Evaluation of Code-based Signature Schemes

Partha Sarathi Roy[1], Kirill Morozov[2], Kazuhide Fukushima[1], and Shinsaku Kiyomoto[1]

[1] Information Security Laboratory, KDDI Research, Inc., Japan
`{pa-roy,ka-fukushima,kiyomoto}@kddi-research.jp`
[2] Department of Computer Science and Engineering, University of North Texas, USA
`Kirill.Morozov@unt.edu`

Abstract. Code-based cryptographic schemes recently raised to prominence as quantum-safe alternatives to the currently employed number-theoretic constructions, which do not resist quantum attacks. In this article, we discuss the Courtois-Finiasz-Sendrier signature scheme and derive code-based signature schemes using the Fiat-Shamir transformation from code-based zero-knowledge identification schemes, namely the Stern scheme, the Jain-Krenn-Pietrzak-Tentes scheme, and the Cayrel-Veron-El Yousfi scheme. We analyze the security of these code-based signature schemes and derive the security parameters to achieve the 80-bit and 128-bit level of classical security. To derive the secure parameters, we have studied the hardness of *Syndrome Decoding Problem.* Furthermore, we implement the signature schemes, based on the Fiat-Shamir transform, which were mentioned above, and compare their performance on a PC.

Keywords: post-quantum cryptography · code-based cryptography · signature scheme · identification scheme.

1 Introduction

Digital signature scheme is an important primitive in the arsenal of public-key cryptography. Security of the schemes, already used in practice, relies on the number-theoretic hardness assumptions. Unfortunately, these problems are known to be solvable in polynomial time on quantum computers using Shor's algorithm [33]. Hence, quantum computers would be able to break popular cryptosystems such as RSA or ElGamal (including its elliptic-curve variant) in polynomial time. Given these circumstances, it is important to consider the transition to digital signature schemes which are secure in the post-quantum era. Code-based digital signature schemes are the one of such promising alternatives.

The journey of code-based public-key cryptography started with the work by McEliece [25]. McEliece proposed a public-key encryption scheme based on the hardness of the general decoding problem (see Section 2). In 1986, Niederreiter [28] proposed another code-based public-key encryption scheme, which can

be seen as a dual version of the McEliece construction. Security of the Neiderrieter encryption scheme is based on the hardness of the syndrome decoding problem (see Section 2). When both McEliece and Niederreiter schemes are seen as functions, one can observe that their outputs are quite sparse in the respective domains. For instance, decodable syndromes that represent the Niederreiter ciphertexts normally constitute a negligible fraction of the binary strings of the corresponding length. Therefore, constructing the FDH-type signature (a la RSA signature), from the Niederreiter PKE is not a straightforward task. The first successful construction of such code-based signature scheme, in 2001, by Courtois et al. [11] is based on the Niederreiter public-key encryption scheme with particular parameters. Specifically, high-rate codes are used, which makes the signature feasible via rejection sampling. It is worth noting that apart from the Courtois et al. scheme, secure code-based signature proposals are mainly represented by the constructions obtained from code-based identification schemes via the Fiat-Shamir transform. Hence, the motivation for this work is to study the practical performance of these prospective signature schemes.
Organization: Recalling the basic definitions and notions in Section 2, we will discuss various code-based signature schemes in Section 3. We will study their secure parameters in Section 4. In Section 5, we will describe the implementation results of the Stern signature scheme, the Jain-Krenn-Pietrzak-Tentes signature scheme, and the Cayrel-Veron-El Yousfi signature scheme and conclude in Section 6.

2 Preliminaries

2.1 Notations

We will use the following notations throughout the article.

- $\{0,1\}^*$: bit string of arbitrary length.
- $\mathbb{F}_q$: Galois field of q elements.
- $\mathbb{F}_q^n$: Vector with n elements over $\mathbb{F}_q$.
- $\mathbb{F}_q^{m\times n}$: Matrix with m rows and n columns whose elements are in $\mathbb{F}_q$.
- $\mathsf{wt}(c)$: Hamming weight of string c, i.e., the number of non-zero positions of the string.
- $\|$: Concatenation of strings.
- S^T: Transposition of the matrix S.
- S_n: Set of permutations over n-length string.
- $[A, B]$: An interactive protocol between the parties A and B. For simplicity, we will also use this notation for the transcript of the corresponding protocol. Then, $[A, B] = 1$ will denote the fact that the protocol is accepted by B.

2.2 Zero-knowledge Identification Scheme

An identification scheme consists of three probabilistic, polynomial-time algorithms (G, P, V) such that:

- The randomized key generation algorithm G takes as input the security parameter 1^λ. It outputs a pair of keys $(\mathsf{pk}, \mathsf{sk})$, where pk is called the public key and sk is called the private key. We assume the security parameter is implicit in both pk and sk.
- P and V are the probabilistic algorithms. The prover algorithm P takes as input a private key sk and the verification algorithm V takes as input a public key pk. At the conclusion of the protocol, V outputs 1 or $\perp$.

The following properties hold:

Completeness: $[P(\mathsf{sk}), V(\mathsf{pk})] = 1$.
Honest V always accepts honest P.
Soundness: $\Pr([P^*, V(\mathsf{pk})] = 1) = \mathsf{negl}(\lambda)$.
Cheating P^* (not knowing sk) is rejected with overwhelming probability.
Zero-knowledge: $[P(\mathsf{sk}), V^*(\mathsf{pk})] \approx [Sim, V^*(\mathsf{pk})]$, where Sim is the simulator.
Cheating V^* learns nothing about sk.
For details of the identification schemes, see [22].

2.3 Digital Signature

A *digital signature scheme* Σ = (**Key Generation**, **Signature Generation**, **Signature Verification**) consists of three algorithms.

Key Generation: The key generation algorithm takes a security parameter 1^λ and outputs a pair of keys $(\mathsf{pk}, \mathsf{sk})$.
Signature Generation: The signature generation algorithm takes a message m and a private key sk as inputs and outputs a signature σ on the message m.
Signature Verification: The signature verification algorithm takes as input a public key pk, a message m and a signature σ, and outputs a bit denoting accept or reject, respectively.

The standard security notion for a signature scheme is *existential unforgeability against chosen message attack* (EUF-CMA) and *strongly existential unforgeability against chosen message attack* (SEUF-CMA): The forger gets a public key from a challenger who generates a key pair $(\mathsf{pk}, \mathsf{sk})$. The forger can query a signing oracle on polynomially many messages m_i hereby obtaining signatures σ_i.

We say that the forger wins the EUF-CMA game, if the forger successfully outputs a pair (m^*, σ^*), where σ^* is a valid signature of a message m^* under the private key with the restriction that m^* has never appeared in the query phase.

We say that the forger wins the SEUF-CMA game, if the forger successfully outputs a pair (m^*, σ^*), where σ^* is a valid signature of a message m^* under the private key with the restriction that (m^*, σ^*) has never appeared in the query phase. For more details on digital signatures, see, e.g., [22].

2.4 Security Assumptions

An $[n, k]$ linear code $\mathcal{C}$ is a subspace of dimension k of the vector space $\mathbb{F}_q^n$. A linear code can be described by its parity-check matrix H. The parity-check matrix describes the code as follows:

$$x \in \mathcal{C} \Leftrightarrow Hx^T = 0 \quad (\forall x \in \mathbb{F}_q^n).$$

The product Hx^T is known as the *syndrome* of the vector x.

Definition 1. *Gilbert-Varshamov Bound:*
Let $\mathcal{C}$ be an $[n, k]$ linear code over $\mathbb{F}_q$. The Gilbert-Varshamov (GV) Distance is the largest integer ω such that

$$\frac{k}{n} \geq 1 - H_q\left(\frac{\omega}{n}\right) - \epsilon, \textit{where} \quad 0 < \epsilon \leq 1 - H_q\left(\frac{\omega}{n}\right).$$

Here, H_q is the q-ary entropy function defined as follows:

$$H_q(x) = x\log_q(q-1) - x\log_q x - (1-x)\log_q(1-x).$$

Now, we describe the main hard problems on which the security of code-based signature schemes, presented in the paper, relies.

Definition 2. *Syndrome Decoding Problem (SDP) [4]:*

Input: *a matrix $H \in \mathbb{F}_2^{(n-k)\times n}$, a positive integer ω, and a vector $s \in \mathbb{F}_2^{n-k}$.*
Output: *a codeword x such that $\mathsf{wt}(x) = \omega'$ where $0 < \omega' \leq \omega$ and $Hx^T = s$.*

This problem is NP-complete [8], which means that at least some instances of the problem are hard. However, it is a common belief that it is hard on the average (for well-chosen parameter ranges), which means that random instances are hard. It is easy to show that, with overwhelming probability (taken over the choice of a random code), there exists a unique solution to SDP if the weight ω is below the GV Bound.

An extension of SDP over arbitrary finite field is as follows:

Definition 3. *q-ary Syndrome Decoding Problem (q-SDP) [4]:*

Input: *a matrix $H \in \mathbb{F}_q^{(n-k)\times n}$, a non negative integer ω and a vector $s \in \mathbb{F}_q^{n-k}$.*
Output: *a codeword x such that of $\mathsf{wt}(x) = \omega'$ where $0 < \omega' \leq \omega$ and $Hx^T = s$.*

q-SDP is proven to be NP-hard by S. Barg [5].

Definition 4. *Permuted Goppa Syndrome Decoding Problem (PGSDP) [26]:*

Input: *An $(n-k) \times n$ parity check matrix H for a binary irreducible Goppa code G capable of correcting up to ω errors, a random $(n-k)\times(n-k)$ binary non-singular matrix S and a random $n \times n$ permutation matrix P.*
Output: *A vector $x \in \mathbb{F}_2^n$ of Hamming weight at most ω, such that $H^{pub}x^T = s$, where $H^{pub} = SHP$.*

Definition 5. *General Decoding Problem (GDP):*

Input: *a matrix* $G \in \mathbb{F}_2^{k \times n}$*, a possitive integer* ω*, and a vector* $y \in \mathbb{F}_2^n$*.*
Output: *a pair* $(m, e) \in \mathbb{F}_2^k \times \mathbb{F}_2^n$ *such that* $\mathsf{wt}(e) = \omega'$ *where* $0 < \omega' \leq \omega$ *and* $mG \oplus e = y$*.*

We observe that this problem was reformulated by Jain et al. in [20] as the Exact Learning Parity with Noise (xLPN) problem. Its decisional version is as follows:

Definition 6. *Exact Learning Parity with Noise (xLPN) [20]:*

Input: *a matrix* $A \in \mathbb{F}_2^{k \times n}$*,* $s \in \mathbb{F}_2^k$*,* $e \in \{0,1\}^n$*, such that* $\mathsf{wt}(e) = \omega$*.*
Decide: *if a given* n*-bit vector is* $sA \oplus e$ *or a random one.*

3 Signature Schemes

During the last thirty years, there were several proposals of signature schemes based on linear codes. The initial attempts [1,19,37] have no security reduction to any hard problem. The first successful construction featuring a security reduction was by Courtois et al. [11]. Currently, there are two major directions in the literature, both invoking the random oracle model: the FDH-like scheme by Courtois, Finiasz, and Sendrier [11] (we will refer to it as the CFS signature) and the signatures based on identification schemes using the Fiat-Shamir (FS) transform (see, e.g., [2,32]). Dallot [12] modified the CFS signature scheme (we will refer to it as the mCFS signature), where a randomly chosen value will be concatenated with the message rather than an accumulative counter used in the original CFS signature scheme. The mCFS scheme was proven existentially unforgeable under chosen message attack (EUF-CMA) by Dallot [12] under hardness of the Goppa-Parametrized Bounded Decoding (GPBD) problem and the Goppa-Code Distinguishing (GD) problem. Even with the existence of a distinguisher [16] for the Goppa codes of high rate, a simple modification can provide SEUF-CMA security for the CFS signature [26]. However, from the practical perspective, signing time of mCFS signature is somewhat high due to the difficulty of finding decodable syndromes. A variant of CFS was proposed by Barreto et al. [6] with an aim to reduce the key size. However, the *key recovery* attack [17] against it devised. One can construct signature schemes using Fiat-Shamir transformation [30] on zero-knowledge identification schemes by Stern [35], Jain et al. [20] and Cayrel et al. [10]. Note that Jain et al. [20] pointed out a flaw in the proof of zero-knowledge property of Veron's code-based identification scheme [36], and then they provided an alternative scheme which is indeed ZK. As a security assumption, Jain et al. used the so-called Exact-LPN (xLPN) problem [20], which is, in fact, identical to the general decoding problem considered both in Veron's paper, and in the paper by Roy et al. [32].[3] We need to use the extended version

[3] Jain et al. presented the ZK identification scheme based on the standard LPN problem as well, but it had the soundness error 4/5, while that based on xLPN had the soundness error 2/3. Hence, the former scheme would result in signatures of larger size.

Algorithm 1: Signature Generation in mCFS signature scheme

```
  Input: Private key $\mathsf{sk} = (U, H_0, P)$, message $m$, and the system parameters
  Output: Signature $Sig$
1 $r_i \leftarrow_\$ \{0,1\}^{n-k}$;
2 Compute $x = \text{Decode}_{H_0}(U^{-1}h(M\|r_i))$;
3 If no $x$ was found, then go to Step 1;
4 $Sig = (r_i, xP)$;
5 return $Sig$;
```

of Fiat-Shamir transformation [3] to derive a signature scheme from Cayrel et al.'s scheme [10]. Kabatianskii et al. [21] proposed signature schemes based on random error-correcting codes (we will refer to them as the KKS signature). It was shown by Otmani et al. [29] that the KKS-like signatures are not secure, even when used only once (i.e., as one-time signatures).

In this section, we will describe the constructions of mCFS [12,26] and the signatures based on Stern's [35], Jain et al.'s [20] and Cayrel et al.'s [10] identification schemes. For simplicity, we will call them Stern's, Jain et al.'s, and Cayrel et al.'s signature, respectively.

3.1 mCFS Signature Scheme

System Parameters The mCFS signature scheme uses the following system parameters:

- Positive integer n (length of the code),
- Positive integer k such that $k < n$ (dimension of the code),
- Set $\omega = (n-k)/\log_2 n$,
- Random oracle: $h : \{0,1\}^* \to \mathbb{F}_2^{n-k}$.

Key Generation The key generation algorithm outputs the pair of the private key sk and public key pk.

1. Parity check matrix $H_0 \in \mathbb{F}_2^{(n-k)\times n}$ of an (n,k) binary Goppa code $\mathcal{C}_0$ decoding ω errors.
2. Non-singular matrix $U \in \mathbb{F}_2^{(n-k)\times(n-k)}$ sampled randomly.
3. Permutation matrix $P \in \mathbb{F}_2^{n\times n}$ sampled randomly.
4. Calculate $H = UH_0P$.
5. Output private key $\mathsf{sk} = (U, H_0, P)$ and public key $\mathsf{pk} = H$.

Signature Generation The signature generation algorithm takes as input private key sk and a message m, and output Sig. The detailed algorithm is described in Algorithm 1.

Signature Verification The signature verification algorithm takes as input public key pk, message m, *Sig*. Compute $s = Hx^T$, $s' = h(M \| r_i)$, then output 1 if the following respective equations hold:

$$\begin{cases} \text{Check } s = s' \\ \text{Check } \mathsf{wt}(s) \le \omega \end{cases},$$

or $\perp$ otherwise.

3.2 Stern's Signature Scheme

System Parameters The Stern signature scheme uses the following system parameters:

– Positive integer n (length of the code),
– Positive integer k such that $k < n$ (dimension of the code),
– Positive integer ω (minimum distance of the code),
– Matrix $H \in \mathbb{F}_2^{(n-k)\times n}$ sampled randomly.
– Random oracle: $h : \{0,1\}^* \to \{0,1\}^*$.
– Random oracle: $\mathcal{O} : \{0,1\}^* \to \{0,1,2\}$.

Key Generation The key generation algorithm outputs the pair of the private key sk and public key pk.

1. Sample a vector $s \in \mathbb{F}_2^n$ such that $\mathsf{wt}(s) = \omega$.
2. Calculate a vector $y \in \mathbb{F}_2^{n-k}$ as $y = Hs^T$.
3. Output private key $\mathsf{sk} = s$ and public key $\mathsf{pk} = y$.

Signature Generation The signature generation algorithm takes as input private key sk and a message m, and output *Sig*. The detailed algorithm is described in Algorithm 2.

Signature Verification The signature verification takes as input public key pk, message m, and *Sig*. Compute $b_i \leftarrow \mathcal{O}(m \| c_i)$ and output 1 if the following respective equation holds for all $1 \le i \le \delta$:

$$\begin{cases} \text{Check } c_{i,0} = h(\sigma_i \| Hu_i) \text{ and } c_{i,1} = \sigma_i(u_i). & (b_i = 0) \\ \text{Check } c_{i,0} = h(\sigma_i \| H(u_i \oplus s) \oplus y) \text{ and} & \\ \quad c_{i,2} = h(\sigma_i(u_i \oplus s)). & (b_i = 1) \\ \text{Check} c_{i,1} = \sigma_i(u_i), c_{i,2} = h(\sigma_i(u_i) \oplus \sigma_i(s)), & \\ \quad \text{and } \mathsf{wt}(\sigma_i(s)) = \omega. & (b_i = 2) \end{cases},$$

or $\perp$ otherwise.

Algorithm 2: Signature Generation in Stern's signature scheme

```
Input: Private key $\mathsf{sk} = s$, Message $m$, and System parameters
Output: Signature $Sig$
for $i \leftarrow 0$ to $\delta - 1$ do
    $u_i \leftarrow_\$ \mathbb{F}_2^n$;
    $\sigma_i \leftarrow_\$ S_n$;
    $c_{i,0} \leftarrow h(\sigma_i \| Hu_i)$;
    $c_{i,1} \leftarrow \sigma_i(u_i)$;
    $c_{i,2} \leftarrow h(\sigma_i(u_i \oplus s))$;
    $c_i = (c_{i,0} \| c_{i,1} \| c_{i,2})$;
    $b_i = \mathcal{O}(m \| c_i)$;
    $rsp_i \leftarrow \begin{cases} \sigma_i \| u_i & (b_i = 0) \\ \sigma_i \| (u_i \oplus s) & (b_i = 1) \\ \sigma_i(u_i) \| \sigma_i(s) & (b_i = 2) \end{cases}$;
    $sig_i = c_i \| rsp_i$;
end
$Sig \leftarrow sig_0 \| sig_1 \| \cdots \| sig_{\delta-1}$;
return $Sig$;
```

3.3 Jain et al.'s Signature Scheme

In this section, we will derive the signature scheme from the identification scheme of [20]. We will present a slightly modified design [32]. Here, commitment function is implemented by collision-resistant hash function.

System Parameters

- Positive integer n (length of the code),
- Positive integer k such that $k < n$ (dimension of the code).
- Positive integer ω (minimum distance of the code).
- Matrix $A \in \mathbb{F}_2^{k \times n}$ sampled randomly.
- Radom oracle: $h : \{0,1\}^* \to \{0,1\}^*$.
- Random oracle: $\mathcal{O} : \{0,1\}^* \to \{0,1,2\}$.

Key Generation The key generation algorithm outputs the pair of the private key sk and public key pk.

1. Sample $(s, e) \leftarrow_\$ \mathbb{F}_2^k \times \mathbb{F}_2^n$ such that $\mathsf{wt}(e) = \omega$.
2. Calculate $y = sA \oplus e$.
3. Output private key $\mathsf{sk} = e$ and public key $\mathsf{pk} = y$.

Signature Generation The signature generation algorithm takes private key sk and a message m, output Sig, and system parameters as inputs. Algorithm 3 describes the detailed algorithm.

Algorithm 3: Signature Generation in Jain et al.'s signature scheme

```
   Input: Private key $\mathsf{sk} = e$, Message $m$, and System parameters
   Output: Signature $Sig$
1  for $i \leftarrow 0$ to $\delta - 1$ do
2      $u_i \leftarrow_{\$} \mathbb{F}_2^n$;
3      $v_i \leftarrow_{\$} \mathbb{F}_2^k$;
4      $\sigma_i \leftarrow_{\$} S_n$;
5      $y_{i,0} \leftarrow v_i A \oplus u_i$;
6      $c_{i,0} \leftarrow h(\sigma_i \| y_{i,0})$;
7      $y_{i,1} \leftarrow \sigma_i(u_i)$;
8      $c_{i,1} \leftarrow h(y_{i,1})$;
9      $y_{i,2} \leftarrow \sigma_i(u_i \oplus e)$;
10     $c_{i,2} \leftarrow h(y_{i,2})$;
11     $c_i \leftarrow c_{i,0} \| c_{i,1} \| c_{i,2}$;
12     $b_i \leftarrow \mathcal{O}(m \| c_i)$;
13     $rsp_i \leftarrow \begin{cases} \sigma_i \| y_{i,0} \| y_{i,1} & (b_i = 0) \\ \sigma_i \| y_{i,0} \| y_{i,2} & (b_i = 1) \\ y_{i,1} \| y_{i,2} & (b_i = 2) \end{cases}$;
14     $sig_i \leftarrow c_i \| rsp_i$;
15 end
16 $Sig \leftarrow sig_0 \| sig_1 \| \cdots \| sig_{\delta-1}$;
17 return $Sig$;
```

Signature Verification The signature verification algorithm takes as input public key pk, message m, Sig, and system parameters. It computes $b_i \leftarrow \mathcal{O}(m \| c_i)$ and outputs 1 if the following respective equation holds for all $0 \leq i \leq \delta - 1$:

$$\begin{cases} \text{Check } c_{i,0} \leftarrow h(\sigma_i \| y_{i,0}), & \\ \quad c_{i,1} \leftarrow h(y_{i,1}), & (b_i = 0) \\ \quad y_{i,0} \oplus \sigma_i^{-1}(y_{i,1}) = xA, \text{for some } x \text{ and } \sigma_i \in S_n & \\ \text{Check } c_{i,0} \leftarrow h(\sigma_i \| y_{i,0}), & \\ \quad c_{i,2} \leftarrow h(y_{i,2}), & (b_i = 1) \\ \quad \text{and } y_{i,0} \oplus \sigma_i^{-1}(y_{i,2}) \oplus y = xA, \text{for some } x & \\ \text{Check } c_{i,1} \leftarrow h(y_{i,1}), & \\ \quad c_{i,2} \leftarrow h(y_{i,2}), & (b_i = 2) \\ \quad \text{and } \mathsf{wt}(y_{1,i} \oplus y_{2,i}) = \omega & \end{cases},$$

or $\perp$ otherwise.

3.4 Cayrel et al.'s Signature Scheme

To present the Cayrel et al., signature scheme, first we introduce a special transformation that will be used in the scheme.

Definition 7. *[10] Let $\sigma \in S_n$ and $\gamma = (\gamma_1, \cdots, \gamma_n) \in (\mathbb{F}_q^*)^n$ such that $\gamma_i \neq 0$ for all i. The transformation $\Pi_{\gamma,\sigma}$ is defined as follows:*

$$\begin{aligned}\Pi_{\gamma,\sigma} :& \mathbb{F}_q^n \to \mathbb{F}_q^n \\ & v \mapsto \left(\gamma_{\sigma[0]} v_{\sigma[0]}, \gamma_{\sigma[1]} v_{\sigma[1]}, \cdots, \gamma_{\sigma[n-1]} v_{\sigma[n-1]}\right).\end{aligned}$$

Notice that this transformation is linear transformation, and satisfies $\Pi_{\gamma,\sigma}(v + w) = \Pi_{\gamma,\sigma}(v) + \Pi_{\gamma,\sigma}(w)$ and $\Pi_{\gamma,\sigma}(\alpha v) = \alpha \Pi_{\gamma,\sigma}(v)$ for all $v, w, \alpha \in \mathbb{F}_q$. Furthermore, the transformation preserves the Hamming weight of the vector.

Now, we are in the state to present the signature scheme:

System Parameters The signature scheme uses the following system parameters:

- Positive integer n (length of codeword),
- Positive integer k such that $k < n$ (dimension of the code),
- Positive integer ω (minimum distance of the code),
- Matrix $H \in \mathbb{F}_q^{(n-k)\times n}$ sampled randomly,
- Random oracle $h : \{0,1\}^* \to \{0,1\}^*$,
- Random oracle $\mathcal{O}_1 : \{0,1\}^* \to \mathbb{F}_q^n$,
- Random oracle $\mathcal{O}_2 : \{0,1\}^* \to \{0,1\}$.

Key Generation The key generation algorithm outputs the pair of the private key s and public key y.

1. Sample a vector $s \in \mathbb{F}_q^n$ such that $\mathsf{wt}(s) = \omega$.
2. Calculate a vector $y \in \mathbb{F}_q^{n-k}$ as $y = Hs^T$.
3. Output private key s and public key y.

Signature Generation The signature generation algorithm takes as input private key s and a message m, and output a signature *Sig*. The detailed algorithm is described in Algorithm 4.

Signature Verification The signature verification algorithm takes public key y, message m, and signature Sig as inputs. It computes $\alpha_i \leftarrow \mathcal{O}_1(m\|c_i)$ and $b_i \leftarrow \mathcal{O}_2(m\|c_i\|\alpha_i\|\beta_i)$; then, it outputs 1 if the following respective equation holds for all $0 \le i \le \delta - 1$:

$$\begin{cases}\text{Check } c_{i,0} = h(\sigma_i\|\gamma_i\|H\Pi^{-1}_{\gamma_i,\sigma_i}(\beta_i) - \alpha_i y) & (b_i = 0) \\ \text{Check } c_{i,1} = h(\beta_i - \alpha_i \Pi_{\gamma_i,\sigma_i}(s)\|\Pi_{\gamma_i,\sigma_i}(s)) & \\ \quad \text{and } \mathsf{wt}(\Pi_{\gamma,\sigma}(s)) = \omega & (b_i = 1)\end{cases},$$

or $\perp$ otherwise.

Algorithm 4: Signature Generation in Cayrel et al.'s signature scheme

```
Input: Private key $\mathsf{sk} = s$, Message $m$, and System parameters
Output: Signature $Sig$
for $i \leftarrow 0$ to $\delta - 1$ do
    $u_i \leftarrow_{\$} \mathbb{F}_q^n$;
    $\sigma_i \leftarrow_{\$} S_n$;
    $\gamma_i \leftarrow_{\$} (\mathbb{F}_q \setminus \{0\})^n$;
    $c_{i,0} \leftarrow h(\sigma_i \| \gamma_i \| H u_i)$;
    $c_{i,1} \leftarrow h(\Pi_{\gamma_i,\sigma_i}(u_i) \| \Pi_{\gamma_i,\sigma_i}(s))$;
    $c_i \leftarrow c_{i,0} \| c_{i,1}$;
    $\alpha_i \leftarrow \mathcal{O}_1(m \| c_i)$;
    $\beta_i \leftarrow \Pi_{\gamma_i,\sigma_i}(u_i + \alpha_i s)$;
    $b_i \leftarrow \mathcal{O}_2(m \| c_i \| \alpha_i \| \beta_i)$;
    $rsp_i \leftarrow \begin{cases} \sigma_i \| \gamma_i & (b_i = 0) \\ \Pi_{\gamma_i,\sigma_i}(s) & (b_i = 1) \end{cases}$;
    $sig_i = c_i \| \beta_i \| rsp_i$;
end
$Sig \leftarrow sig_0 \| sig_1 \| \cdots \| sig_{\delta-1}$;
return $Sig$;
```

4 Security Analysis and Parameter Selection

4.1 Time Complexity of SDP

Security of code-based primitives, considered in this work, relies on the hardness of SDP or its dual version GDP, where the weight ω is around GV-bound for the instance of SDP. However, it is also required to consider the hardness of SDP for other plausible regions of ω, which may open a new avenue for construction of code-based primitives with advanced functionality [15]. In this section, we will discuss the two SDP solvers, we call them SDP Solver-I and SDP Solver-II, to analyze the time complexity of SDP for all possible values of ω. We note that these constructions are rather folklore, although we are not aware of their systematic study, and hence it is presented here[4] .

SDP Solver-I: The SDP Solver-I solves SDP instances $\mathsf{SDP}(H, v, \omega)$ where $(n-k)/2 \leq w \leq (n+k)/2$ in a polynomial time. The SDP Solver-I outputs a vector x such that $Hx^T = v$ and $\mathsf{wt}(x) = \omega$ in a polynomial time for given matrix H, vector v, and positive integer ω. The matrix H can be represented as $H = (H_1 H_2)$ where $H_1 \in \mathbb{F}_2^{(n-k)\times(n-k)}$ and $H_2 \in \mathbb{F}_2^{(n-k)\times k}$. We can assume H_1 is invertible without loss of generality. Otherwise, we can permute the columns of H to make sure that its first $n-k$ columns comprise an invertible matrix.

[4] Just before the publication of this report, we became aware of the work by Debris-Alazard et al. [13], which, in particular, contains an explanation of this issue.

Algorithm 5: SDP Solver-I

```
Input: Matrix $H$, Vector $v$, and Positive integer $w$ such that
       $(n-k)/2 \leq w \leq (n+k)/2$
Output: vector $x$ such that $Hx^T = v$ and $\mathsf{wt}(x) = \omega$
1 repeat
2     $x_2 \leftarrow_{\$} \mathbb{F}_2^k$ such that $\mathsf{wt}(x_2) = \omega - (n-k)/2$;
3     $x_1 \leftarrow H_1^{-1}(v + H_2 x_2^T)$;
4     $x \leftarrow (x_1 \| x_2)$;
5 until $\mathsf{wt}(x) = \omega$;
6 return $x$;
```

Theorem 1. *The SDP Solver-I outputs a vector x such that $Hx = v$ and $\mathsf{wt}(x) = \omega$ in a polynomial time for given matrix H, vector v, and natural number ω.*

Proof. The validity of the codeword can be evaluated as

$$\begin{aligned} Hx^T &= (H_1 H_2)(x_1 \| x_2)^T \\ &= H_1 x_1^T + H_2 x_2^T \\ &= H_1 H_1^{-1}(v + H_2 x_2^T) + H_2 x_2^T \\ &= v + H_2 x_2^T + H_2 x_2^T \\ &= v, \end{aligned}$$

in the third equation, we substitute $x_1^T = H_1^{-1}(v + H_2 x_2^T)$.

The validity of the Hamming weight is shown as

$$\begin{aligned} \mathsf{wt}(x) &= \mathsf{wt}(x_1) + \mathsf{wt}(x_2) \\ &= (n-k)/2 + [w - (n-k)/2] \\ &= \omega. \end{aligned}$$

All operations in the algorithm are completed in a polynomial time. The SDP Solver-I terminates if the Hamming weight of $x_1 = H_1^{-1}(v + H_2 x_2^T)$ is exactly $(n-k)/2$. The probability that it has weight $\frac{n-k}{2}$can be estimated as

$$\begin{aligned} \Pr\left[\mathsf{wt}(x_1) = \frac{n-k}{2}\right] &= \frac{\binom{n-k}{\frac{n-k}{2}}}{2^{n-k}} \\ &= \frac{(n-k)!}{\left[\left(\frac{n-k}{2}\right)!\right]^2 2^{n-k}} \\ &\approx \sqrt{\frac{2}{\pi(n-k)}} \end{aligned}$$

using Stirling's approximation and an assumption that x_1 follows the binomial distribution $B(n-k, 1/2)$. The expectation of the number of iteration is

Table 1. Experimental results of SDP Solver-I

Value	Theoretical	Experimental
$\mathsf{wt}(x_1)$	$(n-k)/2 = 768$	768.0
Variance of $\mathsf{wt}(x_1)$	$\sqrt{n-k}/2 = 19.60$	19.59
Iteration number	$\sqrt{\pi(n-k)/2} = 49.12$	49.51

```
Algorithm 6: SDP Solver-II
   Input: matrix $H$, vector $v$, positive integer $\omega$ such that $0 < \omega < (n-k)/2$
   Output: vector $x$ such that $Hx^T = v$ and $\mathsf{wt}(x) = \omega$
 1 $v' \leftarrow v + H\mathbf{1}$;
 2 $\omega' \leftarrow n - \omega$;
 3 $x' \leftarrow \mathsf{SDP}^{O(H,v',\omega')}$;
 4 return $x' + \mathbf{1}$;
```

$\sqrt{\pi(n-k)/2}$ or $\mathcal{O}\left(\sqrt{n-k}\right)$. Thus, the SDP Solver-I terminates in polynomial time. □

We implement the SDP Solver-I to experimentally confirm our assumption that x_1 follows that the binomial distribution $B(n-k, 1/2)$ and the estimation of the number of iteration in the SDP Solver-I are valid. Table 1 shows that the experimental results agreed well with our theoretical predictions for $n = 3,072$ and $k = n/2 = 1,536$.

SDP Solver-II: The SDP Solver-II reduces an SDP instance $\mathsf{SDP}(H, v, \omega)$ where $(n+k)/2 < \omega < n$ to another SDP instance $\mathsf{SDP}(H, v', \omega')$ where $\omega' = n - \omega$ and $0 < \omega' < (n-k)/2$. The SDP Solver-II has an oracle to another SDP solver for SDP instances $\mathsf{SDP}(H, v', \omega')$ where $0 < \omega' < (n-k)/2$ that outputs a vector x' such that $Hx'^T = v'$ and $\mathsf{wt}(x) = \omega'$. Algorithm 6 describes the detailed algorithm of the SDP Solver-II.

Theorem 2. *The SDP Solver-II outputs a vector x such that $Hx^T = v$ and $\mathsf{wt}(x) = \omega$ in a polynomial time for given a matrix H, a vector v, and a positive integer ω such that $(n+k)/2 < \omega < n$ if an SDP solver for SDP instances SDP(H, v, ω') where $0 < \omega' < (n-k)/2$ exists.*

Proof. The validity of the codeword can be evaluated as

$$\begin{aligned} Hx^T &= H(x' + \mathbf{1})^T \\ &= Hx'^T + H\mathbf{1}^T \\ &= v' + H\mathbf{1}^T \\ &= (v + H\mathbf{1}^T) + H\mathbf{1}^T \\ &= v. \end{aligned}$$

The validity of the Hamming weight is shown as

$$\begin{aligned}\mathsf{wt}(x) &= \mathsf{wt}(x' + \mathbf{1})\\ &= n - \mathsf{wt}(x')\\ &= n - (n - \omega)\\ &= \omega.\end{aligned}$$

$(n + k)/2 < \mathsf{wt}(x) < n$ holds since $0 < \mathsf{wt}(x') < (n - k)/2$. Furthermore, all operations in the algorithm are completed in polynomial time. Thus, the SDP Solver-II terminates in a polynomial time. $\square$

Discussion: The most efficient known algorithm to attack SDP (in the high-noise regime, i.e., when the number of errors is proportional to n) is the Information Set Decoding (ISD) algorithm by Prange [31], and it generalizations and improvements, notably [34,18,24,9,7]. We will use the method of Finiasz and Sendrier [18], since it provides a concrete running time estimates. Next, we compare the method of Finiasz and Sendrier with our observations above. We have plotted the graph by taking $n = 3,072$ and $k = n/2 = 1,536$ in Fig. 1. It explicitly shows the region of ω yielding hard instances. We can observe that hard instances of SDP correspond to a relatively narrow region of "small" values of ω (as well as their "large" counterpart). It is apparent that the method of Finiasz and Sendrier [18] was designed to cover the error weights up to $(n-k)/2$. Hence, it should be used as we described in Solver-II when the error weight ω is moderately high, specifically, beyond $(n - k)/2$ that is 768 in Fig. 1.

4.2 Parameter Selection

The mCFS signature was shown SEUF-CMA secure [26]. Signature schemes, constructed using Fiat-Shamir transformation and its extension to the 5-pass case from zero-knowledge identification schemes, are EUF-CMA secure [3,30]. Moreover, EUF-CMA and SEUF-CMA security include the security against key-recovery attack [22]. Therefore, to choose secure parameters, it is required to measure the hardness of the problem, to which the EUF-CMA or SEUF-CMA security is reduced.

The security of the Stern and Cayrel et al. signatures are reduced to the hardness of SDP and qSDP, respectively. The most efficient known algorithm to attack SDP is the Information Set Decoding (ISD) algorithm by Prange [31] and its extensions, as discussed in the end of Section 4.1. The complexity of q-ary SDP can be evaluated using the method of [27], which is an extension of [18] from binary to an arbitrary finite field.

Security of the Jain et al. signature scheme is reduced to the hardness of xLPN. Since xLPN here used in the "high-noise" regime, the best attack algorithm is still ISD. We thus have used the method of [18] to measure its complexity. Security of mCFS reduced to PGSD problem. Here again, the best attack would use ISD, but now the code parameters correspond to mCFS, and so we have used the modified method of [24] as in [23].

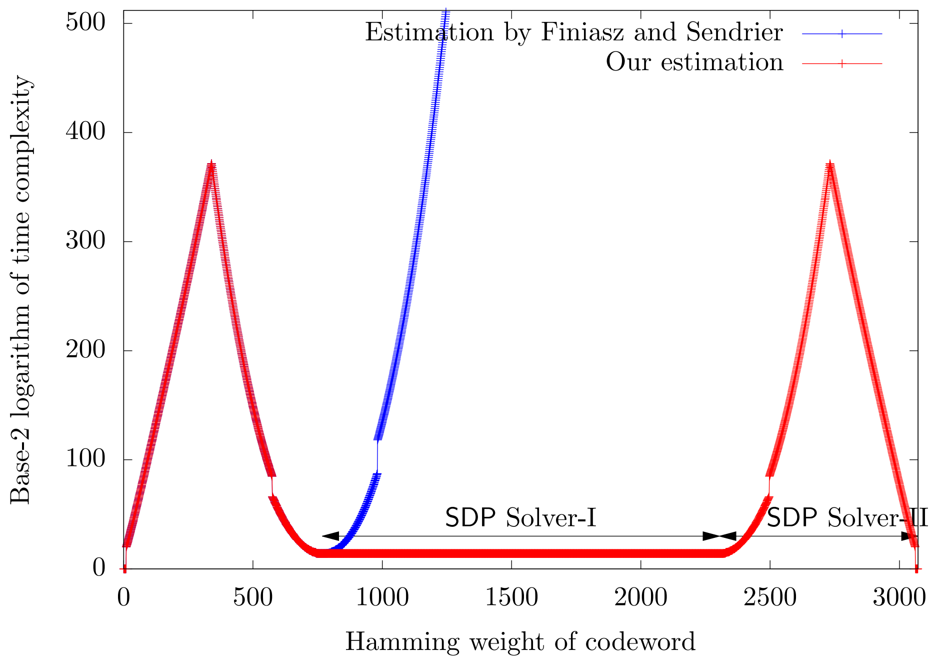

Fig. 1. Time complexity of SDP

mCFS Signature Scheme: The mCFS signature scheme is based on the Niederreiter cryptosystem that uses high-rate Goppa codes. For given integers m and t, the length of the binary (irreducible) Goppa code is taken, for simplicity, to be $n = 2^m$, with dimension $k = n - mt$, and hence they are t-error correcting. The so called complete decoding is used [Algorithm 1, [23]], and the actual weight of the error vector that comes out of the decoder is denoted by ω. For secure parameters, we will consider the estimation by Landais and Sendrier [23]. More specifically, we have used the non-asymptotic version of [24] (see appendix A of [23]). The work-factor formula is as follows [23]:

$$WF_{\mathsf{MMT}}(n,k,\omega) = \min_{p,l} \frac{2l}{\epsilon(p,l)} \left(\frac{1}{2^l} + \frac{1}{L_0^2} \right) \tag{1}$$

where

$$\epsilon(p,l) = \frac{\binom{n-k-l}{\omega-p}}{\left(\min \binom{n}{\omega}, 2^{n-k}\right)} \; and \; L_0 = \begin{pmatrix} \frac{k+l}{2} \\ \frac{p}{4} \end{pmatrix}.$$

The system parameters and data sizes are presented in Table 2[5] as in [23].

[5] Note that the signature size can be reduced by trading it for the verification time (see section 5.5 of [23]). We have calculated the size of secret key by assuming that S is generated using a pseudorandom generator with a 128-bit seed

Table 2. System parameters and data sizes for mCFS signature scheme.

Parameter	80-bit Security	126-bit Security
Independent parameters:		
n	262144	16777216
Derived parameters:		
k	261946	16776976
ω	11	12
Data size:		
sk	6.5 MB	604 MB
pk	5.3 MB	500 MB
Signature	198 bit	288 bit

Stern's Signature Scheme: We select parameters n, k and ω according to the GV bound i.e.,

$$\frac{k}{n} = 1 - H_2\left(\frac{\omega}{n}\right), \tag{2}$$

to maximize the security against attacks using the ISD algorithm as [18]

$$WF_{\mathsf{FS}}(n,k,\omega) = \min_p \frac{2l \min\left(\binom{n}{\omega}, 2^{n-k}\right)}{(1-e^{-1})\binom{n-k-l}{\omega-p}\sqrt{\binom{k+l}{p}}} \tag{3}$$

where

$$l = \log_2\left(2\omega\sqrt{\binom{k}{p}}\right).$$

We select $k = n/2$ and ω around $0.110n$. The number of rounds δ depends on the *soundness error* of the underlying identification scheme. The soundness error of one round [35] is $2/3$. Hence, the number of round δ should satisfy $(2/3)^\delta < 2^{-\lambda}$, , where λ is the security parameter, corresponding to the soundness error (i.e., security failure probability) $2^{-\lambda}$. System parameters and data sizes are presented in Table 3.

Jain et al.'s Signature Scheme: We select parameters n, k and ω according to Eq. (3) and (2). The number of rounds δ depends on the *soundness error* of the underlying identification scheme. Soundness error of the Jain et al. identification scheme [20] is $2/3$. Hence, δ should satisfy $(2/3)^\delta < 2^{-\lambda}$. System parameters and data sizes are presented in Table 4.

Cayrel et al.'s Signature Scheme: We select parameters n, k and ω according to the GV bound i.e.,

$$\frac{k}{n} = 1 - H_q\left(\frac{\omega}{n}\right),$$

to maximize the security against attacks using the ISD algorithm. We select $k = n/2$ and ω is around $0.380n$.

Table 3. System parameters and data sizes for Stern's signature scheme.

Parameter	80-bit Security	128-bit Security
Independent parameters:		
n	620	1,024
δ	137	219
Derived parameters:		
k	310	512
ω	68	112
Data size:		
sk	620 bit	1024 bit
pk	310 bit	512 bit
Signature	93.3 kB	245 kB
Systemf param.	24.0 kB	65.5 kB

Table 4. System parameters and data sizes of Jain et al.'s signature scheme.

Name	80-bit Security	128-bit Security
Independent parameters:		
n	620	1,024
δ	137	219
Derived parameters:		
k	310	512
ω	68	112
Data size:		
sk	930 bit	1536 bit
pk	620 bit	1024 bit
Signature	95.11 kB	263 kB
System param.	24.0 kB	65.5 kB

The relevant formula to evaluate the work factor is as follows [27]:

$$\begin{aligned} &WF_{\mathsf{qISD}}(n,k,\omega,q) \\ &= \min_{l,p_1,p_2} \frac{N_{p,q}(l)}{\sqrt{q-1}} \left(\lambda_q^{-1} \left(\frac{2(q-1)l}{(q\binom{l}{p_2'})(q-1)p_2'} + p_2 \right) \right. \\ &\times \left. \sqrt{\binom{k}{p_1}\binom{l}{p_2}(q-1)^{p-1} + K_q \frac{\binom{k}{p_1}\binom{l}{p_2}(q-1)^{p-1}}{q^l}} \right) \end{aligned}$$

where

$$N_{p,q}(l) = \frac{\min(\binom{n}{\omega}(q-1)^{\omega}, q^{n-k})}{\binom{n-k-l}{\omega-p}\binom{k}{p_1}\binom{i}{p_2}(q-1)^{\omega}},$$

$p = p_1 + p_2$, $p_2' = \lfloor p_2/2 \rfloor$, $\lambda_q = 1 - e^{-1} \approx 0.63$ and $\lfloor p_2/2 \rfloor$ denotes the maximum integer that does not exceed $p_2/2$. The value n should satisfy $WF_{\mathsf{ISD}}(n,k,\omega) > 2^{\lambda}$.

The number of rounds δ depends on the *soundness error* of the underlying identification scheme. The soundness error of the Cayrel et al. identification

Table 5. System parameters and data sizes for Cayrel et al.'s signature scheme over $\mathbb{F}_{256}$.

Parameter	80-bit Security	128-bit Security
Independent parameters:		
n	144	230
δ	80	128
Derived parameters:		
k	72	115
ω	54	87
Data size:		
sk	1.152 bit	1.840 bit
pk	576 bit	920 bit
Signature	89.6 kB	229 kB
System param.	10.4 kB	26.5 kB

scheme [10] is $\frac{q}{2(q-1)}$. Hence, δ should satisfy $(1/2)^\delta < 2^{-\lambda}$. The parameters and data sizes are presented in Table 5.

5 Implementations

We implement Stern's, Jain et al.'s and Cayrel et al.'s signature schemes with 128-bit level of classical security in C language. Eight variables as the elements of $\mathbb{F}_2$ in Stern's and Jain et al.'s signature schemes, and a variable as an element of $\mathbb{F}_{256}$ in Cayrelet al.'s signature scheme are stored to an eight-bit `uint8` variable. Our implementation uses a pre-computation table for multiplication between $\mathbb{F}_{256}$ elements. We used SHA3-256 to implement random oracles used in the signature schemes. For example, a random oracle $h : \{0,1\}^* \to \{0,1\}^{1024}$ is implemented as $v \mapsto$ SHA3-256($\texttt{0x00}\|v$)$\|$SHA3-256($\texttt{0x01}\|v$)$\| \cdots \|$SHA3-256($\texttt{0x03}\|v$). One byte prefix `0x00`, `0x01`, …, `0x03` are auxiliary inputs to achieve independent hash functions. Durstenfeld Shuffle [14] in Algorithm 7 that outputs a random permutation within $\mathcal{O}(n)$ computational complexity, is used in the signature schemes.

The signature size in Cayrel et al.'s signature scheme is smaller and the execution time of the signature generation algorithm is smaller as compared to Stern's signature scheme. Conversely, the signature verification algorithm in Stern signature scheme is faster than that in Cayrel et al. signature scheme since it consists only of hash calculations, permutations, exclusive-or operations, and Hamming weight checks. Table 6 shows the execution time of the signature schemes on a PC with a 3.5 GHz CPU and 16 GB of RAM as well as the size of the secret key, public key, and a signature. The size of the input messages is 32 B.

Discussion: We implemented the Stern's signature scheme, the Jain-Krenn-Pietrzak-Tentes's signature scheme, and the Cayrel-Veron-El Yousfi's signature

Algorithm 7: Durstenfeld Shuffle

```
Input: Natural number $n$
Output: Random permutation $\sigma[i]$ with $n$ elements
for $i \leftarrow 0$ to $n-1$ do
    $\sigma[i] \leftarrow i$
end
for $i \leftarrow 0$ to $n-1$ do
    target $\leftarrow_{\$} \{0, 1, \ldots, n-1\}$;
    tmp $\leftarrow \sigma[\text{target}]$;
    $\sigma[\text{target}] \leftarrow \sigma[n-1-i]$;
    $\sigma[n-1-i] \leftarrow$ tmp;
end
return $\sigma[i]$;
```

Table 6. Summary of execution time and data size of the signature schemes

	Stern	Jain et al.	Cayrel et al.
Keygen	0.0170 ms	0.0201 ms	0.339 ms
Sign	31.5 ms	16.5 ms	24.3 ms
Verify	2.27 ms	135 ms	9.81 ms
sk	1024 bit	1536 bit	1840 bit
pk	512 bit	1024 bit	920 bit
System prams.	65.5 kB	65.5 kB	229 kB
Signature	245 kB	263 kB	229 kB

scheme and compared their performance on a PC. Our implementation shows the Stern's signature scheme to be the most efficient regarding the execution time of key generation and signature verification. The signature generation of the Jain et al.'s scheme is the fastest, but the signature verification of the scheme is slowest.

6 Conclusion

Among the code-based signature schemes Stern [35] features the smallest public key size (512 bits for 128-bit security). We note that we consider the "classical" 128-bit security. It is an open problem to construct a secure code-based signature scheme, especially with a good balance between the key sizes and the signature size. A recent result by Debris-Alazard et al. [13] makes a step in this direction.

Acknowledgments

The authors would like to express sincere appreciation to Prof. Tsuyoshi Takagi for insightful discussions on security evaluation and implementation of code-based signature schemes.

References

1. Alabbadi, M., Wicker, S.B.: Digital signature schemes based on error-correcting codes. In: Proceedings. IEEE International Symposium on Information Theory. pp. 199–199. IEEE (1993)
2. Alaoui, S.M.E.Y., Cayrel, P.L., Bansarkhani, R.E., Hoffmann, G.: Code-Based Identification and Signature Schemes in Software. In: CD-ARES Workshops 2013. pp. 122–136 (2013)
3. Alaoui, S.M.E.Y., Dagdelen, Ö., Véron, P., Galindo, D., Cayrel, P.L.: Extended security arguments for signature schemes. In: International Conference on Cryptology in Africa. pp. 19–34. Springer (2012)
4. Augot, D., Finiasz, M., Sendrier, N.: A Fast Provably Secure Cryptographic Hash Function. Cryptology ePrint Archive, Report 2003/230 (2003), https://eprint.iacr.org/2003/230
5. Barg, S.: Some new np-complete coding problems. Problemy Peredachi Informatsii **30**(3), 23–28 (1994)
6. Barreto, P.S., Cayrel, P.L., Misoczki, R., Niebuhr, R.: Quasi-dyadic cfs signatures. In: International Conference on Information Security and Cryptology. pp. 336–349. Springer (2010)
7. 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: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 520–536. Springer (2012)
8. Berlekamp, E., McEliece, R., van Tilborg, H.: On the Inherent Intractability of Certain Coding Problems. IEEE Transactions on Information Theory **24**(3), 384–386 (1978)
9. Bernstein, D.J., Lange, T., Peters, C.: Smaller decoding exponents: ball-collision decoding. In: Annual Cryptology Conference. pp. 743–760. Springer (2011)
10. Cayrel, P.L., Véron, P., Alaoui, S.M.E.Y.: A zero-knowledge identification scheme based on the q-ary syndrome decoding problem. In: International Workshop on Selected Areas in Cryptography. pp. 171–186. Springer (2010)
11. Courtois, N.T., Finiasz, M., Sendrier, N.: How to Achieve a McEliece-Based Digital Signature Scheme. In: Advances in Cryptology – ASIACRYPT 2001. ASIACRYPT 2001. Lecture Notes in Computer Science. vol. 2248, pp. 157–174 (2001)
12. Dallot, L.: Towards a Concrete Security Proof of Courtois, Finiasz and Sendrier Signature Scheme. In: Research in Cryptology. WEWoRC 2007. Lecture Notes in Computer Science. vol. 4945, pp. 65–77 (2008)
13. Debris-Alazard, T., Sendrier, N., Tillich, J.P.: Wave: A new code-based signature scheme. arXiv preprint arXiv:1810.07554 (2018)
14. Durstenfeld, R.: Algorithm 235: Random permutation. Commun. ACM **7**(7), 420– (Jul 1964). https://doi.org/10.1145/364520.364540, http://doi.acm.org/10.1145/364520.364540
15. Ezerman, M.F., Lee, H.T., Ling, S., Nguyen, K., Wang, H.: A Provably Secure Group Signature Scheme from Code-Based Assumptions. In: Advances in Cryptology – ASIACRYPT 2015. Lecture Notes in Computer Science. vol. 9452, pp. 260–285 (2015)
16. Faugére, J.C., Gauthier-Umana, V., Otmani, A., Perret, L., Tillich, J.P.: A Distinguisher for High-Rate McEliece Cryptosystems. In: Information Theory Workshop (ITW), IEEE. pp. 282–286 (2011)

17. Faugère, J.C., Otmani, A., Perret, L., de Portzamparc, F., Tillich, J.P.: Structural weakness of compact variants of the mceliece cryptosystem. In: 2014 IEEE International Symposium on Information Theory. pp. 1717–1721. IEEE (2014)
18. Finiasz, M., Sendrier, N.: Security Bounds for the Design of Code-Based Cryptosystems. In: Advances in Cryptology – ASIACRYPT 2009. vol. 5912, pp. 88–105 (2009)
19. Harn, L., Wang, D.: Cryptanalysis and modification of digital signature scheme based on error-correcting code. Electronics Letters **28**(2), 157–159 (1992)
20. Jain, A., Krenn, S., Pietrzak, K., Tentes, A.: Commitments and efficient zero-knowledge proofs from learning parity with noise. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 663–680. Springer (2012)
21. Kabatianskii, G., Krouk, E., Smeets, B.: A Digital Signature Scheme Based on Random Error-Correcting Codes. In: Proceedings of the 6th IMA International Conference on Cryptography and Coding (1997)
22. Katz, J.: Digital signatures. Springer Science & Business Media (2010)
23. Landais, G., Sendrier, N.: Implementing CFS. In: International Conference on Cryptology in India. pp. 474–488. Springer (2012)
24. May, A., Meurer, A., Thomae, E.: Decoding random linear codes in $\tilde{O}(2^{0.054n})$. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 107–124. Springer (2011)
25. McEliece, R.J.: A public-key cryptosystem based on algebraic. Coding Thv **4244**, 114–116 (1978)
26. Morozov, K., Roy, P.S., Steinwandt, R., Xu, R.: On the security of the courtois-finiasz-sendrier signature. Open Mathematics **16**(1), 161–167 (2018), https://www.degruyter.com/downloadpdf/j/math.2018.16.issue-1/math-2018-0011/math-2018-0011.pdf
27. Niebuhr, R., Persichetti, E., Cayrel, P.L., Bulygin, S., Buchmann, J.: On lower bounds for information set decoding over &fopf; q and on the effect of partial knowledge. International journal of information and Coding Theory **4**(1), 47–78 (2017)
28. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Prob. Control and Inf. Theory **15**(2), 159–166 (1986)
29. Otmani, A., Tillich, J.P.: An Efficient Attack on All Concrete KKS Proposals. In: Post-Quantum Cryptography. PQCrypto 2011. Lecture Notes in Computer Science. vol. 7071, pp. 98–116 (2011)
30. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology **13**(3), 361–396 (2000)
31. Prange, E.: The use of information sets in decoding cyclic codes. IRE Transactions on Information Theory **8**(5), 5–9 (1962)
32. Roy, P.S., Morozov, K., Fukushima, K., Kiyomoto, S., Takagi, T.: Security analysis and efficient implementation of code-based signature schemes. In: Proceedings of the 5th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP,. pp. 213–220. INSTICC, SciTePress (2019). https://doi.org/10.5220/0007259102130220
33. Shor, P.: Polynomial-time algorithm for prime factorization and discrete logarithms on a quantum computer: proc. In: 35th Annual Symp. on the Foundations of Computer Science. vol. 124 (1994)
34. Stern, J.: A method for finding codewords of small weight. In: International Colloquium on Coding Theory and Applications. pp. 106–113. Springer (1988)

35. Stern, J.: A new identification scheme based on syndrome decoding. In: Advances in Cryptology – CRYPTO 1993. Lecture Notes in Computer Science. vol. 773, pp. 13–21 (1994)
36. Véron, P.: Improved identification schemes based on error-correcting codes. Applicable Algebra in Engineering, Communication and Computing **8**(1), 57–69 (1997)
37. Xinmei, W.: Digital signature scheme based on error-correcting codes. Electronics Letters **26**(13), 898–899 (1990)