{aux}_{\alpha^*}^{(\kappa)} = (\mathsf{com}_i^{(\kappa)})_{i\in[1,N]\backslash\alpha}$

  $\mathsf{rsp}^{(\kappa)} = (\mathbf{z}_1^{(\kappa)}, \mathbf{z}_2^{(\kappa)}, \mathbf{z}_3^{(\kappa)}, z_4^{(\kappa)}, \mathsf{aux}_{\alpha^*}^{(\kappa)})$

end

$\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]\backslash K}, (\mathsf{rsp}^{(\kappa)})_{\kappa\in K})$

return $\sigma = (h, \mathsf{rsp})$

Fig. 19 Keygen and Sign algorithms for Sig 2

$\underline{\mathsf{Verify}(\mathsf{pk}, \sigma, m)}$

Parse σ as $\sigma = (h, \mathsf{rsp})$ and rsp as $\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M] \setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa \in K})$

$(K, A) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

Compute $\bar{r}, (\bar{r}_\alpha^{(\kappa)}, \bar{r}^{(\kappa)})_{(\kappa,\alpha) \in K \times A}$ from ξ

for $k \in [1, M] \setminus K$ **do**

  Compute $\mathsf{a\bar{u}x}^{(k)}$ from $\theta^{(k)}$

end

for $(\kappa, \alpha) \in K \times A$ **do**

  Compute $(\bar{\pi}_i^{(\kappa)}, \bar{\mathbf{u}}_i^{(\kappa)}, \bar{\mathbf{v}}_i^{(\kappa)})_{i \in [1,N] \setminus \alpha}$ from $z_4^{(\kappa)}$

  $\bar{\mathbf{z}}_1^{(\kappa)} = \mathbf{z}_1^{(\kappa)} + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i^{(\kappa)}$

  for $i \in [1, N] \setminus \alpha$ **do**

    $\bar{\mathbf{s}}_i^{(\kappa)} = \bar{\pi}_i^{(\kappa)}[\bar{\mathbf{z}}_1^{(\kappa)} \mathbf{G}] + \bar{\mathbf{v}}_i^{(\kappa)}$

  end

  $\bar{\mathbf{s}}^{(\kappa)} = (\bar{\mathbf{s}}_1^{(\kappa)}, \cdots, \bar{\mathbf{s}}_{\alpha-1}^{(\kappa)}, \mathbf{z}_3^{(\kappa)}, \bar{\mathbf{s}}_{\alpha+1}^{(\kappa)}, \cdots, \bar{\mathbf{s}}_N^{(\kappa)})$

  $\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)}, \bar{\mathbf{z}}_1^{(\kappa)} \,||\, \bar{\mathbf{s}}^{(\kappa)}\big)$

  $\mathsf{com}_\alpha^{(\kappa)} = \mathsf{Com}\big(\bar{r}_\alpha^{(\kappa)}, \mathbf{z}_3^{(\kappa)} + \mathbf{z}_2^{(\kappa)} \,||\, z_4^{(\kappa)}\big)$

  $b_1^{(\kappa)} \longleftarrow \big(w_H\big(\mathbf{z}_2^{(\kappa)}\big) = \omega\big)$

end

$b_1 = \bigwedge_{\kappa \in K} b_1^{(\kappa)}$

$b_2 \longleftarrow \big(h = \mathsf{Com}\big(\bar{r}, (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k \in [1,M]}\big)\big)$

return $b_1 \wedge b_2$

Fig. 20 Verify algorithm for Sig 2

J PoK 3 (3-round, without optimization)

See Fig. 21.

Prover$(\mathbf{x}, \mathbf{e}, \mathbf{G}, \mathbf{y})$

for $k \in [1, M]$ **do**

$\quad\theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

$\quad$**for** $i \in [1, N]$ **do**

$\quad\quad\theta_i^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda,\ \pi_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathcal{S}_n$

$\quad\quad\mathbf{u}_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathbb{F}_2^k,\ \mathbf{v}_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathbb{F}_2^n$

$\quad$**end**

$\quad\pi^{(k)} = \pi_N^{(k)} \circ \cdots \circ \pi_1^{(k)}$

$\quad\mathbf{u}^{(k)} = \sum_{i\in[1,N]} \mathbf{u}_i^{(k)}$

$\quad\mathbf{v}^{(k)} = \mathbf{v}_N^{(k)} + \sum_{i\in[1,N-1]} \pi_N^{(k)} \circ \cdots \circ \pi_{i+1}^{(k)}[\mathbf{v}_i^{(k)}]$

$\quad\mathbf{s}_0^{(k)} = (\mathbf{u}^{(k)} + \mathbf{x})\mathbf{G}$

$\quad$**for** $i \in [1, N]$ **do**

$\quad\quad r_i^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda,\ \theta_{i^*}^{(k)} = (\theta_j^{(k)})_{j\in[1,N]\setminus i}$

$\quad\quad\mathbf{s}_i^{(k)} = \pi_i^{(k)}[\mathbf{s}_{i-1}^{(k)}] + \mathbf{v}_i^{(k)}$

$\quad\quad\mathsf{com}_i^{(k)} = \mathsf{Com}\big(r_i^{(k)},\ \pi^{(k)}[\mathbf{y} + \mathbf{u}^{(k)}\mathbf{G}] + \mathbf{v}^{(k)} \,||\, \theta_{i^*}^{(k)}\big)$

$\quad$**end**

$\quad\mathsf{aux}^{(k)} = (\mathsf{com}_i^{(k)})_{i\in[1,N]}$

$\quad r^{(k)} \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{s}^{(k)} = (\mathbf{s}_i^{(k)})_{i\in[1,N]}$

$\quad\mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)},\ \mathbf{u}^{(k)} + \mathbf{x} \,||\, \mathbf{s}^{(k)}\big)$

end

Prover $\xrightarrow{(\mathsf{aux}^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]}}$ Verifier

Verifier$(\mathbf{G}, \mathbf{y})$

$\kappa \xleftarrow{\$} [1, M]$

$\alpha \xleftarrow{\$} \{0,1\}$

Prover $\xleftarrow{(\kappa, \alpha)}$ Verifier

$\mathbf{z}_1 = \mathbf{u}_\alpha^{(\kappa)} + \mathbf{x},\ \mathbf{z}_2 = \pi^{(\kappa)}[\mathbf{e}]$

$\mathbf{z}_3 = \mathbf{s}_\alpha^{(\kappa)},\ z_4 = \theta_{\alpha^*}^{(\kappa)}$

$\mathsf{rsp} = (r_\alpha^{(\kappa)}, r^{(\kappa)}, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, z_4)$

Prover $\xrightarrow{(\theta^{(k)})_{k\in[1,M]\setminus\kappa},\ \mathsf{rsp}}$ Verifier

for $k \in [1, M] \setminus \kappa$ **do**

$\quad$Compute $\bar{\mathsf{aux}}^{(k)}$ from $\theta^{(k)}$

$\quad$**if** $(\mathsf{aux}^{(k)} \neq \bar{\mathsf{aux}}^{(k)})$ **return** 0

end

Compute $(\bar{\pi}_i^{(\kappa)}, \bar{\mathbf{u}}_i^{(\kappa)}, \bar{\mathbf{v}}_i^{(\kappa)})_{i\in[1,N]\setminus\alpha}$ from z_4

$\bar{\mathbf{z}}_1 = \mathbf{z}_1 + \sum_{i\in[1,N]\setminus\alpha} \bar{\mathbf{u}}_i^{(\kappa)}$

$\bar{\mathbf{s}}_0^{(\kappa)} = \bar{\mathbf{z}}_1\mathbf{G},\ \bar{\mathbf{s}}_\alpha^{(\kappa)} = \mathbf{z}_3$

for $i \in [1, N] \setminus \alpha$ **do**

$\quad\bar{\mathbf{s}}_i^{(\kappa)} = \bar{\pi}_i^{(\kappa)}[\bar{\mathbf{s}}_{i-1}^{(\kappa)}] + \bar{\mathbf{v}}_i^{(\kappa)}$

end

$\bar{\mathbf{s}}^{(\kappa)} = (\bar{\mathbf{s}}_i^{(\kappa)})_{i\in[1,N]}$

$b_1 \longleftarrow \big(\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(r^{(\kappa)},\ \bar{\mathbf{z}}_1 \,||\, \bar{\mathbf{s}}^{(\kappa)}\big)\big)$

$b_2 \longleftarrow \big(\mathsf{com}_\alpha^{(\kappa)} = \mathsf{Com}\big(r_\alpha^{(\kappa)},\ \bar{\mathbf{s}}_N^{(\kappa)} + \mathbf{z}_2 \,||\, z_4\big)\big)$

$b_3 \longleftarrow \big(w_H(\mathbf{z}_2) = \omega\big)$

Fig. 21 3-round HVZK PoK for the GD problem (without optimization)

K PoK 3 (3-round, with optimizations)

See Fig. 22.

$\underline{\mathsf{Prover}(\mathbf{x}, \mathbf{e}, \mathbf{G}, \mathbf{y})}$

$\xi \xleftarrow{\$} \{0,1\}^\lambda$
for $k \in [1, M]$ **do**
  $\theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$
  for $i \in [1, N]$ **do**
    $\theta_i^{(k)} \xleftarrow{\$, \theta^{(k)}} \{0,1\}^\lambda,\ \pi_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathcal{S}_n$
    $\mathbf{u}_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathbb{F}_2^k,\ \mathbf{v}_i^{(k)} \xleftarrow{\$, \theta_i^{(k)}} \mathbb{F}_2^n$
  end
  $\pi^{(k)} = \pi_N^{(k)} \circ \cdots \circ \pi_1^{(k)},\ \mathbf{u}^{(k)} = \sum_{i \in [1,N]} \mathbf{u}_i^{(k)}$
  $\mathbf{v}^{(k)} = \mathbf{v}_N^{(k)} + \sum_{i \in [1,N-1]} \pi_N^{(k)} \circ \cdots \circ \pi_{i+1}^{(k)}[\mathbf{v}_i^{(k)}]$
  $\mathbf{s}_0^{(k)} = (\mathbf{u}^{(k)} + \mathbf{x})\mathbf{G}$
  for $i \in [1, N]$ **do**
    $r_{1,i}^{(k)} \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ \mathsf{com}_{1,i}^{(k)} = \mathsf{Com}\big(r_{1,i}^{(k)}, \theta_i^{(k)}\big)$
    $\mathbf{s}_i^{(k)} = \pi_i^{(k)}[\mathbf{s}_{i-1}^{(k)}] + \mathbf{v}_i^{(k)}$
  end
  $r_2^{(k)} \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ \mathsf{com}_2^{(k)} = \mathsf{Com}\big(r_2^{(k)}, \pi^{(k)}[\mathbf{y} + \mathbf{u}^{(k)}\mathbf{G}] + \mathbf{v}^{(k)}\big)$
  $\mathbf{s}^{(k)} = (\mathbf{s}_i^{(k)})_{i \in [1,N]},\ \mathsf{aux}^{(k)} = (\mathsf{com}_{1,i}^{(k)}, \mathsf{com}_2^{(k)})_{i \in [1,N]}$
  $r^{(k)} \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ \mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)}, \mathbf{u}^{(k)} + \mathbf{x} \,||\, \mathbf{s}^{(k)}\big)$
end
$r \xleftarrow{\$, \xi} \{0,1\}^\lambda,\ h = \mathsf{Com}\big(r, (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k \in [1,M]}\big)$

$\xrightarrow{\quad h \quad}$

$\underline{\mathsf{Verifier}(\mathbf{G}, \mathbf{y})}$

$K \xleftarrow{\$} \{K \subset \{1, \cdots, M\}, |K| = \tau\}$
$A \xleftarrow{\$} \{0,1\}^\tau$

$\xleftarrow{\quad (K, A) \quad}$

for $(\kappa, \alpha) \in K \times A$ **do**
  $\mathbf{z}_1^{(\kappa)} = \mathbf{u}_\alpha^{(\kappa)} + \mathbf{x},\ \mathbf{z}_2^{(\kappa)} = \pi^{(\kappa)}[\mathbf{e}],\ \mathbf{z}_3^{(\kappa)} = \mathbf{s}_\alpha^{(\kappa)}$
  $z_4^{(\kappa)} = \theta_{\alpha^*}^{(\kappa)} = (\theta_j^{(\kappa)})_{j \in [1,N] \setminus \alpha}$
  $\mathsf{rsp}^{(\kappa)} = (\mathbf{z}_1^{(\kappa)}, \mathbf{z}_2^{(\kappa)}, \mathbf{z}_3^{(\kappa)}, z_4^{(\kappa)}, \mathsf{com}_{1,\alpha}^{(\kappa)})$
end
$\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M] \setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa \in K})$

$\xrightarrow{\quad \mathsf{rsp} \quad}$

Compute $\bar{r}, \bar{r}', (\bar{r}_{1,i}^{(\kappa)}, \bar{r}_2^{(\kappa)})_{i \in [1,N]}^{\kappa \in K}$ from ξ
for $k \in [1, M] \setminus K$ **do**
  Compute $\mathsf{aux}^{(k)}$ from $\theta^{(k)}$
end
for $(\kappa, \alpha) \in K \times A$ **do**
  Compute $(\theta_i^{(\kappa)}, \bar{\pi}_i^{(\kappa)}, \bar{\mathbf{u}}_i^{(\kappa)}, \bar{\mathbf{v}}_i^{(\kappa)})_{i \in [1,N] \setminus \alpha}$ from z_4
  $\bar{\mathbf{z}}_1^{(\kappa)} = \mathbf{z}_1^{(\kappa)} + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i^{(\kappa)}$
  $\bar{\mathbf{s}}_0^{(\kappa)} = \bar{\mathbf{z}}_1^{(\kappa)}\mathbf{G},\ \bar{\mathbf{s}}_\alpha^{(\kappa)} = \mathbf{z}_3^{(\kappa)}$
  for $i \in [1, N] \setminus \alpha$ **do**
    $\mathsf{com}_{1,i}^{(\kappa)} = \mathsf{Com}\big(r_{1,i}^{(\kappa)}, \theta_i^{(\kappa)}\big)$
    $\bar{\mathbf{s}}_i^{(\kappa)} = \bar{\pi}_i^{(\kappa)}[\bar{\mathbf{s}}_{i-1}^{(\kappa)}] + \bar{\mathbf{v}}_i^{(\kappa)}$
  end
  $\bar{\mathbf{s}}^{(\kappa)} = (\bar{\mathbf{s}}_i^{(\kappa)})_{i \in [1,N]}$
  $\mathsf{com}^{(\kappa)} = \mathsf{Com}\big(\bar{r}^{(\kappa)}, \bar{\mathbf{z}}_1 \,||\, \bar{\mathbf{s}}^{(\kappa)}\big)$
  $\mathsf{com}_2^{(\kappa)} = \mathsf{Com}\big(\bar{r}_2^{(\kappa)}, \bar{\mathbf{s}}_N^{(\kappa)} + \mathbf{z}_2\big)$
  $b_1^{(\kappa)} \longleftarrow \big(w_H\big(\mathbf{z}_2^{(\kappa)}\big) = \omega\big)$
end
$b_1 = \bigwedge_{\kappa \in K} b_1^{(\kappa)}$
$b_2 \longleftarrow \big(h = \mathsf{Com}\big(\bar{r}, (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k \in [1,M]}\big)\big)$
return $b_1 \wedge b_2$

Fig. 22 3-round HVZK PoK for the GD problem (with optimizations)

L Sig 3

See Figs. 23 and 24.

$\underline{\mathsf{Keygen}(\lambda)}$

$\rho_1 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{x} \xleftarrow{\$,\rho_1} \mathbb{F}_2^k,\ \mathbf{e} \xleftarrow{\$,\rho_1} \mathbb{F}_2^n \text{ such that } w_H(\mathbf{e}) = \omega$

$\rho_2 \xleftarrow{\$} \{0,1\}^\lambda,\ \mathbf{G} \xleftarrow{\$,\rho_2} \mathbb{F}_2^{k\times n},\ \mathbf{y} = \mathbf{x}\mathbf{G} + \mathbf{e}$

$\textbf{return } (\mathsf{sk}, \mathsf{pk}) = (\rho_1, (\rho_2, \mathbf{y}))$

$\underline{\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}, m)}$

$\xi \xleftarrow{\$} \{0,1\}^\lambda$

$\textbf{for } k \in [1, M] \textbf{ do}$

$\quad \theta^{(k)} \xleftarrow{\$} \{0,1\}^\lambda$

$\quad \textbf{for } i \in [1, N] \textbf{ do}$

$\quad\quad \theta_i^{(k)} \xleftarrow{\$,\theta^{(k)}} \{0,1\}^\lambda,\ \pi_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathcal{S}_n$

$\quad\quad \mathbf{u}_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathbb{F}_2^k,\ \mathbf{v}_i^{(k)} \xleftarrow{\$,\theta_i^{(k)}} \mathbb{F}_2^n$

$\quad \textbf{end}$

$\quad \pi^{(k)} = \pi_N^{(k)} \circ \cdots \circ \pi_1^{(k)},\ \mathbf{u}^{(k)} = \sum_{i\in[1,N]} \mathbf{u}_i^{(k)}$

$\quad \mathbf{v}^{(k)} = \mathbf{v}_N^{(k)} + \sum_{i\in[1,N-1]} \pi_N^{(k)} \circ \cdots \circ \pi_{i+1}^{(k)}[\mathbf{v}_i^{(k)}]$

$\quad \mathbf{s}_0^{(k)} = (\mathbf{u}^{(k)} + \mathbf{x})\mathbf{G}$

$\quad \textbf{for } i \in [1, N] \textbf{ do}$

$\quad\quad r_{1,i}^{(k)} \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ \mathsf{com}_{1,i}^{(k)} = \mathsf{Com}\big(r_{1,i}^{(k)}, \theta_i^{(k)}\big)$

$\quad\quad \mathbf{s}_i^{(k)} = \pi_i^{(k)}[\mathbf{s}_{i-1}^{(k)}] + \mathbf{v}_i^{(k)}$

$\quad \textbf{end}$

$\quad r_2^{(k)} \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ \mathsf{com}_2^{(k)} = \mathsf{Com}\big(r_2^{(k)}, \pi^{(k)}[\mathbf{y} + \mathbf{u}^{(k)}\mathbf{G}] + \mathbf{v}^{(k)}\big)$

$\quad \mathbf{s}^{(k)} = (\mathbf{s}_i^{(k)})_{i\in[1,N]},\ \mathsf{aux}^{(k)} = (\mathsf{com}_{1,i}^{(k)}, \mathsf{com}_2^{(k)})_{i\in[1,N]}$

$\quad r^{(k)} \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ \mathsf{com}^{(k)} = \mathsf{Com}\big(r^{(k)}, \mathbf{u}^{(k)} + \mathbf{x} \,||\, \mathbf{s}^{(k)}\big)$

$\textbf{end}$

$r \xleftarrow{\$,\xi} \{0,1\}^\lambda,\ h = \mathsf{Com}\big(r, (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k\in[1,M]}\big)$

$(K, A) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

$\textbf{for } (\kappa, \alpha) \in K \times A \textbf{ do}$

$\quad \mathbf{z}_1^{(\kappa)} = \mathbf{u}_\alpha^{(\kappa)} + \mathbf{x},\ \mathbf{z}_2^{(\kappa)} = \pi^{(\kappa)}[\mathbf{e}],\ \mathbf{z}_3^{(\kappa)} = \mathbf{s}_\alpha^{(\kappa)}$

$\quad z_4^{(\kappa)} = \theta_{\alpha^*}^{(\kappa)} = (\theta_j^{(\kappa)})_{j\in[1,N]\setminus\alpha}$

$\quad \mathsf{rsp}^{(\kappa)} = (\mathbf{z}_1^{(\kappa)}, \mathbf{z}_2^{(\kappa)}, \mathbf{z}_3^{(\kappa)}, z_4^{(\kappa)}, \mathsf{com}_{1,\alpha}^{(\kappa)})$

$\textbf{end}$

$\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k\in[1,M]\setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa\in K})$

$\textbf{return } \sigma = (h, \mathsf{rsp})$

Fig. 23 Keygen and Sign algorithms for Sig 3

$\underline{\mathsf{Verify}(\mathsf{pk}, \sigma, m)}$

Parse σ as $\sigma = (h, \mathsf{rsp})$ and rsp as $\mathsf{rsp} = (\xi, (\theta^{(k)}, \mathsf{com}^{(k)})_{k \in [1,M] \setminus K}, (\mathsf{rsp}^{(\kappa)})_{\kappa \in K})$

$(K, A) \longleftarrow \mathsf{Hash}(m \,||\, \mathsf{pk} \,||\, h)$

Compute $\bar{r}, \bar{r'}, (\bar{r}_{1,i}^{(\kappa)}, \bar{r}_2^{(\kappa)})_{i \in [1,N]}^{\kappa \in K}$ from ξ

for $k \in [1, M] \setminus K$ **do**

  Compute $\mathsf{aux}^{(k)}$ from $\theta^{(k)}$

end

for $(\kappa, \alpha) \in K \times A$ **do**

  Compute $(\theta_i^{(\kappa)}, \bar{\pi}_i^{(\kappa)}, \bar{\mathbf{u}}_i^{(\kappa)}, \bar{\mathbf{v}}_i^{(\kappa)})_{i \in [1,N] \setminus \alpha}$ from z_4

  $\bar{\mathbf{z}}_1^{(\kappa)} = \mathbf{z}_1^{(\kappa)} + \sum_{i \in [1,N] \setminus \alpha} \bar{\mathbf{u}}_i^{(\kappa)}, \; \bar{\mathbf{s}}_0^{(\kappa)} = \bar{\mathbf{z}}_1^{(\kappa)} \mathbf{G}, \; \bar{\mathbf{s}}_\alpha^{(\kappa)} = \mathbf{z}_3^{(\kappa)}$

  for $i \in [1, N] \setminus \alpha$ **do**

    $\mathsf{com}_{1,i}^{(\kappa)} = \mathsf{Com}(r_{1,i}^{(\kappa)}, \theta_i^{(\kappa)})$

    $\bar{\mathbf{s}}_i^{(\kappa)} = \bar{\pi}_i^{(\kappa)}[\bar{\mathbf{s}}_{i-1}^{(\kappa)}] + \bar{\mathbf{v}}_i^{(\kappa)}$

  end

  $\bar{\mathbf{s}}^{(\kappa)} = (\bar{\mathbf{s}}_i^{(\kappa)})_{i \in [1,N]}$

  $\mathsf{com}^{(\kappa)} = \mathsf{Com}(\bar{r}^{(\kappa)}, \bar{\mathbf{z}}_1 \,||\, \bar{\mathbf{s}}^{(\kappa)})$

  $\mathsf{com}_2^{(\kappa)} = \mathsf{Com}(\bar{r}_2^{(\kappa)}, \bar{\mathbf{s}}_N^{(\kappa)} + \mathbf{z}_2)$

  $b_1^{(\kappa)} \longleftarrow (w_H(\mathbf{z}_2^{(\kappa)}) = \omega)$

end

$b_1 = \bigwedge_{\kappa \in K} b_1^{(\kappa)}$

$b_2 \longleftarrow (h = \mathsf{Com}(\bar{r}, (\mathsf{aux}^{(k)} \,||\, \mathsf{com}^{(k)})_{k \in [1,M]}))$

return $b_1 \wedge b_2$

Fig. 24 Verify algorithm for Sig 3

References

1. Abdalla M., An J.H., Bellare M., Namprempre C.: From identification to signatures via the Fiat-Shamir transform: minimizing assumptions for security and forward-security. In: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (2002).
2. Aguilar C., Gaborit P., Schrek J.: A new zero-knowledge code based identification scheme with reduced communication. In: IEEE Information Theory Workshop (2011).
3. Aguilar Melchor C., Aragon N., Barreto P., Bettaieb S., Bidoux L., Blazy O., Deneuville J.-C., Gaborit P., Ghosh S., Gueron S., Güneysu T., Misoczki R., Persichetti E., Sendrier N., Tillich J.-P., Vasseur V., Zémor G.: BIKE: Bit Flipping Key Encapsulation. NIST Post-Quantum Cryptography Standardization Project (Round 3). https://bikesuite.org (2020).
4. Aguilar Melchor C., Aragon N., Bettaieb S., Bidoux L., Blazy O., Bos J., Deneuville J.-C., Dion A., Gaborit P., Lacan J., Persichetti E., Robert J.-M., Véron P., Zémor G.: Hamming Quasi-Cyclic (HQC). NIST Post-Quantum Cryptography Standardization Project (Round 3). https://pqc-hqc.org (2020).
5. Albrecht M.R., Bernstein D.J., Chou T., Cid C., Gilcher J., Lange T., Maram V., von Maurich I., Misoczki R., Niederhagen R., Patterson K.G., Persichetti E., Peters C., Schwabe P., Sendrier N., Szefer J., Tjhai C.J., Tomlinson M., Wang W.: Classic McEliece. NIST Post-Quantum Cryptography Standardization Project (Round 3). https://classic.mceliece.org (2020).

6. Aragon N., Blazy O., Gaborit P., Hauteville A., Zémor G.: Durandal: a rank metric based signature scheme. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (2019).
7. Barenghi A., Biasse J.-F., Persichetti E., Santini P.: LESS-FM: fine-tuning signatures from a code-based cryptographic group action. In: International Workshop on Post-Quantum Cryptography (PQCrypto) (2021).
8. Becker A., Joux A., May A., Meurer A.: Decoding random binary linear codes in $2^{n/20}$: how $1+1=0$ improves information set decoding. In: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (2012).
9. Bellini E., Caullery F., Gaborit P., Manzano M., Mateu V.: Improved Véron identification and signature schemes in the rank metric. In: IEEE International Symposium on Information Theory (ISIT) (2019).
10. Berlekamp E., McEliece R., Van Tilborg H.: On the inherent intractability of certain coding problems. IEEE Trans. Inf. Theory **24**, 3 (1978).
11. Bernhard D., Pereira O., Warinschi B.: How not to prove yourself: pitfalls of the fiat-shamir heuristic and applications to Helios. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT) (2012).
12. Bettaieb S., Bidoux L., Blazy O., Gaborit P.: Zero-knowledge reparation of the Véron and AGS code-based identification schemes. In: IEEE International Symposium on Information Theory (ISIT) (2021).
13. Beullens W.: Sigma protocols for MQ, PKP and SIS, and fishy signature schemes. In: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (2020).
14. Biasse J.-F., Micheli G., Persichetti E., Santini P.: LESS is more: code-based signatures without syndromes. In: International Conference on Cryptology in Africa (AFRICACRYPT) (2020).
15. Bidoux L., Gaborit P., Kulkarni M., Sendrier N.: Quasi-cyclic stern proof of knowledge. In: IEEE International Symposium on Information Theory (ISIT) (2022).
16. Cayrel P.-L., Véron P., El Yousfi Alaoui S.M.: A zero-knowledge identification scheme based on the q-ary syndrome decoding problem. In: International Conference on Selected Areas in Cryptography (SAC) (2011).
17. Chen K.: A new identification algorithm. In: International Conference on Cryptography: Policy and Algorithms (CPA) (1995).
18. Chen L., Jordan S., Liu Y.-K., Moody D., Peralta R., Perlner R., Smith-Tone D.: Report on post-quantum cryptography. In: US Department of Commerce, National Institute of Standards and Technology (2016).
19. Courtois N., Finiasz M., Sendrier N.: How to achieve a McEliece-based digital signature scheme. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT) (2001).
20. Debris-Alazard T., Sendrier N., Tillich J.-P.: Wave: a new family of trapdoor one-way preimage sampleable functions based on codes. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT) (2019).
21. Don J., Fehr S., Majenz C.: The measure-and-reprogram technique 2.0: multi-round Fiat-Shamir and More. In: International Cryptology Conference (CRYPTO) (2020).
22. Don J., Fehr S., Majenz C., Schaffner C.: Security of the Fiat-Shamir transformation in the quantum random-oracle model. In: International Cryptology Conference (CRYPTO) (2019).
23. Feneuil T., Joux A., Rivain M.: Shared Permutation for Syndrome Decoding: New Zero-Knowledge Protocol and Code-Based Signature. Cryptology ePrint Archive, Report 2021/1576 (2021).
24. Fiat A., Shamir A.: How to prove yourself: practical solutions to identification and signature problems. In: International Cryptology Conference (CRYPTO) (1986).
25. Gaborit P., Schrek J., Zémor G.: Full cryptanalysis of the Chen identification protocol. In: International Workshop on Post-Quantum Cryptography (PQCrypto) (2011).
26. Gueron S., Persichetti E., Santini P.: Designing a practical code-based signature scheme from zero-knowledge proofs with trusted setup. Cryptography **6**(1), 5 (2022).
27. Hamdaoui Y., Sendrier N.: A non asymptotic analysis of information set decoding. In: Cryptology ePrint Archive, Report 2013/162 (2013).
28. Ishai Y., Kushilevitz E., Ostrovsky R., Sahai A.: Zero-knowledge from secure multiparty computation. In: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing (STOC) (2007), pp. 21–30.
29. Jain A., Krenn S., Pietrzak K., Tentes A.: Commitments and efficient zero-knowledge proofs from learning parity with noise. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT) (2012).
30. Kales D., Zaverucha G.: An attack on some signature schemes constructed from five-pass identification schemes. In: International Conference on Cryptology and Network Security (CANS) (2020).

31. Katz J., Kolesnikov V., Wang X.: Improved non-interactive zero knowledge with applications to post-quantum signatures. In: Proceedings of the 2018 ACM Conference on Computer and Communications Security (CCS) (2018).
32. Kiltz E., Lyubashevsky V., Schaffner C.: A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (2018).
33. Liu Q., Zhandry M.: Revisiting post-quantum Fiat-Shamir. In: International Cryptology Conference (CRYPTO) (2019).
34. Lyubashevsky V.: Fiat-Shamir with aborts: applications to lattice and factoring-based signatures. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT) (2009).
35. McEliece R.J.: A public-key cryptosystem based on algebraic coding theory. Coding Thv **4244**, 114–116 (1978).
36. Pointcheval D., Stern J.: Security proofs for signature schemes. In: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (1996).
37. Pointcheval D., Stern J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**(3), 361–396 (2000).
38. Schnorr C.-P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991).
39. Sendrier N.: Decoding one out of many. In: International Workshop on Post-Quantum Cryptography (PQCrypto) (2011).
40. Stern J.: A new identification scheme based on syndrome decoding. In: International Cryptology Conference (CRYPTO) (1993).
41. Unruh D.: Quantum proofs of knowledge. In: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (2012).
42. Unruh D.: Computationally binding quantum commitments. In: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT) (2016).
43. Unruh D.: Post-quantum security of Fiat-Shamir. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT) (2017).
44. Véron P.: Improved identification schemes based on error-correcting codes. Appl. Algebra Eng. Commun. Comput. **8**(1), 57–69 (1997).
45. Zhandry M.: How to construct quantum random functions. In: IEEE Symposium on Foundations of Computer Science FOCS (2012).