The Non-Intersecting Codewords Problem and its Application to MPC-in-the-Head Signatures

Pierre Briaud[1,2], Philippe Gaborit[1], Romaric Neveu[1], and Gilles Zémor[3,4]

[1] XLIM, UMR 7252, France
[2] CNRS, France
[3] Institut de Mathématiques de Bordeaux, UMR 5251, France
[4] Institut universitaire de France (IUF), France

Abstract. Since McEliece introduced the first code-based encryption scheme in 1978, most code-based cryptographic constructions have relied on hard problems related to decoding random linear codes (or variants thereof) or code equivalence. More recently, the use of the MPC-in-the-Head paradigm has enabled the construction of a new class of very competitive digital signature schemes relying on such assumptions, including the NIST submissions Mirath, PERK, RYDE, and SDitH, as well as a recent proposal based on the so-called Subfield Bilinear Collision problem by Huth and Joux (Crypto 2024).
In this work, we enrich the portfolio of code-based MPC-in-the-Head signature schemes by introducing a new hard problem to cryptography, referred to as the Non-Intersecting Codewords (NIC) problem. In this problem, one has to find two codewords of a given linear code such that their supports in the Hamming metric do not intersect. After discussing how to generate hard instances and studying several attacks on it, we show that the NIC problem can be used to construct a competitive MPC-in-the-Head signature scheme. Using generic constructions, we obtain smaller signature sizes than SDitH and PERK, attaining a signature size of 2 934 Bytes for NIST security level I.

1 Introduction

The use of the MPC-in-the-Head (MPCitH) paradigm for zero-knowledge protocols was introduced in [43] and has since revolutionized the construction of post-quantum digital signature schemes, enabling compact schemes based on various unstructured problems unrelated to Euclidean lattices. More recently, the efficiency of such constructions has been further improved thanks to generic optimizations of the underlying MPC protocols [5,46,44,57] and the introduction of two new MPCitH frameworks called Threshold-Computation-in-the-Head [38,39] and VOLE-in-the-Head [15].

This way of building signature schemes proved particularly fruitful in the code-based setting, where SDitH [37,4] initiated a long line of improvements. These works either expanded the range of hardness assumptions that can be efficiently used within MPCitH via new polynomial modelings [3,36,15], or improved the state-of-the-art modeling of one of these problems [14,22,18,12]. As

an illustration of the efficiency of these improvements, we note that among the five MPCitH candidates in the second round of the NIST competition for additional signature schemes, four of them use linear codes. In addition, two Fiat-Shamir code-based proposals do not follow the MPCitH paradigm. These two other schemes offer different performance trade-offs: compared with the MPCitH candidates, LESS [9] achieves slightly smaller signature sizes at the cost of a significantly larger public key, whereas CROSS [8] has faster implementations but larger signature sizes.

The hard problems underlying these MPCitH candidates were used in cryptography prior to the advent of MPC-in-the-Head and have thus been studied and cryptanalyzed for several years. More precisely, SDitH [4] is based on the decoding problem in the Hamming metric, Mirath [2] on the decoding problem in the rank metric (usually referred to as the MinRank problem), RYDE [6] on the decoding problem in the rank metric for codes linear over an extension field, and PERK [1] on the Permuted Kernel Problem. As explained above, the key to achieving efficient MPCitH schemes whose security is based on the hardness of such problems is to find "simple" polynomial modelings, where simplicity refers to the ease of verifying knowledge of a solution. Concurrently with the schemes discussed so far, which are based on well-studied problems, this last observation also led to the introduction of ad hoc hard problems with even simpler modelings to build other MPCitH constructions. A representative example of this approach is the NIST multivariate candidate Biscuit [20], which uses a structured MQ system, even though this scheme was severely impacted by cryptanalysis and did not advance to the second round [25]. A more recent example in the code-based setting is the approach by Huth and Joux [41,42] based on the so-called Subfield Bilinear Collision (SBC) problem. This approach yields schemes that are even more compact than the above NIST submissions.

Contributions. With a motivation similar to that of Huth and Joux, our main contribution is to bring a new problem into the realm of cryptography to construct compact MPC-in-the-Head signature schemes. While the problem they consider draws inspiration from techniques used to solve the discrete logarithm problem in small characteristic fields, our new problem instead arises from purely coding-theoretic notions. This problem, inspired by the theory of intersecting codes [45,52,28], can be defined as follows.

Problem 1 (Non-Intersecting Codewords problem) *Let q be a prime power and let $k,\ n \in \mathbb{N}$. Given an $\mathbb{F}_q$-linear code $\mathcal{C}$ of dimension k and length n, the Non-Intersecting Codewords (NIC) problem asks to find two non-zero codewords $\boldsymbol{u},\ \boldsymbol{v} \in \mathcal{C}$ whose supports with respect to the Hamming metric do not intersect, i.e.,*

$$\mathsf{supp}(\boldsymbol{u}) \cap \mathsf{supp}(\boldsymbol{v}) = \emptyset.$$

In this paper, we will focus on the binary case $q = 2$.

Building cryptography based on Problem 1 requires understanding which instances are hard, which in turn depends on the choice of parameters. We first provide a detailed discussion on generating hard binary NIC instances, where we

highlight the importance of the Hamming weights of the secret vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. This leads us to fix these Hamming weights in the instances we consider for our signature scheme. More precisely, we consider a code $\mathcal{C}$ obtained by generating a random code $\mathcal{C}_0$ together with two non-intersecting vectors $\boldsymbol{u}$ and $\boldsymbol{v}$, both of fixed Hamming weight w, outside $\mathcal{C}_0$, and setting

$$\mathcal{C} := \mathcal{C}_0 + \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathbb{F}_2},$$

where $\langle S \rangle_{\mathbb{F}}$ denotes the $\mathbb{F}$-vector space spanned by a set S of vectors. In practice, one is given the code $\mathcal{C}_0$ as well as two linearly independent vectors $\boldsymbol{t}$, $\boldsymbol{t}' \in \mathcal{C}$ which, together with $\mathcal{C}_0$, span $\mathcal{C}$. We choose the value of w both to limit the number of solutions different from $(\boldsymbol{u}, \boldsymbol{v})$ and to make these solutions as difficult as possible to find. Our reasoning here follows a similar line of argument as [41] in the context of Subfield Bilinear Collision, where the authors define instances with a planted solution (they are called *normalized*) and give preliminary comments on the hardness of their problem in [41, §2.2].

We then provide a detailed security analysis of such hard instances of Problem 1. First, we consider combinatorial attacks that consist in enumerating all solutions of weight w to the decoding problem defined by $\mathcal{C}_0$ and the noisy codeword $\boldsymbol{t}$ or $\boldsymbol{t}'$, as one can efficiently recover the NIC solutions from this set of vectors. Moreover, in the binary case, we remark that a similar attack can be performed by decoding $\boldsymbol{t}+\boldsymbol{t}'$ in $\mathcal{C}_0$ with weight $2w$. Secondly, we study in detail a rather natural algebraic approach based on modeling NIC as a system of bilinear equations. In this approach, one is reduced to solving the following n equations in the variables $\boldsymbol{x} \in \mathbb{F}_2^{k-2}$ and $\boldsymbol{y} \in \mathbb{F}_2^{k-2}$:

$$\mathcal{B}(\boldsymbol{x}, \boldsymbol{y}) := (\boldsymbol{t} + \boldsymbol{x}\boldsymbol{G}_0) * (\boldsymbol{t}' + \boldsymbol{y}\boldsymbol{G}_0) = \boldsymbol{0}_n. \tag{1}$$

The complexity of solving this type of bilinear system with Gröbner bases is difficult to estimate. Indeed, this complexity is well understood for generic bilinear systems with fewer equations than variables [35], while our system $\mathcal{B}$ is overdetermined and structured. Interestingly enough, a similar difficulty arises in the case of SBC, as this problem can also be modeled by an overdetermined bilinear system. In the SBC case, Huth and Joux relied on [35] to estimate the complexity of algebraic approaches. Even though one might expect lower complexity since the SBC system is overdetermined, this should not affect security, as the cost reported in [35] is already too high to be realistic and the SBC system is only mildly overdetermined. In our case, where the system is more overdetermined due to the rate chosen for $\mathcal{C}$, we do not rely on [35] and instead provide a dedicated analysis. For this analysis, we follow the work of [7], which applies to generic overdetermined bilinear systems, but we account for the particular structure of $\mathcal{B}$ and include the field equations over $\mathbb{F}_2$ in the Gröbner basis computation. We analyze in the same way hybrid algebraic techniques that leverage the knowledge of the Hamming weight w of the solutions. Concretely, these hybrid techniques perform better than the plain Gröbner basis algorithm, but they do not outperform the combinatorial solvers discussed above, to which they reduce when all variables are fixed.

We finally provide our signature scheme. The underlying zero-knowledge protocol is based on the bilinear modeling from Equation (1), which we directly integrate into the MPCitH paradigm. For that purpose, we can use either the Threshold-Computation-in-the-Head (TCitH) or the VOLE-in-the-Head (VOLEitH) framework. Table 1 reports the sizes of the public key and the signature for a precise choice of parameters corresponding to NIST security level I and provides a comparison to other code-based signature schemes.

Scheme	Assumption	Framework	$\lvert\sigma\rvert$	$\lvert\mathsf{pk}\rvert$	$\lvert\sigma\rvert + \lvert\mathsf{pk}\rvert$
LESS [9]	Code Equivalence Problem	Σ protocol	2 481 B	13 939 B	15 430 B
SDitH [4]	Decoding Problem (Hamming metric)	VOLEitH	3 705 B	70 B	3 775 B
PERK [1]	Permuted Kernel Problem	VOLEitH	3 480 B	100 B	3 580 B
$\mathrm{SBC}^{\mathrm{M}}_{\mathrm{VOLE}}$ [42]	SBC	VOLEitH	2 786 B	48 B	2 834 B
This work	**NIC**	**TCitH**	**3 126 B**	**234 B**	**3 360 B**
This work	**NIC**	**VOLEitH**	**2 962 B**	**234 B**	**3 196 B**

Table 1. Signature and public key sizes in Bytes for several code-based signature schemes in comparison to ours (NIST security level I parameters).

We do not claim any novelty in the protocol itself or in generic optimizations compatible with our construction. However, we note that the NIC problem is particularly well-suited to the MPCitH paradigm, as this problem can be expressed as a set of quadratic constraints, whereas current proofs of knowledge for the decoding problem in the Hamming metric and for the Permuted Kernel Problem rely on higher-degree polynomials. We do perform worse compared to the most compact scheme based on SBC [42], but SBC can also be viewed as more structured, since it relies on an extension field that is leveraged in the proof of knowledge. In that respect, the NIC problem offers an interesting trade-off between compactness and unstructuredness.

2 Preliminaries

Vectors are denoted by small bold letters such as $\boldsymbol{u}$, and are row vectors. Matrices are denoted by bold capital letters such as $\boldsymbol{G}$. For q a prime power, $\mathbb{F}_q$ denotes the finite field with q elements. As mentioned above, $\langle S \rangle_{\mathbb{F}}$ denotes the $\mathbb{F}$-vector space spanned by a set S of vectors. We reserve the notation $\langle \mathcal{S} \rangle$, without a subscript, for the ideal generated by a set $\mathcal{S}$ of polynomials. We denote by $\{1..N\}$ the set of integers ranging from 1 to N.

The component-wise or *star-product* of two vectors $\boldsymbol{u} := (u_1, u_2, \ldots, u_n)$ and $\boldsymbol{v} := (v_1, v_2, \ldots, v_n)$ of length n is the vector $\boldsymbol{u} * \boldsymbol{v}$ of length n defined by

$$\boldsymbol{u} * \boldsymbol{v} := (u_1 v_1, u_2 v_2, \ldots, u_n v_n).$$

2.1 Linear codes

A linear code $\mathcal{C}$ of length n and dimension $k \leq n$ over $\mathbb{F}_q$ is an $\mathbb{F}_q$-linear subspace of $\mathbb{F}_q^n$ of dimension k. We refer to it as an $[n,k]$-code. A generator matrix for $\mathcal{C}$ is a matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$ such that $\mathcal{C} = \{\boldsymbol{mG},\ \boldsymbol{m} \in \mathbb{F}_q^k\}$ and a parity-check matrix is a matrix $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times n}$ such that $\mathcal{C} = \{\boldsymbol{y} \in \mathbb{F}_q^n,\ \boldsymbol{yH}^\mathsf{T} = \boldsymbol{0}\}$. The support of a vector $\boldsymbol{v} = (v_1, \ldots, v_n) \in \mathbb{F}_q^n$ with respect to the Hamming metric is defined as

$$\mathsf{supp}(\boldsymbol{v}) := \{i \in \{1..n\},\ v_i \neq 0\},$$

and the Hamming weight of $\boldsymbol{v}$ is the cardinality of this support, namely

$$w_H(\boldsymbol{v}) := \#\mathsf{supp}(\boldsymbol{v}).$$

Throughout the paper, we will often refer to the decoding problem in the Hamming metric, which we define in Problem 2.

Problem 2 (Decoding problem in the Hamming metric) *Let q be a power of a prime and let $k,\ n \in \mathbb{N}$ with $k \leq n$. Given $\boldsymbol{G} \in \mathbb{F}_q^{k\times n}$ a full-rank generator matrix of an $[n,k]$-code $\mathcal{C}$ and a vector $\boldsymbol{t} \in \mathbb{F}_q^n$, the Decoding problem in the Hamming metric asks to retrieve a vector $\boldsymbol{e} \in \mathbb{F}_q^n$ of Hamming weight $w_H(\boldsymbol{e}) = w$ such that $\boldsymbol{t} = \boldsymbol{mG} + \boldsymbol{e}$ for some $\boldsymbol{m} \in \mathbb{F}_q^k$. The corresponding decoding instance will be written as $(\boldsymbol{G}, \boldsymbol{t})$.*

We will use the following results to estimate the average number of solutions to Problem 2 when $\mathcal{C}$ is a random linear code.

Lemma 1. *The number of words of Hamming weight $w \leq n$ in $\mathbb{F}_q^n$ is given by*

$$N_{q,n,w} := \tbinom{n}{w}(q-1)^w.$$

Definition 1 (Gilbert–Varshamov distance). *For integers q and $k \leq n$, the Gilbert–Varshamov distance $d_{GV}(q,n,k)$ is defined as the largest integer d such that*

$$\sum_{w=0}^{d} N_{q,n,w} \leq q^{n-k}.$$

The Gilbert–Varshamov relative distance is defined as $\delta_{GV}(q,n,k) := d_{GV}(q,n,k)/n$.

With high probability, the parameters of a random code are close to the Gilbert–Varshamov distance. In this paper, a random code of length n and dimension k over $\mathbb{F}_q$ will always refer to a code with generator matrix

$$\boldsymbol{G} := \begin{bmatrix}\boldsymbol{I_k}\ \boldsymbol{A}\end{bmatrix} \in \mathbb{F}_q^{k\times n},$$

where $\boldsymbol{A} \in \mathbb{F}_q^{k\times(n-k)}$ is sampled uniformly at random. The probability that a fixed non-zero vector $\boldsymbol{u} \in \mathbb{F}_q^n$ lies in a such a code $\mathcal{C}$ is *upper-bounded* by $1/q^{n-k}$, and the probability that two non-zero and non-collinear vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ lie in $\mathcal{C}$ is upper-bounded by $1/q^{2(n-k)}$. With that in mind, we have the following statement.

Lemma 2. *For integers q, n, k, w such that $k \leq n$ and $w \leq n$, the average number of solutions of Hamming weight w to a decoding instance $(\boldsymbol{G}, \boldsymbol{t})$, where $\boldsymbol{G} = \left[\boldsymbol{I_k}\ \boldsymbol{A}\right] \in \mathbb{F}_q^{k \times n}$ is a generator matrix of a random $[n,k]$-code and $\boldsymbol{t} \in \mathbb{F}_q^n$, denoted by $N_{q,n,w,k}$, is such that*

$$N_{q,n,w,k} \leq \frac{N_{q,n,w}}{q^{n-k}} = \frac{\binom{n}{w}(q-1)^w}{q^{n-k}}. \tag{2}$$

For $\boldsymbol{t} = \boldsymbol{0}_n$, this yields the average number of words of Hamming weight w in $\mathcal{C}$.

The definition of our new problem is closely related to the notion of an intersecting code, as introduced in [45,52]. An intersecting code is a code in which the supports of any two nonzero codewords intersect. We will use the following result on the probability that a random $[n,k]$-code is non-intersecting. This result is not new and can be found as existence theorems in [23] or [28].

Proposition 1. *Let $\mathcal{C}$ be a random code of length n and dimension k over $\mathbb{F}_q$, with generator matrix $\left[\boldsymbol{I_k}\ \boldsymbol{A}\right]$ for a uniformly sampled $\boldsymbol{A} \in \mathbb{F}_q^{k \times (n-k)}$. Then, $\mathcal{C}$ is intersecting with probability $p_\cap$ such that*

$$p_\cap \geq 1 - (2q-1)^n \cdot \frac{1}{q^{2(n-k)}}. \tag{3}$$

Proof. First, we count the number of nonzero pairs of vectors that are non-intersecting. To do so, let $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{F}_q^n \setminus \{\boldsymbol{0}_n\} \times \mathbb{F}_q^n \setminus \{\boldsymbol{0}_n\}$ be non-intersecting. Then, each pair of coordinates (x_i, y_i) can take every possible value *except* $x_i \neq 0$ *and* $y_i \neq 0$ for the same index $i \in \{1..n\}$. There are, at most, $(q^2 - (q-1)^2)^n = (2q-1)^n$ such pairs of vectors. As the probability that a fixed nonzero vector in $\mathbb{F}_q^n$ lies in a random code $\mathcal{C}$ of length n and dimension k over $\mathbb{F}_q$ is upper-bounded by $1/q^{n-k}$, we deduce that the expected number of non-intersecting pairs in $\mathcal{C}$ is upper-bounded by

$$(2q-1)^n \cdot \frac{1}{q^{2(n-k)}}.$$

From Markov's inequality, it follows that a random code contains at least one pair of non-intersecting codewords with a probability

$$\Pr(\mathcal{C} \text{ is not intersecting}) \leq (2q-1)^n \cdot \frac{1}{q^{2(n-k)}},$$

Finally,

$$\Pr(\mathcal{C} \text{ is intersecting}) \geq 1 - (2q-1)^n \cdot \frac{1}{q^{2(n-k)}}.$$

□

We will use this result for $q = 2$ to fix parameters later on. In this case, the right-hand side of Equation (3) becomes $1 - \varepsilon(n,k)$ for

$$\varepsilon(n,k) = \left(\frac{3^n}{4^{(n-k)}}\right).$$

This latter quantity is < 1 whenever

$$k < \tfrac{1}{2}\log_2(\tfrac{4}{3})n. \tag{4}$$

2.2 MPC-in-the-Head paradigm

The Multi-Party Computation in the Head (MPCitH) paradigm, introduced in [43], allows building efficient zero-knowledge proofs by borrowing techniques from secure multi-party computation. Concretely, an MPCitH zero-knowledge proof is an interactive protocol between a prover and a verifier in which the prover emulates an MPC protocol to prove the knowledge of a witness $\boldsymbol{w} \in \mathbb{F}_q^m$ corresponding to a public statement $\boldsymbol{x}$, such that $(\boldsymbol{x}, \boldsymbol{w}) \in \mathcal{R}$ for some relation $\mathcal{R}$. More recently, the MPCitH paradigm has attracted significant attention, as it has proven to be very successful for constructing efficient signature schemes [4,6,2,1,17,13], where the zero-knowledge proof is transformed into a non-interactive protocol via the Fiat-Shamir heuristic [40].

In this context, we may focus on relations of the form $(\boldsymbol{x}, \boldsymbol{w}) \in \mathcal{R} \Leftrightarrow \mathcal{F}_{\boldsymbol{x}}(\boldsymbol{w}) = \boldsymbol{0}$, where $\mathcal{F}_{\boldsymbol{x}}$ represents a polynomial modeling of a hard problem instance, or is itself already a multivariate polynomial system in some cases. To prove that the witness $\boldsymbol{w}$ is a solution to $\mathcal{F}_{\boldsymbol{x}}$, the prover shares this witness into N shares – denoted by $[\![\boldsymbol{w}]\!]_i$ – by using a secret sharing scheme, and then performs operations on these shares as depicted in Figure 1. The whole protocol is then repeated a number $\tau \geq 1$ of times to ensure that an adversary cannot predict the challenges.

[Public data] Polynomial constraints $\mathcal{F}_{\boldsymbol{x}} = (f_1, \ldots, f_n) \in \mathbb{F}_2[x_1, \ldots, x_m]^n$ that model a hard problem.

Step 1: Generation of the shares.

1. The prover generates the N shares $[\![\boldsymbol{w}]\!]_1, \ldots, [\![\boldsymbol{w}]\!]_N$;
2. The prover commits to the N shares, and sends the commitment to the verifier.
3. The verifier sends a challenge $\gamma_{1,1}, \ldots, \gamma_{n,\rho}$ to the prover;

Step 2: MPC protocol.

1. The prover computes $[\![\alpha^{(l)}]\!]_i = \sum_{j=1}^{n} \gamma_{j,l} f_j([\![\boldsymbol{w}]\!]_i)$ for $i \in \{1..N\}$ and $l \in \{1..\rho\}$;
2. The prover commits to the shares of $\alpha^{(1)}, \ldots, \alpha^{(\rho)}$ and sends the commitment to the verifier.

Step 3: Reveal of the views.

1. The verifier asks the prover ℓ shares of $\boldsymbol{w}$, and one (or several) shares of all $\alpha^{(l)}$;
2. The prover transmits the shares. The verifier checks whether the prover was honest.

Fig. 1. High-level description of the MPC-in-the-Head framework.

The Threshold-Computation-in-the-Head (TCitH) [38,39] and the VOLE-in-the-Head (VOLEitH) frameworks [15] were introduced later to improve the performance of such types of proofs. We provide a brief discussion of these frameworks, as they are used in our signature scheme.

In TCitH, the key improvement is the efficient use of Shamir's secret sharing (SSS) instead of an additive secret sharing scheme, which enables much simpler MPC protocols. For instance, given SSS shares $[\![a]\!]_1, \ldots, [\![a]\!]_N$ and $[\![b]\!]_1, \ldots, [\![b]\!]_N$, the prover can compute $[\![a \cdot b]\!]_i$ by simply computing $[\![a]\!]_i \cdot [\![b]\!]_i$, while performing

the same operation with additive shares requires significantly more computation. The original TCitH framework was largely improved in [39], where it was in particular applied to a general MPC protocol that checks polynomial constraints [39, Protocol 7]. The resulting proof of knowledge is a 5-pass protocol, with soundness (over $\mathbb{F}_2$)

$$\frac{d}{N} + \left(1 - \frac{d}{N}\right) \cdot \frac{1}{2^\rho},$$

where d is the common degree of the polynomials in $\mathcal{F}_{\boldsymbol{x}}$, and ρ is the number of repetitions of the MPC protocol.

Although introduced earlier than TCitH, VOLEitH was shown in [38] to be expressible using the same formalism. For instance, a VOLEitH proof also uses SSS, and generates the shares in the same way as in TCitH. However, it differs in several points. A first point is that shares are merged: instead of τ sharings composed of N shares, there is only *one* sharing composed of N^τ shares. This reduces the soundness error to

$$\frac{d}{N^\tau} + \left(1 - \frac{d}{N^\tau}\right) \cdot \frac{1}{2^\rho},$$

at the cost of computations in a larger field. A second point, implied by the first one, is that a new pass is added to the protocol to verify the merging of the shares. The resulting proof is then a 7-pass protocol. This last difference introduces an extra parameter B to control the cheating probability of the new pass.

While the polynomial modeling $\mathcal{F}_{\boldsymbol{x}}$ – and thus the performance of the scheme – depends on the underlying hard problem, we observe that the TCitH and VOLEitH frameworks contain a substantial problem-agnostic component corresponding to Steps 1 and 3 in Figure 1. In particular, those frameworks largely benefit from *generic optimizations* such as those introduced in [14]. In this work, we will consider the following ones:

- Improvements to the GGM trees, by introducing a threshold T_{open};
- Incorporation of a proof-of-work, providing p_{work} bits of additional security and reducing the number of repetitions required in the Fiat–Shamir transform.

Note that other generic optimizations exist, such as the correlated trees of [47,41], or the MinMax commitments from [42]. However, their interaction with the two aforementioned ones is not straightforward from a security point of view, and we will not take them into account. In particular, our signature scheme will strictly follow Mirath and RYDE in considering only these former optimizations.

3 NIC-Based Signature Scheme

In this section, we show that one can use Problem 1 to construct an MPCitH signature scheme. This scheme is based on a set of polynomial constraints that

directly yield an efficient protocol for verifying a NIC solution, and we can make it even more competitive by relying on the TCiTH and VOLEitH frameworks. Before presenting our signature scheme, we begin with a discussion on the number of solutions to a NIC instance, which is comparable to that in [41, §2.2] for SBC. This discussion will guide us in selecting hard instances to instantiate our construction.

3.1 Planted NIC instances

Our scheme requires a NIC instance with only one or a small number of solutions. Constructing such instances is not trivial, and this is the subject of the following discussion.

To construct such an instance, one natural approach is to choose a code $\mathcal{C}$ with an intersecting subcode $\mathcal{C}_0$ that is as large as possible. Recalling Proposition 1 and its instantiation in the case $q = 2$ given by Equation (4), we start from a random code $\mathcal{C}_0$ of dimension $k' = k - 2$ and length n over $\mathbb{F}_2$ such that

$$n \geq \frac{2}{\log_2(\frac{4}{3})} k' \approx 4.82k'. \tag{5}$$

Such a code is expected to be intersecting with high probability, lower-bounded in Equation (3). Then, we enforce a non-trivial solution by adding two non-collinear vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ with disjoint supports. Such vectors fall outside $\mathcal{C}_0$ with probability at least

$$1 - \frac{2}{2^{n-k'}} = 1 - \frac{2}{2^{n-k+2}}.$$

In this case, we set

$$\mathcal{C} := \mathcal{C}_0 \oplus \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathbb{F}_2}.$$

Hamming weight constraints. So far, we have not specified how to choose the non-intersecting vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ apart from the fact that they should lie outside $\mathcal{C}_0$. Due to the non-intersecting condition, their Hamming weights should already satisfy

$$w_H(\boldsymbol{u}) + w_H(\boldsymbol{v}) = w_H(\boldsymbol{u} + \boldsymbol{v}) \leq n,$$

which imposes the constraint $\min\big(w_H(\boldsymbol{u}), w_H(\boldsymbol{v})\big) < n/2$, also mentioned in [28, Lemma 1.2]. Then, we would like to avoid having $w_H(\boldsymbol{u} + \boldsymbol{v}) = n$, since the all-one vector $\mathbf{1}_n = \boldsymbol{u} + \boldsymbol{v}$ would then belong to $\mathcal{C}$ and for any $\boldsymbol{c} \in \mathcal{C}$, the pair $(\mathbf{1}_n - \boldsymbol{c}, \boldsymbol{c})$ would be a solution to the NIC instance defined by $\mathcal{C}$. We can obtain stronger constraints on the Hamming weights of $\boldsymbol{u}$ and $\boldsymbol{v}$ by generalizing this argument. Namely, if there exists a nonzero codeword $\boldsymbol{c} \in \mathcal{C}_0$ such that

$$\mathsf{supp}(\boldsymbol{c}) \subset \mathsf{supp}(\boldsymbol{u}) \cup \mathsf{supp}(\boldsymbol{v}),$$

then the pair $(\boldsymbol{c}, \boldsymbol{u}+\boldsymbol{v}-\boldsymbol{c})$ will be a non-trivial solution to the NIC problem. More generally, if there exist two distinct codewords $\boldsymbol{c},\ \boldsymbol{c}' \in \mathcal{C}$ such that $\mathsf{supp}(\boldsymbol{c}') \subset \mathsf{supp}(\boldsymbol{c})$, then $(\boldsymbol{c}', \boldsymbol{c} + \boldsymbol{c}')$ will be a non-trivial solution to the NIC problem.

This discussion guides us to choose $\boldsymbol{u}$ and $\boldsymbol{v}$ with a common prescribed Hamming weight w. To later fix this value in our signature scheme, we start by giving an upper bound on the probability that such $\boldsymbol{u}$ and $\boldsymbol{v}$ introduce other non-intersecting codewords. We sum this up in the following lemma.

Lemma 3. *Let $\mathcal{C}$ be a binary instance of Problem 1 defined by*

$$\mathcal{C} := \mathcal{C}_0 + \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathbb{F}_2},$$

where $\mathcal{C}_0$ is a random code defined by a generator matrix in systematic form $\boldsymbol{G} := \begin{bmatrix} \boldsymbol{I_k} & \boldsymbol{A} \end{bmatrix}$ for a uniformly sampled matrix $\boldsymbol{A} \in \mathbb{F}_2^{k \times (n-k)}$, where k and n are chosen according to Equation 3 and where $\boldsymbol{u},\ \boldsymbol{v} \in \mathbb{F}_2^n$ are non-intersecting vectors of weight w. Then, the probability that there exists a pair of vectors $(\boldsymbol{u}', \boldsymbol{v}') \neq (\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C} \times \mathcal{C}$ such that $(\boldsymbol{u}', \boldsymbol{v}')$ is a solution to the NIC instance is upper-bounded by

$$p_0 + \varepsilon_{\boldsymbol{u}} + \varepsilon_{\boldsymbol{v}} + \varepsilon_{\boldsymbol{u},\boldsymbol{v}},$$

where

$$p_1 := \frac{3^n}{2^{2(n-k)}}, \quad p_2 := \frac{2^{n-w}}{2^{n-k}}, \quad p_3 := \frac{2^w}{2^{n-k}},$$
$$p_4 := \frac{2^{n-2w}}{2^{n-k}}, \quad p_5 := \frac{2^{2w}}{2^{n-k}},$$

$$\varepsilon_{\boldsymbol{u}} = \varepsilon_{\boldsymbol{v}} := 2 \cdot p_1 + 2 \cdot p_2 + p_3 \quad \textit{and} \quad \varepsilon_{\boldsymbol{u},\boldsymbol{v}} := 5 \cdot p_1 + 2 \cdot p_4 + p_5,$$

and $p_0 := 3^n/4^{n-k}$. Furthermore, with probability at least $1 - 3/2^{n-k}$, none of $\boldsymbol{u}$, $\boldsymbol{v}$, and $\boldsymbol{u} + \boldsymbol{v}$ are codewords of $\mathcal{C}_0$.

Proof. First, Proposition 1 gives a lower bound $1 - p_0 := 1 - 3^n/4^{n-k}$ on the probability $p_\cap$ that the random code $\mathcal{C}_0$ is intersecting. In other words, the probability that this code is non-intersecting is upper-bounded by p_0. To argue that we do not introduce a new solution, we first look at an upper bound on the probability that $\mathcal{C}_0 + \langle \boldsymbol{u} \rangle_{\mathbb{F}_2}$ is not intersecting. Only after will we consider $\mathcal{C}_0 + \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathbb{F}_2}$. Recall that, for two distinct fixed vectors $\boldsymbol{x},\ \boldsymbol{y} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}$, the probability that both belong to $\mathcal{C}_0$ can be upper-bounded by $1/2^{2(n-k)}$.

Our first goal is to upper-bound the probability that $(\boldsymbol{c} + \boldsymbol{u}, \boldsymbol{c}' + \boldsymbol{u})$ is non-intersecting for some $\boldsymbol{c} \in \mathcal{C}_0$ and $\boldsymbol{c}' \in \mathcal{C}_0 + \langle \boldsymbol{u} \rangle_{\mathbb{F}_2}$. To that end, we divide the possibilities of introducing a non-intersecting pair into several cases, for $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}_n\} \times \mathbb{F}_2^n \setminus \{\boldsymbol{0}_n\}$ with $\boldsymbol{x} \neq \boldsymbol{y}$:

- $(\boldsymbol{x} + \boldsymbol{u}, \boldsymbol{y})$ is non-intersecting. Then, $u_i = 0$ implies $(x_i, y_i) \neq (1, 1)$, while if $u_i = 1$, $(x_i, y_i) \neq (0, 1)$. There are thus at most 3^n pairs of vectors $(\boldsymbol{x}, \boldsymbol{y})$ that make $(\boldsymbol{x} + \boldsymbol{u}, \boldsymbol{y})$ non-intersecting, and the probability that one or more pairs $(\boldsymbol{x}, \boldsymbol{y})$ lie in $\mathcal{C}_0$ is upper-bounded by

$$p_1 := \frac{3^n}{2^{2(n-k)}};$$

– $(\boldsymbol{x}+\boldsymbol{u}, \boldsymbol{y}+\boldsymbol{u})$ is non-intersecting. Then, $u_i = 0$ implies $(x_i, y_i) \neq (1, 1)$, while $u_i = 1$ implies $(x_i, y_i) \neq (0, 0)$. Once again, the probability that one or more pairs of such $\boldsymbol{x}, \boldsymbol{y}$ lie in $\mathcal{C}_0$ is upper-bounded by p_1.

We still have to treat the cases $(\boldsymbol{x}, \boldsymbol{u})$, $(\boldsymbol{x}+\boldsymbol{u}, \boldsymbol{u})$, and $(\boldsymbol{x}+\boldsymbol{u}, \boldsymbol{x})$. As 2^{n-w} vectors (including $\mathbf{0}_n$) have a support that does not intersect with that of $\boldsymbol{u}$, the probability for $(\boldsymbol{x}, \boldsymbol{u})$ to be non-intersecting and $\boldsymbol{x} \in \mathcal{C}_0 \setminus \{\mathbf{0}_n\}$ is upper-bounded by

$$p_2 := \frac{2^{n-w}}{2^{n-k}}.$$

On the other hand, if $(\boldsymbol{x}+\boldsymbol{u}, \boldsymbol{u})$ is non-intersecting, then the support of $\boldsymbol{u}$ is included in the one of $\boldsymbol{x}$, which gives the upper bound p_2 once again (there are 2^{n-w} vectors whose support includes that of $\boldsymbol{u}$). For the last case, if $(\boldsymbol{x}+\boldsymbol{u}, \boldsymbol{x})$ is non-intersecting, then the support of $\boldsymbol{x}$ is included in the support of $\boldsymbol{u}$. There are 2^w such vectors, and so the probability that at least one of them is in $\mathcal{C}_0$ is

$$p_3 := \frac{2^w}{2^{n-k}}.$$

Overall, the probability that $\mathcal{C}_0 + \langle \boldsymbol{u} \rangle_{\mathbb{F}_2}$ is non-intersecting is upper-bounded by

$$\varepsilon_{\boldsymbol{u}} := 2 \cdot p_1 + 2 \cdot p_2 + p_3.$$

We now study the effect of adding $\boldsymbol{v}$ to the code $\mathcal{C}_0 + \langle \boldsymbol{u} \rangle_{\mathbb{F}_2}$. First, we denote by $\varepsilon_{\boldsymbol{v}}$ an upper bound on the probability that $\mathcal{C}_0 + \langle \boldsymbol{v} \rangle_{\mathbb{F}_2}$ is a non-intersecting code (we can clearly take $\varepsilon_{\boldsymbol{v}} = \varepsilon_{\boldsymbol{u}}$). Then, we aim at bounding the probability that $(\boldsymbol{c}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{c}'+\boldsymbol{u}+\boldsymbol{v})$ is non-intersecting, for some $\boldsymbol{c} \in \mathcal{C}_0 + \langle \boldsymbol{u} \rangle_{\mathbb{F}_2}$ and $\boldsymbol{c}' \in \mathcal{C}_0 + \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathbb{F}_2}$. We proceed as previously, for $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{F}_2^n \setminus \{\mathbf{0}_n\} \times \mathbb{F}_2^n \setminus \{\mathbf{0}_n\}$ and $\boldsymbol{x} \neq \boldsymbol{y}$, consider:

– $(\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{y}+\boldsymbol{u}+\boldsymbol{v})$ is non-intersecting. As $\boldsymbol{u}$ and $\boldsymbol{v}$ are non-intersecting, if $u_i = 1$, then $v_i = 0$, and reciprocally $v_i = 1$ implies $u_i = 0$. Thus, if $u_i = 1$ or if $v_i = 1$, then $(x_i, y_i) \neq (0, 0)$. If $u_i = 0$ and $v_i = 0$, then $(x_i, y_i) \neq (1, 1)$, and so the probability that such $\boldsymbol{x}, \boldsymbol{y}$ are in the code $\mathcal{C}_0$ is upper-bounded by p_1;
– $(\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{y}+\boldsymbol{v}), (\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{y}+\boldsymbol{u}), (\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{y})$, or $(\boldsymbol{x}+\boldsymbol{u}, \boldsymbol{y}+\boldsymbol{v})$ are non-intersecting. With the same reasoning, the probability that $\boldsymbol{x}$ and $\boldsymbol{y}$ are in $\mathcal{C}_0$ is upper-bounded by $4 \cdot p_1$ (once per case).

Finally, the last new cases to consider are $(\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{x})$, $(\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{u}+\boldsymbol{v})$, and $(\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{x})$ non-intersecting. The probability that $\boldsymbol{x} \in \mathcal{C}_0$ is non-intersecting with $\boldsymbol{u}+\boldsymbol{v}$ is at most

$$p_4 := \frac{2^{n-2w}}{2^{n-k}}.$$

The same goes for $(\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{u}+\boldsymbol{v})$: this is a case where the support of $\boldsymbol{u}+\boldsymbol{v}$ is included in $\boldsymbol{x}$, which yields the same probability. Finally, $(\boldsymbol{x}+\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{x})$ is

non-intersecting whenever the support of $\boldsymbol{x}$ is included in that of $\boldsymbol{u}+\boldsymbol{v}$. There are 2^{2w} such vectors, resulting in the upper bound

$$p_5 := \frac{2^{2w}}{2^{n-k}}.$$

Thus, the probability to have added a non-intersecting vector into $\mathcal{C}+\langle \boldsymbol{u}\rangle_{\mathbb{F}_2}$ is

$$\varepsilon_{\boldsymbol{u},\boldsymbol{v}} := 5\cdot p_1 + 2\cdot p_4 + p_5.$$

We conclude by putting everything together: the probability that there exists at least one pair of non-intersecting codewords $(\boldsymbol{x},\boldsymbol{y}) \neq (\boldsymbol{u},\boldsymbol{v})$ in $\mathcal{C}_0+\langle \boldsymbol{u},\boldsymbol{v}\rangle_{\mathbb{F}_2}$ is upper-bounded by

$$\underbrace{\varepsilon_{\boldsymbol{u}}}_{\mathcal{C}_0+\langle \boldsymbol{u}\rangle_{\mathbb{F}_2}} + \underbrace{\varepsilon_{\boldsymbol{v}}}_{\mathcal{C}_0+\langle \boldsymbol{v}\rangle_{\mathbb{F}_2}} + \underbrace{\varepsilon_{\boldsymbol{u},\boldsymbol{v}}}_{\mathcal{C}_0+\langle \boldsymbol{u},\boldsymbol{v}\rangle_{\mathbb{F}_2}},$$

which is the statement of the lemma. Note that this bound is not tight, as we may have counted some vectors several times.

Finally, the probability that $\boldsymbol{u}\in\mathcal{C}_0$ is upper-bounded by $1/2^{n-k}$. The same goes for $\boldsymbol{v}$ and $\boldsymbol{u}+\boldsymbol{v}$. The probability that at least one of the three vectors lies in $\mathcal{C}_0$ is thus upper-bounded by $3/2^{n-k}$, and the desired lower bound that none of them do follows. □

Using this lemma, our goal is now to determine the optimal weight w, i.e., to choose between $n/3$, $n/4$, $n/5$, or any other value. For that purpose, observe that the value p_1 defined in Lemma 3 does not depend on w. Then, finding the optimal value w for $\boldsymbol{u}$ and $\boldsymbol{v}$ amounts to (roughly) minimizing $\max(p_2,p_3,p_4,p_5)$, that is we aim at minimizing the value $\max(n-w,w,n-2w,2w)$. Already, the optimal value seems to lie around $w\approx n/3$, for which the maximum is $2n/3$. Taking w lower would make $n-w$ higher, and a higher w would make $2w$ exceed $2n/3$. Another compelling argument confirms this choice: one can efficiently solve Problem 1 from the knowledge of $\boldsymbol{u}$ or $\boldsymbol{v}$, and even from $\boldsymbol{u}+\boldsymbol{v}$ in the case of $\mathbb{F}_2$. The latter statement is a consequence of the following result.

Lemma 4. *Let $\mathcal{C}$ be a binary instance of Problem 1 defined by*

$$\mathcal{C} := \mathcal{C}_0 + \langle \boldsymbol{u},\boldsymbol{v}\rangle_{\mathbb{F}_2},$$

where $\mathcal{C}_0$, $\boldsymbol{u}$, and $\boldsymbol{v}$ are defined as in Lemma 3. We have:

1. *$\boldsymbol{u}\in(\boldsymbol{s}*\mathcal{C})\cap\mathcal{C}$ and $\boldsymbol{v}\in(\boldsymbol{s}*\mathcal{C})\cap\mathcal{C}$, which implies*

$$\dim\left((\boldsymbol{s}*\mathcal{C})\cap\mathcal{C}\right)\geq 2. \tag{6}$$

2. *$\boldsymbol{t}\in(\boldsymbol{t}*\mathcal{C})\cap\mathcal{C}$ for any other codeword $\boldsymbol{t}\in\mathcal{C}$, which implies*

$$\dim\left((\boldsymbol{t}*\mathcal{C})\cap\mathcal{C}\right)\geq 1. \tag{7}$$

Furthermore, taking the same notation as in Lemma 3, the inequalities (6) and (7) are equalities with probability at least

$$1 - (p_0 + \varepsilon_{\boldsymbol{u}} + \varepsilon_{\boldsymbol{v}} + \varepsilon_{\boldsymbol{u},\boldsymbol{v}}) .$$

Proof. First, note that $\boldsymbol{t} \in (\boldsymbol{t} * \mathcal{C}) \cap \mathcal{C}$ is true for any codeword $\boldsymbol{t} \in \mathcal{C}$, which directly yields the second statement. For the first statement, we have

$$\boldsymbol{s} * \boldsymbol{u} = (\boldsymbol{u} + \boldsymbol{v}) * \boldsymbol{u} = \boldsymbol{u} * \boldsymbol{u} = \boldsymbol{u},$$

where the last equality holds because we are working over $\mathbb{F}_2$. This shows that $\boldsymbol{u} \in (\boldsymbol{s} * \mathcal{C}) \cap \mathcal{C}$, and the statement for $\boldsymbol{v}$ is analogous. To argue that (6) and (7) are equalities with the stated lower bound on the probability, observe that for any $\boldsymbol{t} \in \mathcal{C}$, $(\boldsymbol{t} * \mathcal{C}) \cap \mathcal{C}$ includes all the codewords of $\mathcal{C}$ that have a support included in $\boldsymbol{t}$. The presence of a vector linearly independent of $\boldsymbol{t}$ in this space directly yields a NIC solution. Lemma 3 yields the result. □

Inequality (6) tells us that finding specific codewords of weight $2w$ in $\mathcal{C}$ should be essentially as difficult as finding codewords of weight w. Thus, the previous choice of $w \approx n/3$ ensures that the number of codewords of weight w is approximately the same as the number of codewords of weight $2w$ in $\mathcal{C}$. Moreover, since we have the constraint $k/n \approx 0.2$ from the dimension of $\mathcal{C}_0$, this number of codewords is exponential in n, as we are well above the Gilbert–Varshamov bound $\delta_{\mathrm{GV}} \approx 0.22$. More precisely, taking a standard lower bound on the binomial coefficient $\binom{n}{n/3}$ yields an estimate of roughly $2^{k-0.08n}$ codewords of weight $n/3$ in $\mathcal{C}$. Thus, this choice of w allows for a situation where the probability of having a unique solution, given in Lemma 3, is very close to 1, while making $\boldsymbol{u} + \boldsymbol{v}$ as hard to find as $\boldsymbol{u}$ and $\boldsymbol{v}$.

Decoding problem formulation. The importance of the Hamming weight in generating hard NIC instances motivates us to introduce the problem in a slightly different, yet entirely equivalent, formulation. Recall that the public code $\mathcal{C}$ was computed as

$$\mathcal{C} = \mathcal{C}_0 \oplus \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathbb{F}_2}.$$

Instead, one could reveal $\mathcal{C}_0$ together with two noisy codewords $\boldsymbol{t} = \boldsymbol{u} + \boldsymbol{c}$ and $\boldsymbol{t}' = \boldsymbol{v} + \boldsymbol{c}'$, where $\boldsymbol{c}$ and $\boldsymbol{c}'$ are two elements of $\mathcal{C}_0$, and ask to decode them under the constraint that the errors have disjoint supports. The equivalence with the original formulation follows in the same way as the equivalence between Problem 2 and the problem of finding low-weight codewords: given $\mathcal{C}_0$ and $\boldsymbol{t}$, $\boldsymbol{t}'$, computing $\mathcal{C} = \mathcal{C}_0 \oplus \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\mathbb{F}_2}$ is straightforward. At the same time, the reciprocal reduction can be performed in polynomial time by changing the basis of $\mathcal{C}$.

We will use this second formulation in the rest of the paper, as it leads to a simpler description of the MPCitH protocol and of the cryptanalysis of NIC. Combining this formulation with our discussion on the Hamming weights leads us to define the following problem, whose instances are expected to have a unique (planted) solution. We introduce a constant parameter δ added to $n/3$ to provide greater flexibility, mainly when n is not divisible by 3.

Problem 3 (Planted Binary NIC problem with parameter δ) *Let δ be a fixed integer constant, and let k' and n be integers chosen with respect to Equation (5). Let $\mathcal{C}_0$ be a random code of dimension k' and length n over $\mathbb{F}_2$ with generator matrix $\left[\boldsymbol{I}_{\boldsymbol{k'}}\ \boldsymbol{A}\right]$ for a randomly sampled $\boldsymbol{A} \in \mathbb{F}_2^{k' \times (n-k')}$. Finally, let $\boldsymbol{u},\ \boldsymbol{v} \in \mathbb{F}_2^n$ be non-intersecting vectors of weight $w = n/3 + \delta$ outside of $\mathcal{C}_0$ and let*

$$(\boldsymbol{t}, \boldsymbol{t}') = (\boldsymbol{u} + \boldsymbol{c}, \boldsymbol{v} + \boldsymbol{c}'),$$

where $\boldsymbol{c},\ \boldsymbol{c}' \in \mathcal{C}_0$. The planted binary Non-Intersecting Codewords problem of parameter δ asks to find two non-zero vectors $\tilde{\boldsymbol{u}},\ \tilde{\boldsymbol{v}} \in \mathbb{F}_2^n$, such that, for some $\boldsymbol{d},\ \boldsymbol{d}' \in \mathcal{C}_0$, we have:

$$\mathsf{supp}(\tilde{\boldsymbol{u}}) \cap \mathsf{supp}(\tilde{\boldsymbol{v}}) = \emptyset \quad \textit{with} \quad \boldsymbol{t} = \tilde{\boldsymbol{u}} + \boldsymbol{d} \quad \textit{and} \quad \boldsymbol{t}' = \tilde{\boldsymbol{v}} + \boldsymbol{d}'.$$

In the following, we continue to refer to k as the dimension of $\mathcal{C}$, so that $k' = k-2$.

3.2 NIC polynomial constraints

We now give the set of polynomial constraints that we use to instantiate our signature scheme. This is nothing more than the direct modeling of Problem 3 already discussed in the introduction, namely the set of n bilinear equations in the variables $\boldsymbol{x} \in \mathbb{F}_2^{k-2}$ and $\boldsymbol{y} \in \mathbb{F}_2^{k-2}$ given by the n coordinates of the star-product

$$\mathcal{B}(\boldsymbol{x}, \boldsymbol{y}) := (\boldsymbol{t} + \boldsymbol{x}\boldsymbol{G}_0) * (\boldsymbol{t}' + \boldsymbol{y}\boldsymbol{G}_0), \tag{8}$$

where $(\boldsymbol{t}, \boldsymbol{t}') \in \mathbb{F}_2^n \times \mathbb{F}_2^n$ is the public statement and where $\boldsymbol{G}_0 \in \mathbb{F}_2^{(k-2)\times n}$ is a full-rank generator matrix of $\mathcal{C}_0$. By definition, we know that the planted NIC solution $(\boldsymbol{u}, \boldsymbol{v})$ satisfies $\boldsymbol{t} = \boldsymbol{u} + \widetilde{\boldsymbol{x}}\boldsymbol{G}_0$ and $\boldsymbol{t}' = \boldsymbol{v} + \widetilde{\boldsymbol{y}}\boldsymbol{G}_0$ for some $\widetilde{\boldsymbol{x}},\ \widetilde{\boldsymbol{y}} \in \mathbb{F}_2^{k-2}$, and $\boldsymbol{u} * \boldsymbol{v} = \boldsymbol{0}_n$. This shows that $\mathcal{B}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}) = \boldsymbol{0}_n$, and thus $(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}})$ is a witness for $(\boldsymbol{t}, \boldsymbol{t}')$. This witness is checked by evaluating the n polynomials of $\mathcal{B}$.

3.3 Final signature scheme

This final subsection is voluntarily brief, as our signature scheme closely follows the structure of known MPCitH signature schemes [17,6,2] once the underlying hard problem and the corresponding polynomial constraints have been set. The MPC protocol to prove knowledge of the solution to a NIC instance is given in Figure 2, where we represent only one of the N shares. This protocol must be performed for all $i \in \{1..N\}$. The repetition of the MPC protocol ρ times, achieved by using $n \cdot \rho$ challenges, amplifies the soundness from $1/2$ to $1/2^\rho$. Furthermore, the shares of 0 are the same as described in [39, Protocol 7], to guarantee the security of the scheme.

This protocol is then integrated into the MPC-in-the-Head paradigm outlined in Figure 1, using either the TCitH or the VOLEitH frameworks. This results in

Public values: A full-rank matrix $\boldsymbol{G}_0 \in \mathbb{F}_2^{(k-2)\times n}$ and vectors $(\boldsymbol{t}, \boldsymbol{t}') = (\boldsymbol{u} + \boldsymbol{x}\boldsymbol{G}_0, \boldsymbol{v} + \boldsymbol{y}\boldsymbol{G}_0) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$, where $\boldsymbol{u},\ \boldsymbol{v} \in \mathbb{F}_2^n$ is the solution to the (planted) NIC instance.
Secret of the prover: The two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}_2^{k-2}$.

Output: Shares $[\![\alpha^{(l)}]\!]_i$ where each $\alpha^{(l)}$ is in $\mathbb{F}_2$

1. The prover computes shares of 0: $[\![0]\!]_{i,1}, \ldots, [\![0]\!]_{i,l}$ of degree 2;
2. The prover receives coefficients $\gamma_{1,1}, \ldots, \gamma_{n,\rho} \in \mathbb{F}_q$;
3. The prover computes $[\![\boldsymbol{u}]\!]_i = \boldsymbol{t} + [\![\boldsymbol{x}]\!]_i \boldsymbol{G}_0$ and $[\![\boldsymbol{v}]\!]_i = \boldsymbol{t}' + [\![\boldsymbol{y}]\!]_i \boldsymbol{G}_0$;
4. The prover computes $[\![\boldsymbol{c}]\!]_i = [\![\boldsymbol{u}]\!]_i * [\![\boldsymbol{v}]\!]_i$;
5. The prover computes $[\![\alpha^{(l)}]\!]_i = [\![0]\!]_{i,l} + \sum_{j=1}^{n} \gamma_{j,l} [\![\boldsymbol{c}_j]\!]_i$ for all $l \in \{1..\rho\}$;
6. The prover outputs $[\![\alpha^{(1)}]\!]_i, \ldots, [\![\alpha^{(l)}]\!]_i$.

Fig. 2. Polynomial constraints protocol for shares $[\![\boldsymbol{x}]\!]_i$ and $[\![\boldsymbol{y}]\!]_i$ to prove knowledge of the solution to a NIC instance.

signature sizes of:

$$\begin{cases} \mathrm{Size}_{\mathrm{TCitH}} := 4\lambda + \tau \left(\underbrace{2 \cdot (k-2)}_{\Delta\boldsymbol{x}, \Delta\boldsymbol{y}} + \underbrace{\rho}_{[\![\alpha]\!]} + \underbrace{2\lambda}_{\substack{\text{Commitment} \\ \text{Reveal}}} \right) + \underbrace{\lambda \cdot T_{\mathsf{open}}}_{\substack{\text{Randomness} \\ \text{Generation}}} \\ \mathrm{Size}_{\mathrm{VOLEitH}} := 4\lambda + \tau \left(\underbrace{2 \cdot (k-2)}_{\substack{\Delta\boldsymbol{x}, \Delta\boldsymbol{y} \\ \text{and corrections}}} + \underbrace{\rho}_{\substack{[\![\alpha]\!] \\ \text{consistency} \\ \text{check}}} + \underbrace{2\lambda}_{\substack{\text{Commitment} \\ \text{Reveal}}} + \underbrace{(\rho + B)}_{\substack{\text{VOLE} \\ \text{correlations}}} \right) + \underbrace{\lambda \cdot T_{\mathsf{open}}}_{\substack{\text{Randomness} \\ \text{Generation}}} \end{cases},$$

where:

- $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{y}$ are shares of $\boldsymbol{x}$ and $\boldsymbol{y}$ that must be transmitted with more than just a random seed (see [39,15] for more details on the randomness generation). In the case of VOLEitH, corrections of shares are also needed;
- The value τ is the number of repetitions of the proof of knowledge. It is chosen such that $\left(\frac{d}{N}\right)^{\tau} \leq 2^{-\lambda + p_{\mathrm{work}}}$ for TCitH and $\frac{d}{N^{\tau}} \leq 2^{-\lambda + p_{\mathrm{work}}}$ for VOLEitH, where λ is the security parameter;
- The value ρ is the number of repetitions of the MPC protocol inside a single repetition of the proof of knowledge. It is such that $\rho \geq \lambda$. We also require that some $\mu \geq \log_2(N)$ divides ρ;
- The value B is such that $B \cdot \log_2(q) = B \geq 16$.

Security of the scheme. Under the assumption that Problem 3 is hard, the resulting signature scheme is EUF-CMA secure in the random oracle model. Roughly speaking, an adversary must either:

- Solve the NIC instance;

– Or anticipate the coefficients $\gamma_{1,1}, \ldots, \gamma_{n,\rho}$ for all τ rounds;
– Or anticipate which share indices must be revealed;
– Or find a collision in the hash functions.

We refer to [39,2,6,48] for a formal proof of the TCitH framework, and to [15,13,12,1] for analogous proofs in the VOLEitH setting. We stress that all these proofs are independent of the underlying NIC problem. In the rest of the paper, we will study the cryptanalysis of Problem 3 and give the parameters of our signature scheme.

4 Combinatorial Attacks

The first type of attacks on Problem 3 exploits the Hamming weight constraints on the secret vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ by reducing the problem to solving decoding instances in $\mathcal{C}_0$. First, recall from Section 3.1 that the NIC problem can be efficiently solved given knowledge of either $\boldsymbol{u}$, $\boldsymbol{v}$, or $\boldsymbol{u}+\boldsymbol{v}$. Based on this observation, the attacker may try to recover $\boldsymbol{u}$ or $\boldsymbol{v}$ by enumerating all solutions of weight $w = n/3 + \delta$ to the decoding instances $(\boldsymbol{G}_0, \boldsymbol{t})$ or $(\boldsymbol{G}_0, \boldsymbol{t}')$, or to recover $\boldsymbol{u} + \boldsymbol{v}$ by enumerating all solutions of weight $2w$ to the decoding instance $(\boldsymbol{G}_0, \boldsymbol{t} + \boldsymbol{t}')$. From this list of solutions, extracting the valid candidate, or the very small number of valid candidates, can be done efficiently. This leads to the following two algorithms to solve Problem 3.

Algorithm 1 *Let $(\mathcal{C}_0, \boldsymbol{t}, \boldsymbol{t}')$ be an instance of Problem 3 with parameter δ and let $\boldsymbol{G}_0 \in \mathbb{F}_2^{(k-2)\times n}$ be a full-rank generator matrix of $\mathcal{C}_0$. First, call an algorithm that finds all solutions of weight w to the decoding instance $(\boldsymbol{G}_0, \boldsymbol{t})$, and store these solutions in a list L. Then, for each $\boldsymbol{c} \in L$, solve the linear system in the variables $\boldsymbol{y}$ given by*

$$\boldsymbol{c} * (\boldsymbol{t}' + \boldsymbol{y}\boldsymbol{G}_0) = \boldsymbol{0}_n.$$

Finally, accept $\boldsymbol{c}$ for $\boldsymbol{u}$ if and only if this linear system admits a solution. Such a solution is the remaining component of the corresponding NIC solution.

Algorithm 2 *We use the notation of Algorithm 1 and we let $\mathcal{C}$ be the linear code $\mathcal{C}_0$ augmented with $\boldsymbol{t}$ and $\boldsymbol{t}'$. First, call an algorithm that finds all solutions of weight $2w$ to the decoding instance $(\boldsymbol{G}_0, \boldsymbol{t} + \boldsymbol{t}')$, and store these solutions in a list L. Then, for each $\boldsymbol{c} \in L$, form the vector space $(\boldsymbol{c} * \mathcal{C}) \cap \mathcal{C}$ and compute its dimension. Accept $\boldsymbol{c}$ if and only if*

$$\dim\left((\boldsymbol{c} * \mathcal{C}) \cap \mathcal{C}\right) \geq 2.$$

For such a $\boldsymbol{c}$, the solutions to the NIC instance are eventually obtained by enumerating pairs of codewords $(\boldsymbol{e}, \boldsymbol{e}')$ of $(\boldsymbol{c} * \mathcal{C}) \cap \mathcal{C}$, and keeping only those such that $\boldsymbol{e} * \boldsymbol{e}' = \boldsymbol{0}_n$.

The correctness of Algorithm 1 is immediate, while that of Algorithm 2 follows directly from Lemma 4.

We now estimate the complexity of these two algorithms if Information Set Decoding (ISD) [56,16,49,50] is used in the first step. Recall that Problem 3 was constructed so that there exist exponentially many codewords of Hamming weight both $\approx n/3$ and $\approx 2n/3$ in the public code $\mathcal{C}$, and such codewords typically give solutions to the decoding instances we are interested in as $\mathcal{C}_0 \subset \mathcal{C}$ and $\boldsymbol{t},\ \boldsymbol{t}' \in \mathcal{C}$. Using Lemma 2, we expect the number of solutions of weight $w = n/3 + \delta$ (resp. $2w = 2n/3 + 2\delta$) to the decoding instances $(\boldsymbol{G}_0, \boldsymbol{t})$ and $(\boldsymbol{G}_0, \boldsymbol{t}')$ (resp. $(\boldsymbol{G}_0, \boldsymbol{t} + \boldsymbol{t}')$) to be $N_{2,n,w,k-2}$ (resp. $N_{2,n,2w,k-2}$). In this parameter regime, we further expect advanced ISD algorithms [16,49,50] to be able to recover all these solutions in amortized time poly or constant, and thus the complexity of the first step is essentially determined by the number of solutions. Concretely, one can use the estimator [33] to find the value of one individual ISD cost $\mathcal{K}_{\mathsf{ISD}}(2, n, w, k-2)$ (resp. $\mathcal{K}_{\mathsf{ISD}}(2, n, 2w, k-2)$), which is typically very small, and then multiply this value with $N_{2,n,w,k-2}$ (resp. $N_{2,n,2w,k-2}$). By combining this with the cost of the second step, which boils down to dense linear algebra, we obtain the following complexity estimates for Algorithms 1 and 2, respectively.

Proposition 2. *Let $N_{2,n,w,k-2}$ be the average number of solutions of weight $w = n/3+\delta$ to a random decoding instance over $\mathbb{F}_2$ with length n and dimension $k-2$ as defined in Lemma 2, and let $\mathcal{K}_{\mathsf{ISD}}(2, n, w, k-2)$ be the cost of finding one of these solutions using Information Set Decoding. Solving an instance of Problem 3 with constant parameter δ using Algorithm 1 takes time*

$$\mathcal{O}\left(\max(\mathcal{K}_{\mathsf{ISD}}(2, n, w, k-2) \cdot N_{2,n,w,k-2}, (k-2)^{\omega} \cdot N_{2,n,w,k-2})\right) \tag{9}$$

for $k \to \infty$, where $2 \leq \omega < 3$ is the linear algebra constant.

Proof. The first term is the complexity of finding all solutions $\boldsymbol{c}$ of weight w to the decoding instance $(\boldsymbol{G}_0, \boldsymbol{t})$, while the second term corresponds to the cost of the linear algebra step. More precisely, solving one linear system $\boldsymbol{c} * (\boldsymbol{t} + \boldsymbol{y}\boldsymbol{G}_0) = \boldsymbol{0}$ takes $\mathcal{O}((k-2)^{\omega})$ operations, and this step must be performed at most $N_{2,n,w,k-2}$ times, which yields the result. □

Proposition 3. *We use the notation of Proposition 2. Solving an instance of Problem 3 with constant parameter δ using Algorithm 2 takes time*

$$\mathcal{O}\left(\max(\mathcal{K}_{\mathsf{ISD}}(2, n, 2w, k-2) \cdot N_{2,n,2w,k-2}, k^{\omega} \cdot N_{2,n,2w,k-2})\right) \tag{10}$$

for $k \to \infty$, where $2 \leq \omega < 3$ is the linear algebra constant.

Proof. The only difference compared to Proposition 2, apart from the obvious $2w$ instead of w, lies in the second term, which represents the complexity of computing the intersection $(\boldsymbol{c} * \mathcal{C}) \cap \mathcal{C}$ for each candidate $\boldsymbol{c}$ of weight $2w$. Note that the cost of recovering the second part of the solution can typically be neglected, since this computation is only needed for a successful candidate $\boldsymbol{c}$. □

In the parameter regime of interest for our signature scheme, the combinatorial attacks described here seem to be the limiting factor. However, it is relatively

easy to counteract them. Recall that we must choose inverse code rates greater than 4.82 due to Equation (5). Rounding this value up to 5 and starting from $k = 210$ yields $n = 5k = 1050$. Finally, take $\delta = 0$ in Problem 3, so that $w = n/3 = 350$. For such parameters (n, k, w), one can check that both types of decoding instances mentioned above have, on average, 2^{117} solutions. Finding all of them will thus cost $\mathcal{K}_{\mathsf{ISD}} \cdot 2^{117}$, where the factor $\mathcal{K}_{\mathsf{ISD}}$ is relatively small. Concretely, $\mathcal{K}_{\mathsf{ISD}}$ has a value of $\approx 2^{32}$ using the above estimator [33], resulting in a complexity for Algorithm 1 of $\approx 2^{149}$ field operations, which is dominated by the cost of the first step. Algorithm 2 behaves similarly, as there are as many solutions of weight $2w$ as of weight w when $w = n/3$.

5 Algebraic Attacks

We focus here on the attack based on solving the public system $\mathcal{B}$ given in Equation (8) to recover a witness. We refer the reader to Appendix C for a different algebraic approach. The system $\mathcal{B}$ consists of n affine bilinear equations in $k-2$ variables $\boldsymbol{x}$ and $k-2$ variables $\boldsymbol{y}$, so that it is highly overdetermined for the parameters under consideration. We add the field equations of $\mathbb{F}_2$ to the solving process, since this can only lower the complexity of the attack.

After giving the necessary background on solving bilinear systems in the next subsection, we first analyze the behavior of standard Gröbner basis algorithms when applied to our system, and then turn to hybrid approaches. We will see that these hybrid approaches improve upon the plain Gröbner basis computation, even though they remain less efficient than combinatorial attacks.

5.1 Algebraic cryptanalysis of bilinear systems

We refer to [32] for the relevant notions on polynomial systems and Gröbner bases. We focus here on the analysis of Gröbner basis algorithms applied to bilinear equations, where $\mathcal{F}$ denotes an arbitrary affine bilinear system of m equations in n_x variables $\boldsymbol{x}$ and n_y variables $\boldsymbol{y}$ (by a bilinear affine equation, we mean a bilinear equation with additional linear and constant terms).

Case $m \leq n_x + n_y$. The solving of underdetermined bilinear systems was addressed in [35], where it was shown that bilinear systems are not regular in the usual sense. The authors introduce the notion of a bi-regular bilinear system to capture the behavior of generic bilinear systems with respect to Gröbner basis algorithms. For such bi-regular systems, the syzygy module is well understood, and the Hilbert series is known [35, Theorem 5.1]. Building on these results, [35] proposes a modified version of the F5 algorithm [34], extending the F5 criterion to avoid all reductions to zero when applied to bi-regular bilinear systems.

Case $m > n_x + n_y$. In contrast to the rigorous analysis of [35] in the underdetermined case, the overdetermined case has only been partially investigated in subsequent papers [58,55,7,24,54]. These works rely on Macaulay matrices

indexed by bi-degree, as introduced below, and apply algorithms from the XL family [30]. The bi-degree of a polynomial $f \in R = \mathbb{F}_2[\boldsymbol{x}, \boldsymbol{y}]$ is defined as

$$\boldsymbol{d} = (d_x, d_y) = \big(\deg_{\boldsymbol{x}}(f), \deg_{\boldsymbol{y}}(f)\big),$$

where $\deg_{\boldsymbol{x}}(f)$ (resp. $\deg_{\boldsymbol{y}}(f)$) denotes the partial degree of f with respect to the variables $\boldsymbol{x}$ (resp. $\boldsymbol{y}$). We order bi-degrees componentwise, i.e., $\boldsymbol{d}' \leq \boldsymbol{d}$ if and only if $d'_x \leq d_x$ and $d'_y \leq d_y$. For $\boldsymbol{d} \in \mathbb{N}^2$, let $R_{\boldsymbol{d}}$ denote the $\mathbb{F}_2$-vector space of bihomogeneous polynomials in R of bidegree $\boldsymbol{d}$ and let $R_{\leq \boldsymbol{d}} = \oplus_{\boldsymbol{d}' \leq \boldsymbol{d}} R_{\boldsymbol{d}'}$.

Definition 2 (Homogeneous Macaulay matrix in bi-degree $\boldsymbol{d}$). *Let $\mathcal{F} \in R^m$ be a bihomogeneous system and let $\boldsymbol{d} \in \mathbb{N}^2$. The homogeneous Macaulay matrix of $\mathcal{F}$ in bi-degree $\boldsymbol{d}$ is the matrix $\boldsymbol{M}_{\boldsymbol{d}}(\mathcal{F})$ whose columns are indexed by all monomials $\nu \in R_{\boldsymbol{d}}$, rows are indexed by pairs (μ, f) for all monomials $\mu \in R$ and $f \in \mathcal{F}$ such that $\mu f \in R_{\boldsymbol{d}}$, and coefficients are given by coef$(\nu, \mu f)$.*

Definition 3 (Affine Macaulay matrix in bi-degree $\boldsymbol{d}$). *Let $\mathcal{F} \in R^m$ be a polynomial system and let $\boldsymbol{d} \in \mathbb{N}^2$. The affine Macaulay matrix of $\mathcal{F}$ in bi-degree $\boldsymbol{d}$ is the matrix $\boldsymbol{M}_{\leq \boldsymbol{d}}(\mathcal{F})$ whose columns are indexed by all monomials $\nu \in R_{\leq \boldsymbol{d}}$, rows are indexed by pairs (μ, f) for all monomials $\mu \in R$ and $f \in \mathcal{F}$ such that $\mu f \in R_{\leq \boldsymbol{d}}$, and coefficients are given by coef$(\nu, \mu f)$.*

We now recall how XL works on our working example of an overdetermined affine bilinear system $\mathcal{F}$. For a given bi-degree $\boldsymbol{d}$, we want to find a solution to $\mathcal{F}$ or show that no solution exists by solving the linear system

$$\boldsymbol{M}_{\leq \boldsymbol{d}}(\mathcal{F}) \cdot \boldsymbol{u}^{\mathsf{T}} = \boldsymbol{0}.$$

This will succeed as long as the rank of $\boldsymbol{M}_{\leq \boldsymbol{d}}(\mathcal{F})$ is equal to its number of columns when $\mathcal{F}$ has no solution or this number of columns minus a small constant c when a solution exists, typically $c = 1$. We may say that $\boldsymbol{d}$ is a *solving bi-degree*. This approach was studied in [58,7] in the particular case $\boldsymbol{d} = (1, d_y)$ for $d_y \geq 1$ – referred to as $\boldsymbol{y}$-XL – and for arbitrary bi-degrees in [55,21,24]. Finally, we note that a generalization based on linearizing several Macaulay matrices rather than a single one was proposed recently in [54].

The so-called MutantXL (MXL) variant of XL [27] was introduced to solve the system by considering Macaulay matrices in lower degrees compared to standard XL. MXL relies on the notion of *degree fall*.

Definition 4 (Degree fall, informal definition). *A degree fall in bi-degree $\boldsymbol{d}$ for $\mathcal{F}$ is a polynomial of bi-degree $\boldsymbol{d}' < \boldsymbol{d}$ that belongs to the rowspace of $\boldsymbol{M}_{\leq \boldsymbol{d}}(\mathcal{F})$ but not to that of $\boldsymbol{M}_{\leq \boldsymbol{d}'}(\mathcal{F})$.*

In the bilinear setting, the $\boldsymbol{y}$-MXL algorithm of [7] is the adaptation of $\boldsymbol{y}$-XL following this strategy, but one can generalize this to arbitrary bi-degrees. The point of MXL is that there may exist degree falls in bi-degree $\boldsymbol{d}$ even when $\boldsymbol{d}$ is not a solving bi-degree. In this case, the algorithm would append these degree falls to the initial system $\mathcal{F}$, reconstruct Macaulay matrices in smaller bi-degrees,

and continue until either new degree falls are found or a Macaulay matrix of the appropriate rank has been reached. The hope is that the rest of the computation never goes beyond bi-degree $\boldsymbol{d}$.

Unfortunately, the behavior of MXL is very difficult to predict for a given polynomial system. For instance, it remains poorly understood even in the perhaps simpler setting of generic overdetermined bilinear systems (see, for example, the first sentence in [7, Page 16]).

Complexity estimates for XL/MXL. The complexity of these two approaches depends on both the solving degree and the type of linear algebra algorithm used as a subroutine. The solving degree in XL is typically estimated using the notion of the witness degree [11, Definition 2], [7, Definition 26]. In MXL, a more relevant notion is that of the first fall degree, which is the degree at which the first degree falls occur. The key assumption for using this quantity is that the algorithm does not continue at a higher degree afterwards, which remains an assumption. Among the works [58,55,7,24,54], we note that [7] is the only one to explicitly discuss MXL. Further, they indeed assume that their $\boldsymbol{y}$-MXL variant terminates at the so-called $\boldsymbol{y}$-first fall degree $d_{\boldsymbol{y}\text{-ff}}$ [7, Definition 23 & §4.2].

The complexity of XL or MXL, assuming that the dominant step is on an almost square Macaulay matrix in bi-degree $\boldsymbol{d}$ can be loosely estimated as follows. Using sparse linear algebra techniques, namely Block-Wiedemann [59,29] or Block-Lanczos [53], the complexity is given by

$$2(n_x+1)(n_y+1)M_{\boldsymbol{d}}^2, \tag{11}$$

where $M_{\boldsymbol{d}}$ denotes the number of columns of the Macaulay matrix. By relying instead on dense linear algebra to compute row echelon forms, one obtains

$$7M_{\boldsymbol{d}}^{\omega}, \tag{12}$$

where $2 \leq \omega < 3$ is the linear algebra constant. Overall, the final cost is obtained by first taking the smallest complexity between Equation (11) and Equation (12) for a given Macaulay matrix, and then minimizing over the set of Macaulay matrices from which the system can be solved.

Remark 1 (Deriving our estimates) *We will provide optimistic estimates for the complexity of algebraic approaches to solve the system $\mathcal{B}$. We do not believe this is a concern, since the cost of combinatorial attacks is much lower. We were not able to provide more precise estimates, in particular because (i) the overdetermined bilinear case is not well understood, and (ii) the effect of adding the field equations of $\mathbb{F}_2$ is not well understood either. Going further would require tackling these two questions, which are already challenging in the generic case.*

Our estimates were obtained as follows. We restricted ourselves to studying Macaulay matrices in bi-degrees $(d_x, d_y) = (1, d)$. This choice was motivated by the fact that these matrices are the smallest ones for a fixed total degree $d_x + d_y$. Furthermore, the work of [7] applies specifically to such matrices. A conjectured bivariate generating series, as used in [55,21,24], would in principle allow us to

handle more general Macaulay matrices. However, we did not manage to obtain an analogous series in our structured setting. Moreover, even in the generic overdetermined setting, there is no proof of this conjectured series. To study the complexity of $\boldsymbol{y}$-MXL, we have furthermore assumed that the solving degree coincides with the first fall degree, which represents the most optimistic scenario for the attacker. We are forced to rely on such an assumption, as the termination of $\boldsymbol{y}$-MXL is not well understood even in the generic case.

5.2 Bounds on the $\boldsymbol{y}$-first fall degree

As we have just discussed in Remark 1, our complexity estimates call for bounds on the $\boldsymbol{y}$-first fall degree. Unfortunately, we cannot apply the general results of [7, Proposition 24] due to the structure of our system $\mathcal{B}$. Moreover, [7] did not consider the field equations of $\mathbb{F}_2$, which we do in this work. The peculiar structure previously mentioned is that the bilinear part of each equation in $\mathcal{B}$ is equal to $\ell(\boldsymbol{x})\ell(\boldsymbol{y})$, where ℓ is a linear form in $k-2$ variables.

In the following, we may rewrite our system as

$$(1, 0, \boldsymbol{x})\boldsymbol{G} * (0, 1, \boldsymbol{y})\boldsymbol{G} = \boldsymbol{0}_n,$$

where

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{t} \\ \boldsymbol{t}' \\ \boldsymbol{G}_0 \end{bmatrix} \in \mathbb{F}_2^{k \times n}.$$

We also include the field equations of $\mathbb{F}_2$ to construct the Macaulay matrix, which we may denote by $\boldsymbol{M}^{\mathbb{F}_2}_{\leq(1,d)}(\mathcal{B})$. This matrix is obtained as in Definition 3, where now μ ranges over squarefree monomials and the polynomials μf are reduced by the field equations of $\mathbb{F}_2$. If, furthermore, we assume that the generator matrix $\boldsymbol{G}_0$ can be expressed in systematic form $\boldsymbol{G}_0 = \begin{bmatrix} \boldsymbol{I}_{k-2} & \boldsymbol{A} \end{bmatrix}$, where $\boldsymbol{A} \in \mathbb{F}_2^{(k-2)\times(n-k+2)}$, we obtain

$$(1, 0, \boldsymbol{x}) \begin{bmatrix} \boldsymbol{t} \\ \boldsymbol{t}' \\ \boldsymbol{I}_{k-2} \ \boldsymbol{A} \end{bmatrix} * (0, 1, \boldsymbol{y}) \begin{bmatrix} \boldsymbol{t} \\ \boldsymbol{t}' \\ \boldsymbol{I}_{k-2} \ \boldsymbol{A} \end{bmatrix} = \boldsymbol{0}_n.$$

This new expression yields $k-2$ equations whose bilinear part reduces to $x_i y_i$ for $i \in \{1..k-2\}$, and $n-k+2$ denser affine bilinear equations corresponding to the columns of $\boldsymbol{A}$. These $k-2$ first equations can be used as "reductors" by restricting the construction of the Macaulay matrix to the remaining $n-k+2$ equations and by removing the columns corresponding to monomials divisible by some $x_i y_i$. In the following, the ideal generated by the field equations of $\mathbb{F}_2$ will be denoted by

$$I := \langle x_1^2 - x_1, \ldots, x_{k-2}^2 - x_{k-2}, y_1^2 - y_1, \ldots, y_{k-2}^2 - y_{k-2} \rangle.$$

We also define the ideal

$$J := \langle x_1 y_1, \ldots, x_{k-2} y_{k-2} \rangle,$$

that is generated by the bilinear parts of the above reductors. Finally, we let $f_1, \ldots, f_{n-k+2}$ denote the remaining equations of $\mathcal{B}$, and $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{n-k+2}$ the columns of the matrix $\boldsymbol{A}$.

To analyze the $\boldsymbol{y}$-first fall degree, we proceeded as follows. First, we derived a bound $d_{\mathcal{B}}$ on degree falls in the system $\mathcal{B}$ without adding the field equations of $\mathbb{F}_2$. For this, we relied on the same tools as in [35,58,7], namely Jacobian matrices. We obtained the following expression.

Claim 1 *When the field equations of $\mathbb{F}_2$ are not included in the Gröbner basis computation, we expect degree falls for the first time in bi-degree $(1, d_{\mathcal{B}})$, where $d_{\mathcal{B}}$ is defined as*

$$d_{\mathcal{B}} := \min\left\{ d \in \mathbb{Z}_{>0},\ n\binom{k+d-4}{d-1} + \binom{k-2}{d+1} > (k-2)\binom{k+d-3}{d} \right\}. \quad (13)$$

Second, we obtained a bound $d_{\mathcal{B},\mathbb{F}_2}$ on degree falls in $\mathcal{B}$ together with the field equations of $\mathbb{F}_2$ by an ad hoc argument. Indeed, we could not benefit from using Jacobian matrices in the same way, since the field equations are not bilinear.

Claim 2 *When the field equations of $\mathbb{F}_2$ are included in the Gröbner basis computation, we expect degree falls for the first time in bi-degree $(1, d_{\mathcal{B},\mathbb{F}_2})$, where $d_{\mathcal{B},\mathbb{F}_2}$ is defined as*

$$d_{\mathcal{B},\mathbb{F}_2} := \min\left\{ d \in \mathbb{Z}_{>0},\ (n-k+2)\binom{k-3}{d-1} > d\binom{k-2}{d+1} \right\}, \quad (14)$$

as long as this value is smaller than $d_{\mathcal{B}}$.

We refer to Appendix A for how we obtained $d_{\mathcal{B}}$ and to Appendix B for experiments to verify both Claims 1 and 2. We now detail our ad hoc argument that justifies the expression of $d_{\mathcal{B},\mathbb{F}_2}$. Recall that we expect degree falls in bi-degree $(1, d)$ for the system $\mathcal{B}$ together with the field equations of $\mathbb{F}_2$ whenever the rank of the Macaulay matrix $\boldsymbol{M}^{\mathbb{F}_2}_{\leq(1,d)}(\mathcal{B})$ is greater than the number of monomials of bi-degree $(1, d)$ that are both squarefree and not divisible by any $x_i y_i$. The second point follows from the shape of the reductors described above.

To estimate the number of linearly independent rows in the Macaulay matrix, note that the Frobenius syzygies $f_j^2 - f_j = 0 \bmod I$ do not induce linear dependencies in $\boldsymbol{M}^{\mathbb{F}_2}_{\leq(1,d)}(\mathcal{B})$, as they occur in bi-degree $(2, 2)$. However, already in bi-degree $(1, 2)$, we have the following syzygies

$$\ell_j(\boldsymbol{y}) f_j(\boldsymbol{x}, \boldsymbol{y}) = \ell_j(\boldsymbol{x}) \ell_j(\boldsymbol{y})^2 = \ell_j(\boldsymbol{x}) \ell_j(\boldsymbol{y}) = f_j \bmod I,$$

for each $j \in \{1..n-k+2\}$. We can avoid them when constructing the Macaulay matrix. To this end, we augment each f_j by multiplying it with squarefree monomials μ in all variables $\boldsymbol{y}$ except one, which appears in $\ell_j(\boldsymbol{y})$. We expect the remaining rows to be linearly independent, and their number will determine the rank of $\boldsymbol{M}^{\mathbb{F}_2}_{\leq(1,d)}(\mathcal{B})$ up to a certain point. The number of such rows is

$$(n-k+2)\binom{k-3}{d-1}.$$

To estimate the number of monomials present, which yields the number of columns in the Macaulay matrix, observe that one can write

$$\ell_j(\boldsymbol{x})\ell_j(\boldsymbol{y}) = \sum_{1\le s<t\le k-2} a_{j,s}a_{j,t}(x_s y_t + x_t y_s),$$

where $a_{j,s},\ a_{j,t} \in \mathbb{F}_2$ are coefficients in the vector $\boldsymbol{a}_j$. Multiplying by squarefree monomials of degree $d-1$ in $\boldsymbol{y}$ (indexed by subsets $A \subset \{1..k-2\}$ of size $d-1$) and reducing modulo the ideal as explained above to construct $\boldsymbol{M}^{\mathbb{F}_2}_{\le(1,d)}(\mathcal{B})$ produces polynomials of the form

$$\boldsymbol{y}_A\ell_j(\boldsymbol{x})\ell_j(\boldsymbol{y}) \bmod I + J = \sum_{s<t,\ s\notin A,\ t\notin A} a_{j,s}a_{j,t}(x_s\boldsymbol{y}_{A\cup\{t\}} + x_t\boldsymbol{y}_{A\cup\{s\}}),$$

where the number of possible sums $x_s\boldsymbol{y}_{A\cup\{t\}} + x_t\boldsymbol{y}_{A\cup\{s\}}$ is the number of pairs (i, B) such that $B \subset \{1..k-2\},\ \#B = d$ and $i > \min(B)$, hence

$$d\binom{k-2}{d+1}.$$

By combining the observations on the number of rows that we expect to be linearly independent and the number of columns of exact bi-degree $(1, d)$, the condition for degree falls in bi-degree $(1, d)$ is thus

$$(n-k+2)\binom{k-3}{d-1} > d\binom{k-2}{d+1},$$

which corresponds to Equation (14).

Remark 2 (Asymptotic bevahior of $d_{\mathcal{B}}$ and $d_{\mathcal{B},\mathbb{F}_2}$) *For most relevant parameters, we claim that the value of $d_{\mathcal{B},\mathbb{F}_2}$ will be smaller than that of $d_{\mathcal{B}}$, which confirms that adding the field equations of $\mathbb{F}_2$ improves the efficiency of the solving process. To see this, we express both Equations (13) and (14) as lower bounds on d and we study the coarsest one. On the one hand, by using the fact that*

$$\frac{\binom{k-3}{d-1}}{\binom{k-2}{d+1}} = \frac{d(d+1)}{(k-2)(k-d-2)},$$

Equation (14) is equivalent to

$$(n-k+2)\frac{d(d+1)}{(k-2)(k-d-2)} > d \Leftrightarrow (n-k+2)\frac{d+1}{(k-2)(k-d-2)} > 1,$$

as $d > 0$. After simplification, we obtain

$$d > B_{\mathbb{F}_2}(n,k) := \frac{(k-2)^2 - (n-k+2)}{n}.$$

On the other hand, Equation (13) by neglecting the term $\binom{k-2}{d+1}$ is equivalent to

$$d > B(n,k) := \frac{(k-2)(k-3)}{n-k+2}.$$

When $k = Rn$ is linear in n and $n \to +\infty$, we have $B_{\mathbb{F}_2}(n,k) \sim R^2 n$ and $B(n,k) \sim \frac{R^2}{1-R}n$, and we therefore expect that $d_{\mathcal{B},\mathbb{F}_2} < d_{\mathcal{B}}$ in general.

5.3 Hybrid approaches

The so-called hybrid approach [19] is a general solving technique that combines guesses on variables – or, more generally, the addition of linear constraints in these variables – with Gröbner basis algorithms. Unlike previous works on random bilinear systems which use this technique [7,24], we consider structured guesses that fit the particular shape of $\mathcal{B}$, for which it is easier to estimate the probability of a correct guess. Recall that the vector $\boldsymbol{u}$ (resp. $\boldsymbol{v}$) may be viewed as an error of weight $w = n/3+\delta$ when attempting to decode the noisy codeword $\boldsymbol{t} = \boldsymbol{x}\boldsymbol{G}_0 + \boldsymbol{u}$ (resp. $\boldsymbol{t}' = \boldsymbol{y}\boldsymbol{G}_0 + \boldsymbol{v}$) in $\mathcal{C}_0$.

First approach: fixing zeroes at the same positions in both $\boldsymbol{u}$ and $\boldsymbol{v}$. For some integer $0 \leq a \leq k-2$, one picks an arbitrary subset $I \subset \{1..n\}$ of size a and makes the bet that both $\boldsymbol{u}$ and $\boldsymbol{v}$ are zero on these positions, i.e., $\boldsymbol{u}_I = \boldsymbol{v}_I = \boldsymbol{0}$.

Lemma 5. *The success probability of the first approach is*

$$P_{n,w,a} := \frac{\binom{n-2w}{a}}{\binom{n}{a}}.$$

Proof. The vector $\boldsymbol{u}+\boldsymbol{v}$ is of weight $2w$, so that there are $\binom{n-2w}{a}$ subsets $I \subset \{1..n\}$ of size a which do not intersect its support. □

To verify this bet, one is left with solving Problem 1 in the shortened code $\mathcal{S}_I(\mathcal{C}_0)$, whose dimension is expected to be $k-2-a$ with constant probability. In fact, at this point, since we have just reduced the problem to a smaller instance, we could just as well apply a combinatorial algorithm instead of using XL in the second step. If one chooses to use $\boldsymbol{y}$-XL, we denote by $d_{\mathcal{B},a}$ the bound on the $\boldsymbol{y}$-first fall degree in $\mathcal{B}$ specialized accordingly to solve Problem 1 in the shortened code $\mathcal{S}_I(\mathcal{C}_0)$, and by $d_{\mathcal{B},a,\mathbb{F}_2}$ the corresponding bound when the field equations are also included. Both bounds are obtained using the same analysis as in Section 5.2.

Second approach: fixing zeroes in only one of the two vectors $\boldsymbol{u}$ or $\boldsymbol{v}$. For some integer $0 \leq a \leq k-2$, one picks an arbitrary subset $I \subset \{1..n\}$ of size a and makes the bet that either $\boldsymbol{u}$ or $\boldsymbol{v}$ is zero on this subset of positions, let's say $\boldsymbol{v}$.

Lemma 6. *The success probability of the second approach is*

$$Q_{n,w,a} := \frac{\binom{n-w}{a}}{\binom{n}{a}}.$$

Proof. The vector $\boldsymbol{v}$ is of weight w, hence there are $\binom{n-w}{a}$ subsets $I \subset \{1..n\}$ of size a which do not intersect its support. It must be noted that a subset I such that $I \cap \mathrm{supp}(\boldsymbol{u}) = \emptyset$ but $I \cap \mathrm{supp}(\boldsymbol{v}) \neq \emptyset$ is not successful here. □

To verify this bet, one has to solve two correlated decoding instances in $\mathcal{C}_0$ and $\mathcal{S}_I(\mathcal{C}_0)$ respectively – which can be achieved using essentially the same algorithms as those used for the NIC problem. The specialization of $\mathcal{B}$ one can use for this task, denoted by $\mathcal{B}^*$, is as follows. Recalling that the matrix $\boldsymbol{G}_0$ is in systematic form and assuming for example that $I \subset \{1..k-2\}$, the system $\mathcal{B}^*$ is obtained by fixing $\boldsymbol{y}_I = \boldsymbol{t}'_I$ in $\mathcal{B}$. The equations corresponding to columns $j \in \{1..k-2\} \setminus I$ are unchanged compared to $\mathcal{B}$ and retain the same bilinear term $x_i y_i$, while those corresponding to columns $j \in I$ vanish after the specialization. The remaining $n-k+2$ equations are modified, but the bilinear parts are still products between two linear forms in $\boldsymbol{x}$ and in the non-specialized variables $\boldsymbol{y}$. We can thus adapt the reasoning of Section 5.2 to study degree falls in $\mathcal{B}^*$. More precisely, the relevant Macaulay matrix in bi-degree $(1, d)$ contains

$$(n-k+2)\binom{k-3-a}{d-1}$$

rows, and the relevant sums of the above, of the form $x_s y_{A\cup\{t\}} + x_s y_{A\cup\{t\}}$, fall into three categories. Those for which $A \cap I \neq \emptyset$ or $s,\ t \in I$ become zero, those for which $(A \cup \{s,t\}) \cap I = \emptyset$ are unchanged, and those for which $A \cap I = \emptyset$ and exactly one of s or t belongs to I drop to one monomial. Summing the cardinalities of the last two groups yields a number of monomials equal to

$$d\binom{k-2-a}{d+1} + a\binom{k-2-a}{d} = (k-2-d)\binom{k-2-a}{d} - \binom{k-2-a}{d+1}.$$

From this, we can deduce the following claim.

Claim 3 *When the field equations of $\mathbb{F}_2$ are included in the Gröbner basis computation, we expect degree falls for the first time in bi-degree $(1, d'_{\mathcal{B},a,\mathbb{F}_2})$, where $d'_{\mathcal{B},a,\mathbb{F}_2}$ is defined as*

$$d'_{\mathcal{B},a,\mathbb{F}_2} := \min\left\{ d \in \mathbb{Z}_{>0},\ (n-k+2)\binom{k-3-a}{d-1} > d\binom{k-2-a}{d+1} + a\binom{k-2-a}{d} \right\}, \tag{15}$$

as long as this value is smaller than the value $d'_{\mathcal{B},a}$ obtained from the study of Jacobian matrices defined in Claim 4 of Appendix A.

Assuming that we can solve the underlying polynomial system at the first fall degree, we have everything in place to study the complexity of our hybrid approaches. To see why these approaches should not be more threatening than combinatorial techniques in our parameter regime, we consider an optimistic lower bound from the attacker's perspective (note that solving at the first fall degree is already optimistic in this regard):

$$\min(\mathcal{K}_1(n,k,w), \mathcal{K}_2(n,k,w))\,,$$

where the terms $\mathcal{K}_1(n,k,w)$ and $\mathcal{K}_2(n,k,w)$ correspond to the complexities of the two hybrid approaches, obtained as follows. We instantiate Equations (11)

and (12) with the lower bound $d_{\mathcal{B},a,\mathbb{F}_2}$ (resp. $d'_{\mathcal{B},a,\mathbb{F}_2}$) on the $\boldsymbol{y}$-first fall degree in the specialized system by taking the field equations into account, as this value is typically smaller than $d_{\mathcal{B},a}$ (resp. $d'_{\mathcal{B},a}$), and we multiply the result by the inverse of the success probability $P_{n,w,a}$ from Lemma 5 (resp. $Q_{n,w,a}$ from Lemma 6). This yields

$$\mathcal{K}_1(n,k,w) = \min_{0\le a\le k-2} P_{n,w,a}^{-1} \min\Big((k-2-a)^2 M^2_{\mathcal{K}_1,a,(1,d_{\mathcal{B},a,\mathbb{F}_2})},\ 7M^{\omega}_{\mathcal{K}_1,a,(1,d_{\mathcal{B},a,\mathbb{F}_2})}\Big),$$

$$\mathcal{K}_2(n,k,w) = \min_{0\le a\le k-2} Q_{n,w,a}^{-1} \min\Big((k-2)(k-2-a)^2 M^2_{\mathcal{K}_2,a,(1,d'_{\mathcal{B},a,\mathbb{F}_2})},\ 7M^{\omega}_{\mathcal{K}_2,a,(1,d'_{\mathcal{B},a,\mathbb{F}_2})}\Big),$$

where the number of columns can be approximated by the number of monomials of the maximal degree, i.e.,

$$M_{\mathcal{K}_1,a,(1,d)} := d\binom{k-2-a}{d+1} \quad \text{and } M_{\mathcal{K}_2,a,(1,d)} := M_{\mathcal{K}_1,a,(1,d)} + a\binom{k-2-a}{d}.$$

Continuing with the above parameter set $(n,k,w) = (1050, 210, 350)$, we obtain a complexity of more than 300 bits when no hybrid approach is involved, as the first degree falls should occur in bi-degree $(1, 41)$. The first hybrid approach performs better only by a small margin, also with a complexity of roughly 300 bits for certain choices of a. It is not possible to increase the value of a too much, as the success probability $P_{n,w,a}$ drops quite significantly, which makes the resulting cost too high. Finally, the optimum for the second hybrid approach reduces to fully guessing one of the vectors, which is equivalent to performing the combinatorial attack of Algorithm 1.

6 Parameters and Performance

We finish by giving the parameters of the signature scheme along with its associated performance. The parameters of the NIC problem are chosen according to the attacks described in Sections 4 and 5. For all of them, we choose a value of n that is already divisible by 3, and thus $w = n/3$ with $\delta = 0$ is natural. The best attack on these parameters is the combinatorial attack that applies Algorithm 1 (or, equivalently, Algorithm 2, since both have similar complexity). The parameters of the TCitH framework are chosen to match those used in Mirath [2], while the parameters of the VOLEitH framework follow those in [22]. This ensures a fair comparison with these schemes, as they are all defined over $\mathbb{F}_2$ and involve polynomial constraints of degree two as well.

Comparison to other signature schemes. In Table 4, we provide a non-exhaustive list of other signature schemes that can be reasonably compared to ours in terms of public key and signature sizes. We find that our scheme outperforms the MPCitH NIST candidates SDitH and PERK, based on the decoding problem in the Hamming metric and on the Permuted Kernel Problem, respectively, while remaining comparable to the rank metric-based MPCitH candidates RYDE and

Security Level	NIC Parameters			TCitH Parameters				Sizes	
	k	n	w	(N, τ)	T_{open}	ρ	p_{work}	$\lvert\mathsf{pk}\rvert$	$\lvert\sigma\rvert$
NIST I	210	1050	350	$(2^{11}, 12)$	116	132	8	234 B	3 126 B
NIST III	330	1650	550	$(2^{11}, 19)$	176	192	2	354 B	7 246 B
NIST V	450	2250	700	$(2^{11}, 25)$	240	264	6	482 B	13 033 B

Table 2. Parameters for the TCitH instantiation, with resulting public key and signature sizes in Bytes.

Security Level	NIC Parameters			VOLEitH Parameters				Sizes	
	k	n	w	(N, τ)	T_{open}	ρ	p_{work}	$\lvert\mathsf{pk}\rvert$	$\lvert\sigma\rvert$
NIST I	210	1050	350	$(2^{11}, 11)$	99	128	7	234 B	2 962 B
NIST III	330	1650	550	$(2^{12}, 16)$	162	192	12	354 B	6 888 B
NIST V	450	2250	700	$(2^{12}, 21)$	240	256	6	482 B	12 922 B

Table 3. Parameters for the VOLEitH instantiation, with resulting public key and signature sizes in Bytes.

Mirath, when evaluated under the same frameworks. We also obtain a signature larger by less than 1 kB compared to LESS, but with a much smaller public key – as other MPC-in-the-Head schemes do. The larger signature size, compared to both MQOM and the Huth–Joux scheme, can be explained by the fact that both MQ and SBC are associated with smaller witness sizes. Finally, note that our public key is only a few Bytes larger than those of the other MPCitH schemes, and that its size could be further reduced with a more sophisticated key generation procedure. Overall, we believe that Table 4 shows that the NIC problem is competitive in terms of parameter sizes.

Scheme	Assumption	Framework	$\lvert\sigma\rvert$	$\lvert\mathsf{pk}\rvert$	$\lvert\sigma\rvert + \lvert\mathsf{pk}\rvert$
LESS [9]	Code Equivalence	Σ protocol	2 481 B	13 939 B	15 430 B
SDitH [4]	Decoding in the Hamming metric	VOLEitH	3 705 B	70 B	3 775 B
PERK [1]	Permuted Kernel	VOLEitH	3 480 B	100 B	3 580 B
RYDE [6]	Rank Decoding	TCitH	3 115 B	69 B	3 184 B
Mirath [2]	MinRank	TCitH	2 990 B	57 B	3 047 B
MQOM [17]	MQ	TCitH	2 820 B	52 B	2 872 B
$\mathrm{SBC}^{\mathrm{M}}_{\mathrm{VOLE}}$ [42]	SBC	VOLEitH	2 786 B	48 B	2 834 B
This work	**NIC**	**TCitH**	**3 126 B**	**234 B**	**3 360 B**
This work	**NIC**	**VOLEitH**	**2 962 B**	**234 B**	**3 196 B**

Table 4. Comparison with other signature schemes for the NIST security level I.

Performances. As we have not implemented our construction, we choose to report the performance of Mirath in Table 5 to provide an estimate of our scheme's expected performance. In this last paragraph, we explain why such a methodology seems fair by comparing the number of multiplications over $\mathbb{F}_2$ needed for the verification in both cases. We focus on the NIST level I parameter sets, but the same analysis can be done for higher security levels.

- Roughly speaking, Mirath consists in verifying constraints $\boldsymbol{e}\boldsymbol{H}^\top = \boldsymbol{y}$, where $\boldsymbol{e} \in \mathbb{F}_2^{mn}$ is the "flattening" of a matrix $\boldsymbol{E} \in \mathbb{F}_2^{m\times n}$ of rank r, obtained as the product of two matrices $\boldsymbol{S} \in \mathbb{F}_2^{m\times r}$ and $\boldsymbol{C} \in \mathbb{F}_2^{r\times n}$, and where $\boldsymbol{H}$ is a matrix of size $(mn-k)\times mn$. For NIST security level I, we have $m = n = 42$ and $r = 4$. Using naive matrix multiplication, the product $\boldsymbol{S} \times \boldsymbol{C}$ requires roughly
$$2 \times 42 \times 42 \times 4 = 14\ 112 \quad \text{multiplications.}$$
Next, multiplying the vector $\boldsymbol{e}$ of size mn with the binary matrix $\boldsymbol{H}^\top$ of size $mn \times (mn - k)$, where $k = 1443$, amounts to approximately
$$1\ 764 \times 321 = 566\ 244 \quad \text{multiplications,}$$
disregarding the systematic form of $\boldsymbol{H}$.
- In our case, we have $k = 210$ and $n = 1050$. Multiplying the two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ both of size $k - 2$ with the $(k - 2) \times n$ binary matrix $\boldsymbol{G}_0$ requires roughly
$$2 \times 210 \times 1050 = 441\ 000 \quad \text{multiplications,}$$
disregarding the systematic form of $\boldsymbol{G}_0$. Next, we require an additional n multiplications in $\mathbb{F}_2$ to obtain the final star-product of the computed vectors, i.e., 1 050 multiplications.

Overall, we estimate that the performances of our signature scheme should be quite close to (and actually faster than) those in Table 5, as supported by our estimation and by the very similar timings across all MPCitH-based signature schemes.

Instance	Keygen	Sign	Verify
Mirath-1b-short	0.6 M	24 M	18 M
Mirath-3b-short	1.2 M	112 M	96 M
Mirath-5b-short	1.9 M	155 M	132 M

Table 5. Performances of Mirath v2.1.0's optimized implementation for "short" parameters, using parameters with $q = 2$. Timings are given in Millions of Cycles (M). We expect an implementation of our scheme to achieve roughly the same timings.

References

1. Aaraj, N., Bettaieb, S., Bidoux, L., Budroni, A., Dyseryn, V., Esser, A., Gaborit, P., Kulkarni, M., Mateu, V., Palumbi, M., Perin, L., Tillich, J.P.: PERK. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
2. Adj, G., Aragon, N., Barbero, S., Bardet, M., Bellini, E., Bidoux, L., Chi-Domínguez, J.J., Dyseryn, V., Esser, A., Feneuil, T., Gaborit, P., Neveu, R., Rivain, M., Rivera-Zamarripa, L., Sanna, C., Tillich, J.P., Verbel, J., Zweydinger, F.: Mirath. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
3. Adj, G., Rivera-Zamarripa, L., Verbel, J.: MinRank in the Head: Short Signatures from Zero-Knowledge Proofs. In: AFRICACRYPT 2023 (2023)
4. Aguilar-Melchor, C., Bettaieb, S., Bidoux, L., Feneuil, T., Gaborit, P., Gama, N., Gueron, S., Howe, J., Hülsing, A., Joseph, D., Joux, A., Kulkarni, M., Persichetti, E., Randrianarisoa, T.H., Rivain, M., Yue, D.: SD-in-the-Head Signature Scheme, NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
5. Ames, S., Hazay, C., Ishai, Y., Venkitasubramaniam, M.: Ligero: Lightweight Sublinear Arguments Without a Trusted Setup. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (2017)
6. Aragon, N., Bardet, M., Bidoux, L., Chi-Domínguez, J.J., Dyseryn, V., Feneuil, T., Gaborit, P., Joux, A., Neveu, R., Rivain, M., Tillich, J.P., Vincotte, A.: RYDE. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
7. Baena, J.B., Cabarcas, D., Verbel, J.: On the complexity of solving generic overdetermined bilinear systems. Advances in Mathematics of Communications **17**(5), 1210–1234 (2023)
8. Baldi, M., Barenghi, A., Battagliola, M., Bitzer, S., Gianvecchio, M., Karl, P., Manganiello, F., Pavoni, A., Pelosi, G., Pintore, F., Santini, P., Schupp, J., Signorini, E., Slaughter, F., Wachter-Zeh, A., Weger, V.: CROSS: Codes and Restricted Objects Signature Scheme. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
9. Baldi, M., Barenghi, A., Beckwith, L., Biasse, J.F., Chou, T., Esser, A., Gaj, K., Karl, P., Mohajerani, K., Pelosi, G., Persichetti, E., Saarinen, M.J., Santini, P., Wallace, R., Zweydinger, F.: LESS : Linear Equivalence Signature Scheme. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
10. Bardet, M., Bros, M., Cabarcas, D., Gaborit, P., Perlner, R., Smith-Tone, D., Tillich, J.P., Verbel, J.: Improvements of Algebraic Attacks for Solving the Rank Decoding and MinRank Problems. In: ASIACRYPT 2020 (2020)
11. Bardet, M., Faugère, J.C., Salvy, B., Spaenlehauer, P.J.: On the Complexity of Solving Quadratic Boolean Systems. Journal of Complexity **29**(1), 53–75 (Feb 2013)
12. Baum, C., Beullens, W., Braun, L., Delpech de Saint Guilhem, C., Klooß, M., Majenz, C., Mukherjee, S., Orsini, E., Ramacher, S., Rechberger, C., Roy, L., Scholl, P.: Shorter, Tighter, FAESTer: Optimizations and Improved (QROM) Analysis for VOLE-in-the-Head Signatures. In: CRYPTO 2025 (2025)
13. Baum, C., Beullens, W., Braun, L., de Saint Guilhem, C.D., Klooß, M., Majenz, C., Mukherjee, S., Orsini, E., Ramacher, S., Rechberger, C., Roy, L., Scholl, P.:

FAEST. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
14. Baum, C., Beullens, W., Mukherjee, S., Orsini, E., Ramacher, S., Rechberger, C., Roy, L., Scholl, P.: One Tree to Rule Them All: Optimizing GGM Trees and OWFs for Post-Quantum Signatures. In: ASIACRYPT 2024 (2024)
15. Baum, C., Braun, L., de Saint Guilhem, C.D., Klooß, M., Orsini, E., Roy, L., Scholl, P.: Publicly Verifiable Zero-Knowledge and Post-Quantum Signatures from VOLE-in-the-Head. In: CRYPTO 2023 (2023)
16. Becker, A., Joux, A., May, A., Meurer, A.: Decoding Random Binary Linear Codes in 2n/20: How $1+1=0$ Improves Information Set Decoding. In: EUROCRYPT 2012 (2012)
17. Benadjila, R., Bouillaguet, C., Feneuil, T., Rivain, M.: MQ on my Mind: Post-Quantum Signatures from the Non-Structured Multivariate Quadratic Problem. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 2) (2024)
18. Bettaieb, S., Bidoux, L., Gaborit, P., Kulkarni, M.: Modelings for generic PoK and applications: Shorter SD and PKP based signatures. Cryptology ePrint Archive, Paper 2024/1668 (2024), https://eprint.iacr.org/2024/1668
19. Bettale, L., Faugère, J.C., Perret, L.: Hybrid approach for solving multivariate systems over finite fields. Journal of Mathematical Cryptology **3**(3), 177–197 (2010)
20. Bettale, L., Kahrobaei, D., Perret, L., Verbel, J.: Biscuit: New MPCitH Signature Scheme from Structured Multivariate Polynomials. In: ACNS 2024 (2024)
21. Beullens, W.: Improved Cryptanalysis of UOV and Rainbow. In: EUROCRYPT 2021 (2021)
22. Bidoux, L., Feneuil, T., Gaborit, P., Neveu, R., Rivain, M.: Dual Support Decomposition in the Head: Shorter Signatures from Rank SD and MinRank. In: ASIACRYPT 2024 (2025)
23. Borello, M., Schmid, W., Scotti, M.: The geometry of intersecting codes and applications to additive combinatorics and factorization theory. Journal of Combinatorial Theory, Series A **214**, 106023 (2025)
24. Bouillaguet, C., Feneuil, T., Maire, J., Rivain, M., Sauvage, J., Vergnaud, D.: Multivariate Commitments and Signatures with Efficient Protocols. Cryptology ePrint Archive, Paper 2025/2035 (2025), https://eprint.iacr.org/2025/2035
25. Bouillaguet, C., Sauvage, J.: Preliminary Cryptanalysis of the Biscuit Signature Scheme. IACR Communications in Cryptology **1**(1) (2024)
26. Briaud, P., Lemoine, A., Randriambololona, H., Tillich, J.P.: A heuristic subexponential attack on the McEliece cryptosystem. Cryptology ePrint Archive, Paper 2026/1232 (2026), https://eprint.iacr.org/2026/1232
27. Buchmann, J.A., Ding, J., Mohamed, M.S.E., Mohamed, W.S.A.E.: MutantXL: Solving Multivariate Polynomial Equations for Cryptanalysis. In: Symmetric Cryptography. Dagstuhl Seminar Proceedings (DagSemProc), vol. 9031, pp. 1–7 (2009)
28. Cohen, G., Zémor, G.: Intersecting codes and independent families. IEEE Transactions on Information Theory **40**(6), 1872–1881 (1994)
29. Coppersmith, D.: Solving homogeneous linear equations over GF (2) via block Wiedemann algorithm. Mathematics of Computation **62**, 333–350 (1994)
30. Courtois, N., Klimov, A., Patarin, J., Shamir, A.: Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In: EUROCRYPT 2000 (2000)
31. Couvreur, A., Mora, R., Tillich, J.P.: A New Approach Based on Quadratic Forms to Attack the McEliece Cryptosystem. In: ASIACRYPT 2023 (2023)

32. Cox, D.A., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms. Springer International Publishing (2015). https://doi.org/10.1007/978-3-319-16721-3, https://doi.org/10.1007/978-3-319-16721-3
33. Esser, A., Bellini, E.: Syndrome Decoding Estimator. In: PKC 2022 (2022)
34. Faugère, J.C.: A New Efficient Algorithm for Computing GröBner Bases without Reduction to Zero (F5). In: Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation. p. 75–83. ISSAC '02 (2002)
35. Faugère, J.C., Safey El Din, M., Spaenlehauer, P.J.: Gröbner bases of bihomogeneous ideals generated by polynomials of bidegree (1,1): Algorithms and complexity. Journal of Symbolic Computation **46**(4), 406–437 (2011)
36. Feneuil, T.: Building MPCitH-based signatures from MQ, MinRank, Rank SD and PKP. In: International Conference on Applied Cryptography and Network Security (ACNS) (2024)
37. Feneuil, T., Joux, A., Rivain, M.: Syndrome Decoding in the Head: Shorter Signatures from Zero-Knowledge Proofs. In: CRYPTO 2022 (2022)
38. Feneuil, T., Rivain, M.: Threshold Linear Secret Sharing to the Rescue of MPC-in-the-Head. In: ASIACRYPT 2023 (2023)
39. Feneuil, T., Rivain, M.: Threshold Computation in the Head: Improved Framework for Post-Quantum Signatures and Zero-Knowledge Arguments. Journal of Cryptology (2025)
40. Fiat, A., Shamir, A.: How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In: CRYPTO 1986 (1986)
41. Huth, J., Joux, A.: MPC in the Head Using the Subfield Bilinear Collision Problem. In: CRYPTO 2024 (2024)
42. Huth, J., Joux, A.: VOLE-in-the-Head Signatures from Subfield Bilinear Collisions. In: ASIACRYPT 2025 (2025)
43. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-Knowledge Proofs from Secure Multiparty Computation. SIAM Journal on Computing **39**(3), 1121–1152 (2009)
44. Kales, D., Zaverucha, G.: Efficient Lifting for Shorter Zero-Knowledge Proofs and Post-Quantum Signatures. Cryptology ePrint Archive, Paper 2022/588 (2022), https://eprint.iacr.org/2022/588
45. Katona, G.O.H., Srivastava, J.N.: Minimal 2-coverings of a finite affine space based on GF(2). Journal of Statistical Planning and Inference **8**, 375–388 (1983)
46. Katz, J., Kolesnikov, V., Wang, X.: Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (2018)
47. Kim, S., Lee, B., Son, M.: Relaxed Vector Commitment for Shorter Signatures. In: EUROCRYPT 2025 (2025)
48. Kosuge, H., Xagawa, K.: Towards formal security proofs of MQOM. Cryptology ePrint Archive, Paper 2026/629 (2026), https://eprint.iacr.org/2026/629
49. May, A., Meurer, A., Thomae, E.: Decoding Random Linear Codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In: ASIACRYPT 2011 (2011)
50. May, A., Ozerov, I.: On Computing Nearest Neighbors with Applications to Decoding of Binary Linear Codes. In: EUROCRYPT 2015 (2015)
51. McEliece R. J.: A public-key cryptosystem based on algebraic coding theory. DSN Progress Report (1978)
52. Miklós, D.: Linear binary codes with intersection properties. Discrete Applied Mathematics **9**(2), 187–196 (1984)
53. Montgomery, P.L.: A Block Lanczos Algorithm for Finding Dependencies over GF(2). In: EUROCRYPT 1995 (1995)

54. Ning, K.C., Ran, L., Samardjiska, S.: Multi-homogeneous XL. Cryptology ePrint Archive, Paper 2025/2060 (2025), https://eprint.iacr.org/2025/2060
55. Perlner, R., Smith-Tone, D.: Rainbow Band Separation is Better than we Thought. Cryptology ePrint Archive, Paper 2020/702 (2020), https://eprint.iacr.org/2020/702
56. Prange, E.: The use of information sets in decoding cyclic codes. IRE Transactions on Information Theory **8**(5), 5–9 (1962)
57. Delpech de Saint Guilhem, C., Orsini, E., Tanguy, T.: Limbo: Efficient Zero-knowledge MPCitH-based Arguments. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (2021)
58. Verbel, J., Baena, J., Cabarcas, D., Perlner, R., Smith-Tone, D.: On the Complexity of "Superdetermined" Minrank Instances. In: PQCrypto 2019 (2019)
59. Wiedemann, D.: Solving sparse linear equations over finite fields. IEEE Transactions on Information Theory **32**(1), 54–62 (1986)

A Jacobian Syzygies

In this appendix, we determine the smallest bi-degree $(1, d_{\mathcal{B}})$ at which degree falls occur in the solving of $\mathcal{B}$ when the field equations are not included, as defined in Claim 1. In practice, the expected first fall degree is the minimum between $d_{\mathcal{B}}$ and the value $d_{\mathcal{B},\mathbb{F}_2}$ defined in Claim 2. We focus on non-trivial syzygies for the system formed by the bilinear components of the bilinear affine equations. These syzygies generate lower-degree equations in the solving algorithm. When they are not redundant – something we cannot control directly – they give rise to degree falls in the sense of Definition 4.

Syzygies from Jacobian matrices. In all [35,58,7,54], a key concept for identifying syzygies is the Jacobian matrix taken with respect to a specific block of variables. For a system of polynomials $\mathcal{F} = (f_1, \ldots, f_m) \in R^m$, the Jacobian matrix of $\mathcal{F}$ with respect to $\boldsymbol{x}$ is the matrix defined by

$$\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_{n_x}} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_{n_x}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_{n_x}} \end{bmatrix}.$$

Given a bihomogeneous system $\mathcal{F}$ consisting of bihomogeneous equations of bi-degree $(1, D)$ in $\boldsymbol{x}$ and $\boldsymbol{y}$, the entries of $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F})$ are homogeneous polynomials in $\mathbb{F}_q[\boldsymbol{y}]_D$. By Euler's homogeneous function theorem, one has

$$\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}) \cdot \boldsymbol{x}^{\mathsf{T}} = \begin{bmatrix} f_1 \\ \vdots \\ f_m \end{bmatrix}.$$

Proposition 4 is a simple consequence of this. This result is stated in [58, Propositions 1 & 2] in the particular case $D = 1$.

Proposition 4. *Let $\mathcal{F} \in R^m$ be a bihomogeneous system of bi-degree $(1, D)$. The vector $(h_1, \ldots, h_m) \in \mathbb{F}_q[\boldsymbol{y}]^m$ is a syzygy for $\mathcal{F}$ if and only if it belongs to the left kernel of $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F})$. If this is the case, this syzygy is nontrivial.*

When $D = 1$ and $m \geq n_x + 1$, [35] combined Proposition 4 with Cramer's rule to find $\binom{m}{n_x+1}$ syzygies of bi-degree $(1, n_x)$ in the left kernel of $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F})$. One can obtain a similar statement for the other Jacobian matrix $\mathrm{Jac}_{\boldsymbol{y}}(\mathcal{F})$. When $m \geq n_x + n_y$, it has been shown in [7] that the kernel of the Jacobian matrix contains vectors of a lower degree than in the underdetermined setting. We describe the case of $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F})$. The fact that a formal vector $(v_1, \ldots, v_m) \in (\mathbb{F}_q[\boldsymbol{y}]_{d-1})^m$ belongs to $\ker(\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}))$ can be expressed as a linear system of $n_x\binom{n_y+d-1}{d}$ equations in $m\binom{n_y+d-2}{d-1}$ variables, namely the coefficients of the monomials $\mu \in \mathbb{F}_q[\boldsymbol{y}]_{d-1}$ in the v_i's. From this observation and assuming that the linear equations are independent, [7] deduce that there will be syzygies in bi-degree $(1, d)$ whenever

$$m\binom{n_y + d - 2}{d - 1} > n_x\binom{n_y + d - 1}{d}. \tag{16}$$

Degree falls in $\mathcal{B}$. Coming back to our system $\mathcal{B}$, we use the fact that the bilinear parts of the equations are given by the star-product $\boldsymbol{x}\boldsymbol{G}_0 * \boldsymbol{y}\boldsymbol{G}_0$. Let $\mathcal{F}$ be the system that contains these bilinear parts. If we further assume that

$$\boldsymbol{G}_0 = \begin{bmatrix} \boldsymbol{I}_{k-2} & \boldsymbol{A} \end{bmatrix}, \quad \text{where } \boldsymbol{A}^\mathsf{T} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{n-k+2,1} \\ a_{2,1} & a_{2,2} & \cdots & a_{n-k+2,2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-k+2,1} & a_{n-k+2,2} & \cdots & a_{n-k+2,k-2} \end{bmatrix} \in \mathbb{F}_q^{(n-k+2)\times(k-2)},$$

then the expression of the Jacobian matrix with respect to $\boldsymbol{x}$ is

$$\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}) = \begin{bmatrix} y_1 & 0 & \cdots & 0 \\ 0 & y_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & y_{k-2} \\ a_{1,1}\ell_1(\boldsymbol{y}) & a_{1,2}\ell_1(\boldsymbol{y}) & \cdots & a_{1,k-2}\ell_1(\boldsymbol{y}) \\ a_{2,1}\ell_2(\boldsymbol{y}) & a_{2,2}\ell_2(\boldsymbol{y}) & \cdots & a_{2,k-2}\ell_2(\boldsymbol{y}) \\ \vdots & \vdots & \vdots & \vdots \\ a_{n-k+2,1}\ell_{n-k+2}(\boldsymbol{y}) & a_{n-k+2,2}\ell_{n-k+2}(\boldsymbol{y}) & \cdots & a_{n-k+2,k-2}\ell_{n-k+2}(\boldsymbol{y}) \end{bmatrix}.$$

For $d \geq 1$, let $\mathcal{F}_d$ be the system $\mathcal{F}$ augmented in bi-degree $(1, d)$. We observe that a syzygy in bi-degree $(1, d)$ for $\mathcal{F}$ is simply a linear relation over $\mathbb{F}_q$ among the equations of $\mathcal{F}_d$. To find such a relation, we look for $\mathbb{F}_q$-dependencies among the rows of $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}_d)$. It is convenient to sort the rows so that the equations

$x_i y_i$ are augmented first. With such a row ordering, the matrix $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}_2)$ can be written as:

$$\begin{bmatrix}
y_1^2 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
y_1 y_{k-2} & 0 & \cdots & 0 \\
0 & y_1 y_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & y_{k-2}^2 \\
a_{1,1}\ell_1(\boldsymbol{y})y_1 & a_{1,2}\ell_1(\boldsymbol{y})y_1 & \cdots & a_{1,k-2}\ell_1(\boldsymbol{y})x_1 \\
\vdots & \vdots & \vdots & \vdots \\
a_{1,1}\ell_1(\boldsymbol{y})y_{k-2} & a_{1,2}\ell_1(\boldsymbol{y})y_{k-2} & \cdots & a_{1,k-2}\ell_1(\boldsymbol{y})y_{k-2} \\
\vdots & \vdots & \vdots & \vdots \\
a_{n-k+2,1}\ell_{n-k+2}(\boldsymbol{y})y_{k-2} & a_{n-k+2,2}\ell_{n-k+2}(\boldsymbol{y})y_{k-2} & \cdots & a_{n-k+2,k-2}\ell_{n-k+2}(\boldsymbol{y})y_k
\end{bmatrix}.$$

Coming back to the general case, let $\boldsymbol{u}$ be a formal row vector of length equal to the number of rows of $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}_d)$. We are interested in the linear system on its coefficients that enforces

$$\boldsymbol{u} \cdot \mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}_d) = \boldsymbol{0}.$$

We may write $\boldsymbol{u}$ as $\boldsymbol{u} = (\boldsymbol{v}, \boldsymbol{w})$, where the entries of $\boldsymbol{v}$ are formal coefficients v_μ indexed by monomials $\mu \in \mathbb{F}_q[\boldsymbol{y}]_d$ and the entries of $\boldsymbol{w}$ are formal coefficients $\boldsymbol{w}_{\nu,i}$ indexed by monomials $\nu \in \mathbb{F}_q[\boldsymbol{y}]_{d-1}$ and $i \in \{1..n-k+2\}$. For $j \in \{1..k-2\}$, let $\boldsymbol{m}_j^\mathsf{T}$ denote the j-th column of $\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}_d)$, so that $\langle \boldsymbol{u}, \boldsymbol{m}_j \rangle \in \mathbb{F}_q[\boldsymbol{y}]_d$ is the j-th component of $\boldsymbol{u} \cdot \mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}_d)$. For any multiset $I = \{i_1 \leq i_2 \leq \cdots \leq i_d\}$ of size d in $\{1..k-2\}$, the coefficient of the monomial $\boldsymbol{y}_I$ in $\langle \boldsymbol{u}, \boldsymbol{m}_j \rangle$ is given by

$$[\boldsymbol{y}_I]\left(\langle \boldsymbol{u}, \boldsymbol{m}_j \rangle\right) = \delta_{\min(I),j} v_{\boldsymbol{y}_I} + \sum_{s=1}^{n-k+2} a_{u,j} \left(\sum_{i \in I} a_{s,i} w_{\boldsymbol{y}_I / y_i, s} \right), \tag{17}$$

where $\delta_{i,j}$ is the Kronecker symbol. First, if $j \in I$ and $j \neq i_1 = \min(I)$, then the equality

$$[\boldsymbol{y}_I](\langle \boldsymbol{u}, \boldsymbol{m}_j \rangle) = [\boldsymbol{y}_{(I \setminus \{i_1\}) \cup \{j\}}](\langle \boldsymbol{u}, \boldsymbol{m}_{i_1} \rangle)$$

expresses the equation attached to (j, I) as an equation attached to (j', I') with either $j' \in I'$, $j' = \min(I')$ or $j' \notin I'$. Second, we may decide to use the equations $[\boldsymbol{y}_I](\langle \boldsymbol{u}, \boldsymbol{m}_{\min(I)} \rangle)$ to consume the degree of freedom $v_{\boldsymbol{y}_I}$. In this way, the number of variables in the linear system is the number of coefficients in $\boldsymbol{w}$, namely

$$(n-k+2)\binom{k-2+d-2}{d-1} = (n-k+2)\binom{k+d-4}{d-1},$$

and the number of linearly independent equations is at most the number of pairs (j, I) such that $j \notin I$, i.e.,

$$(k-2)\binom{k+d-3}{d} - (k-2)\binom{k+d-4}{d-1}.$$

Thus, there are more variables than equations whenever

$$(n-k+2)\binom{k+d-4}{d-1} > (k-2)\binom{k+d-3}{d} - (k-2)\binom{k+d-4}{d-1}$$
$$\Leftrightarrow n\binom{k+d-4}{d-1} > (k-2)\binom{k+d-3}{d}.$$

In characteristic 2, there are additional relations between the equations described above. Indeed, for any subset $J \subset \{1..k-2\}$ of size $d+1$ without repetitions, we observe using Equation (17):

$$\sum_{j\in J}[\boldsymbol{y}_{J\setminus\{j\}}](\langle \boldsymbol{u}, \boldsymbol{C}_j\rangle) = 0. \tag{18}$$

This gives $\binom{k-2}{d+1}$ expected relations, so that the new condition for syzygies in bi-degree $(1,d)$ is

$$n\binom{k+d-4}{d-1} + \binom{k-2}{d+1} > (k-2)\binom{k+d-3}{d}.$$

Returning to the affine parts $\mathcal{B}$, this condition directly corresponds to Claim 1.

Remark 3 *Note that these estimates for $\mathcal{B}$ match those for a generic overdetermined bilinear system in odd characteristic with the appropriate number of equations and variables, namely Equation (16) with $m = n$ and $n_x = n_y = k-2$, and that, in characteristic 2, they differ by a small term $\binom{k-2}{d+1}$.*

Degree falls in $\mathcal{B}^*$ from the second hybrid approach. We focus on the system $\mathcal{B}$ where we have fixed $\boldsymbol{y}_I = \boldsymbol{t}'_I$ for a subset $I \subset \{k-2\}$ of size a, and we let $J = \{1..k-2\} \setminus I$. The bilinear part of the bottom $n-k+2$ equations for $i \in \{1..n-k+2\}$ can be written as

$$\ell_{i,J}(\boldsymbol{x}_J)\ell_{i,J}(\boldsymbol{y}_J) + \ell_{i,I}(\boldsymbol{x}_I)\ell_{i,J}(\boldsymbol{y}_J),$$

where we have used the notation $\ell_i(\boldsymbol{x}) = \ell_{i,I}(\boldsymbol{x}_I) + \ell_{i,J}(\boldsymbol{x}_J)$. We construct the Jacobian matrix with respect to $\boldsymbol{x}$ by taking the partial derivative with respect to x_j for $j \in J$ in the leftmost columns and $j \in I$ in the rightmost columns, and we order the rows so that the equations with bilinear part $x_i y_i$ for $i \in J$ correspond to the top rows. In this way, we obtain

$$\mathrm{Jac}_{\boldsymbol{x}}(\mathcal{F}^*) = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{0}_{(k-2-a)\times a} \\ \boldsymbol{A}_2 & \boldsymbol{B} \end{bmatrix}.$$

The matrix

$$\begin{bmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \end{bmatrix} \in \mathbb{F}_q[\boldsymbol{y}_J]^{(n-a)\times(k-2-a)}$$

is the Jacobian matrix of the equations $x_j y_j$ for $j \in J$, and of $\ell_{i,J}(\boldsymbol{x}_J)\ell_{i,J}(\boldsymbol{y}_J)$ for $i \in \{1..n-k+2\}$, with respect to $\boldsymbol{x}_J$. It therefore has the same structure as

in the plain case. The non-zero block $\boldsymbol{B}$ on the right-hand side is the Jacobian matrix of the equations $\ell_{i,I}(\boldsymbol{x}_I)\ell_{i,J}(\boldsymbol{y}_J)$ for $i \in \{1..n-k+2\}$, with respect to $\boldsymbol{x}_I$. Its entries depend both on the coefficients of the $\ell_{i,J}$'s and those of the $\ell_{i,I}$'s.

We can keep the analysis of above and assume that the columns corresponding to the block $\boldsymbol{B}$ add "fresh" linearly independent equations. This gives a total number of equations equal to

$$(k-2-a)\tbinom{k-2-a+d-1}{d} - (k-2-a)\tbinom{k-2-a+d-2}{d-1} + a\tbinom{k-2-a+d-1}{d}.$$

In characteristic 2, we lose an additional $\binom{k-a-2}{d+1}$ equations due to the linear relations arising from (18) for the part $\begin{bmatrix}\boldsymbol{A}_1\\ \boldsymbol{A}_2\end{bmatrix}$ of the Jacobian matrix. The number of variables remains

$$(n-k+2)\binom{k-2-a+d-2}{d-1} = (n-k+2)\binom{k-a+d-4}{d-1}.$$

By putting everything together, we obtain the following claim on the $\boldsymbol{y}$-first fall degree in $\mathcal{B}^*$.

Claim 4 *When the field equations of $\mathbb{F}_2$ are not included in the computation, we expect degree falls for the first time in bi-degree $(1, d'_{\mathcal{B},a})$ in $\mathcal{B}^*$, where $d'_{\mathcal{B},a}$ is defined as*

$$d'_{\mathcal{B},a} = \min\Bigg\{ d \in \mathbb{Z}_{>0},\ (n-a)\binom{k-a+d-4}{d-1} + \binom{k-a-2}{d+1} > (k-2)\binom{k-a+d-3}{d}\Bigg\}.$$

B Experiments on Degree Falls

We constructed Macaulay matrices associated to a system $\mathcal{B}$ over $\mathbb{F}_2$ with the field equations of $\mathbb{F}_2$ added in order to verify Claim 2, and associated to a system $\mathcal{B}$ over a larger field of characteristic 2 in order to verify Claim 1. In both cases, we verified the claims by studying the row space of the resulting matrices. Our experiments were performed in Magma.

Table 6 reports the $\boldsymbol{y}$-first fall degree we observe when we add the field equations of $\mathbb{F}_2$. For the parameters tested in this table, the $\boldsymbol{y}$-first fall degree that is expected from our analysis is $d_{\mathcal{B},\mathbb{F}_2}$, as $d_{\mathcal{B},\mathbb{F}_2} \leq d_{\mathcal{B}}$. For very overdetermined systems $\mathcal{B}$, we actually found a very small number of degree falls at a degree *lower* than $(1, d_{\mathcal{B},\mathbb{F}_2})$. This is indicated in the column "Falls $<(1, d_{\mathcal{B},\mathbb{F}_2})$". We believe that such degree falls are due to coefficient cancellations, which are more likely to occur over $\mathbb{F}_2$ than over larger fields. Indeed, their number depends on the instance, and we have not observed them over larger fields or for less overdetermined systems.

k	n	$d_{\mathcal{B},\mathbb{F}_2}$	Falls $< (1, d_{\mathcal{B},\mathbb{F}_2})$
17	73	3	$4\times : \emptyset$, $2\times : 1$ in $(1,1)$, $3\times : 2$ in $(1,2)$, $1\times : 3$ in $(1,2)$
18	78	3	$6\times : \emptyset$, $4\times : 1$ in $(1,2)$
19	82	3	$8\times : \emptyset$, $1\times : 2$ in $(1,2)$, $1\times : 4$ in $(1,2)$
20	87	3	$8\times : \emptyset$, $2\times : 1$ in $(1,2)$
21	92	4	$2\times : 4$ in $(1,3)$, $4\times : 2$ in (1,3), $1\times : 2$ in $(1,2)$, $3\times : 1$ in $(1,2)$
23	112	4	$2\times : 7$ in $(1,3)$, $1\times : 6$ in $(1,3)$, $4\times : 3$ in $(1,3)$, $1\times : 2$ in $(1,3)$, $1\times : 1$ in $(1,3)$, $1\times : 1$ in $(1,2)$

Table 6. Number of linearly independent degree falls modulo the field equations of $\mathbb{F}_2$ found at a degree lower than $(1, d_{\mathcal{B},\mathbb{F}_2})$, for 10 trials. More precisely, the expression "$x\times : t$ in $(1, d_y)$" indicates that, over x out of 10 trials, we found t linearly independent degree fall polynomials in bi-degree $(1, d_y)$.

Table 7 corresponds to systems over $\mathbb{F}_8$, for which the $\boldsymbol{y}$-first degree fall is predicted by Claim 1. Contrary to the Boolean case, the Macaulay matrices in bi-degree $(1, d)$ for $d < d_{\mathcal{B}}$ always had the expected rank for the parameters we tested. Moreover, the *homogeneous* Macaulay matrix in bi-degree $(1, d_{\mathcal{B}})$ had its rank equal to the number of expected monomials.

k	n	$d_{\mathcal{B}}$	$\boldsymbol{y}$-first fall degree
17	73	4	4
18	78	4	4
19	82	4	4
20	87	4	4

Table 7. Observed $\boldsymbol{y}$-first fall degree obtained by computing row echelon forms of Macaulay matrices $\boldsymbol{M}_{\leq(1,d)}(\mathcal{B})$.

Finally, it is interesting, in order to test the validity of Claim 1, to find parameters (n, k) for which the value of $d_{\mathcal{B}}$ differs from the estimate of [7], which does not take the characteristic of the field into account. For $n = 36$ and $k = 10$, we have $d_{\mathcal{B}} = 2$, while the estimate of [7] predicts degree falls in bi-degree $(1, 3)$. In our tests, we indeed observed degree falls already in bi-degree $(1, 2)$ for these parameters.

C Another Algebraic Approach

We present an alternative algebraic attack based on a different interpretation of Problem 1. We leave it as an open problem to design more efficient attacks on

NIC or to develop more efficient MPC protocols for this problem based on this alternative interpretation. This interpretation is inspired by the approach of [31] to attack the McEliece cryptosystem [51].

Given an arbitrary basis $V := (\boldsymbol{g}_1, \ldots, \boldsymbol{g}_k)$ of the public code $\mathcal{C}$, one may consider the linear code

$$\mathcal{C}_{\mathrm{rel}}(V) := \left\{ (c_{i,j})_{1 \le i \le j \le k} \in \mathbb{F}_2^{\binom{k+1}{2}}, \sum_{1 \le i \le j \le k} c_{i,j} \left(\boldsymbol{g}_i * \boldsymbol{g}_j \right) = 0 \right\} \subseteq \mathbb{F}_2^{\binom{k+1}{2}}.$$

For an element $(c_{i,j})_{1 \le i \le j \le k} \in \mathbb{F}_2^{\binom{k+1}{2}}$, let $\boldsymbol{M_c} = (m_{i,j}) \in \mathbb{F}_2^{k \times k}$ be the skew-symmetric matrix defined by $m_{i,j} = m_{j,i} = c_{i,j}$ for $1 \le i < j \le k$ and $m_{i,i} = 2c_{i,i} = 0$ for $1 \le i \le k$. The matrix code associated to $\mathcal{C}_{\mathrm{rel}}(V)$ is then defined as

$$\mathcal{C}_{\mathrm{mat}}(V) := \{ \boldsymbol{M_c},\ \boldsymbol{c} \in \mathcal{C}_{\mathrm{rel}}(V) \} \subset \mathbb{F}_2^{k \times k}.$$

The point is that such a matrix code contains a rank-2 matrix regardless of the choice of V. To see this, let us write the non-intersecting vectors as $\boldsymbol{u} = \boldsymbol{x}\boldsymbol{G}$ and $\boldsymbol{v} = \boldsymbol{y}\boldsymbol{G}$, so that

$$\boldsymbol{u} * \boldsymbol{v} = \left(\sum_{i=1}^{k} x_i \boldsymbol{g}_i \right) * \left(\sum_{i=1}^{k} y_i \boldsymbol{g}_i \right) = \boldsymbol{0}_n.$$

This equality can be rewritten as

$$\sum_{i<j} (x_i y_j + x_j y_i)(\boldsymbol{g}_i * \boldsymbol{g}_j) + \sum_{i=1}^{k} x_i y_i \boldsymbol{g}_i = \boldsymbol{0}_n,$$

where we use the fact that $\boldsymbol{g}_i * \boldsymbol{g}_i = \boldsymbol{g}_i$ over $\mathbb{F}_2$. Thus, the matrix $\boldsymbol{M_c} = \boldsymbol{x}^\mathsf{T}\boldsymbol{y} + \boldsymbol{y}^\mathsf{T}\boldsymbol{x}$ is indeed a rank-2 matrix in $\mathcal{C}_{\mathrm{rel}}(\boldsymbol{g})$.

To use this observation for an attack, we conversely need to explain how to solve NIC given the knowledge of a rank-2 matrix in $\mathcal{C}_{\mathrm{rel}}(V)$. We can assume a matrix of the same form $\boldsymbol{M} = \boldsymbol{s}^\mathsf{T}\boldsymbol{t} + \boldsymbol{t}^\mathsf{T}\boldsymbol{s}$, which provides a general relation

$$\begin{aligned} \sum_{i<j} (s_i t_j + s_j t_i)(\boldsymbol{g}_i * \boldsymbol{g}_j) + \sum_{i=1}^{k} c_{i,i} \boldsymbol{g}_i = \boldsymbol{0}_n \\ = \underbrace{\left(\sum_{i=1}^{k} s_i \boldsymbol{g}_i \right)}_{:=\boldsymbol{c}_1} * \underbrace{\left(\sum_{i=1}^{k} t_i \boldsymbol{g}_i \right)}_{:=\boldsymbol{c}_2} + \underbrace{\sum_{i=1}^{k} (c_{i,i} - x_i y_i) \boldsymbol{g}_i}_{:=\boldsymbol{c}_3} = \boldsymbol{0}_n, \end{aligned}$$

for some $c_{i,i} \in \mathbb{F}_2$. In other words, such a relation provides three arbitrary codewords $\boldsymbol{c}_1\ \boldsymbol{c}_2,\ \boldsymbol{c}_3 \in \mathcal{C}$ such that $\boldsymbol{c}_1 * \boldsymbol{c}_2 = \boldsymbol{c}_3$. This information is already sufficient to recover a NIC solution, since

$$(\boldsymbol{c}_1 + \boldsymbol{c}_2) * \boldsymbol{c}_3 = \boldsymbol{c}_1^2 * \boldsymbol{c}_2 + \boldsymbol{c}_2^2 * \boldsymbol{c}_1 = \boldsymbol{c}_1 * \boldsymbol{c}_2 + \boldsymbol{c}_1 * \boldsymbol{c}_2 = \boldsymbol{0}_n.$$

The announced algebraic approach is thus an algebraic approach for solving the MinRank problem given by a basis of $\mathcal{C}_{\mathrm{mat}}(V)$, with target rank 2. We have just explained how to reconstruct the NIC solution from a solution to this MinRank problem. To solve MinRank in this case, one could use the Pfaffian modeling suggested in [31, §5.1], or any algebraic modeling of MinRank – for example, Support-Minors [10] – combined with quadratic constraints to enforce that the matrix being searched for is skew-symmetric. Another approach would be to exploit the parametrization $\boldsymbol{M} = \boldsymbol{s}^{\mathsf{T}}\boldsymbol{t} + \boldsymbol{t}^{\mathsf{T}}\boldsymbol{s}$ for such a rank-2 matrix, which might be preferable in our case. This last point was also observed in the McEliece setting [26].