*Journal of Discrete Mathematical Sciences & Cryptography*
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 27 (2024), No. 1, pp. 175–187
DOI : 10.47974/JDMSC-1643

# Security issues of CFS-like digital signature algorithms

Giuseppe D'Alconzo *
*Department of Mathematical Sciences*
*Politecnico di Torino*
*Corso Duca degli Abruzzi 24*
*10129 Torino*
*Italy*

Alessio Meneghetti
Paolo Piasenti
*Department of Mathematics*
*Universitá di Trento*
*Via Sommarive 14*
*38123 Povo (Trento)*
*Italy*

**Abstract**

We analyze the security of some variants of the CFS code-based digital signature scheme. We show how the adoption of some code-based hash functions to improve the efficiency of CFS leads to the ability of an attacker to produce a forgery compatible with the rightful user's public key.

***Subject Classification:*** *(2010) 94A62, 94B05.*

***Keywords:*** *Code-based digital signatures, CFS, Post-quantum cryptography, Cryptanalysis.*

## 1. Introduction

With the discovery and the increasingly closer advent of quantum computers, the most widely adopted signature schemes (e.g. DSA [18], ECDSA [15], EdDSA [7], Schnorr [26]) are often considered not secure

* *E-mail:* `giuseppe.dalconzo@polito.it`

anymore because they are well known to be broken by Shor's algorithm [27]. In spite of this, new cryptosystems based on the integer factorisation problem and the discrete logarithm problem are still being developed [1, 20, 22], as well as variants of existing cryptosystems for specific use cases [4]. Nevertheless, several applications need to be resistant even against quantum attacks, and possible countermeasures are obtained by the exploitation of schemes whose security relies on NP-hard problems, or, more generally, on problems whose solutions are thought to be difficult in both the classic and quantum frameworks of computation. Among these alternatives, some of the most prominent are represented by lattice-based cryptography, multivariate polynomial cryptography, hash-based cryptography and interactive identification schemes. These branches of post-quantum cryptography are all present among the finalists of the NIST post-quantum standardization process[i] (the interested reader can see the overview [9]). Notably, code-based digital signature algorithms are not in this list. It is worth mentioning that Classic McEliece [10] is a code-based Key-Encapsulation Mechanism among the finalists of the competition regarding post-quantum key-agreement protocols, and about forty years of cryptanalysis have shown its resiliency and security (Classic McEliece is based on the works of McEliece [21] and Niederreiter [23]). Instead, the initial code-based digital signature schemes did not pass the first round of selection. Although since the earliest and most historical proposal CFS [11], (which will be discussed later on), plenty of ideas and projects have followed, the issue of finding a viable candidate is still an open and tough problem. The key points here are two: of course security, but also efficiency. The main drawback of CFS is the signing time, in fact the message is hashed with a counter until the digest is a decodable.

Further signature schemes have been provided with KKS [16] and its variants [3, 14]. This scheme converts the message into a decodable syndrome, using a different strategy with respect to CFS, but taking into account the attack highlighted in [8], all the variants have to be considered at best one-time signature schemes; additionally, strong caution has to be taken in the choice of parameters, as shown by [24] which broke all the parameters proposed in [3, 16, 17].

On the other hand, most CFS-like schemes persist to be unbroken, despite their slow speed in the signing process due to the attempt-based

[i] NIST Post-Quantum Standardization process webpage: https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization, Accessed: 2021-12-01.

design. Some schemes try to reduce the signing time using the idea behind KKS: instead of searching for a decodable syndrome through the hash of the message, a map that aims to the space of decodable syndromes is used. An example is the $mCFS_c$ signature [25], that hashes the message into a decodable syndrome using a code-based hash function. Unfortunately, in this work we prove that this approach does not work, leaving room to an attacker to forge a valid signature without knowing the secret key.

This work is organized as follows: in the first section, we present the notation and some basic notions from Coding Theory, then we introduce the two signature schemes CFS and mCFS. The second section presents the variant $mCFS_c$ and the concerning code-based hash function. We show an attack on this construction. The third and final section generalizes the strategy adopted in the $mCFS_c$ signature and shows that such approach leads to an attack.

## 1.1 *Notation*

With $\mathbb{F}_2$ we denote the field with 2 elements and with $(\mathbb{F}_2)^n$ the vector space of dimension *n* over $\mathbb{F}_2$. An $[n,k]$ *binary code* $\mathcal{C}$ is a vector subspace of $(\mathbb{F}_2)^n$ of dimension *k*. The elements of $\mathcal{C}$ are called *codewords*. Every $[n,k]$ binary code can be represented as the kernel of a $(n-k)\times n$ matrix *H* called *parity-check matrix*. The syndrome of a vector $v \in (\mathbb{F}_2)^n$ is given by $s = Hv^{\mathrm{T}}$ and the *Hamming weight* of a vector is the number of its non-zero coordinates. With $\|$ we denote the concatenation of strings or vectors.

## 1.2 *Digital signatures and CFS*

A digital signature is a public-key cryptosystem consisting in three algorithms: K Gen is the key generation algorithm that takes in input a security parameter $\lambda$ and outputs the pair of secret and public keys (sk, pk), a signature algorithm Sign that on input sk and a message *m* outputs a signature $\sigma$ of *m*, and a verifier algorithm *Verify* in which, given a public key pk, a message *m* and a signature $\sigma$, it verifies if the signature of the message is valid and is generated by the corresponding secret key sk.

A digital signature algorithm must satisfy some security proprieties: authentication, non repudiation, integrity, non reusability and unforgeability. See [19] for a more detailed study on digital signatures.

The CFS algorithm [11] consists in producing a signature exploiting the Niederreiter public-key cryptosystem [23]. This scenario entails a substantial difference with respect to RSA, for instance: since trapdoor functions allow digital signatures taking advantage of the unique capability of public key owner to invert those functions, it is clear that only the messages whose hashes fall within the ciphertext space can be signed in this way. In our framework, we would like to deal with a linear code for which there exists an efficient decoding algorithm and for which the set of decodable syndromes (namely the ciphertext space) is as big as possible. In formal terms, given a $(n-k)\times n$ parity-check matrix $H$ of such a code, this translates into having a meaningful portion of vectors $s \in \mathbb{F}_2^{n-k}$ for which there exists a corresponding error pattern $e \in \mathbb{F}_2^n$ of Hamming weight less than the correcting capability of the code $t$, such that the syndrome of $e$ is $s$. Since the fact that the union of the spheres centered in codewords and of radius $t$ covers the whole space $\mathbb{F}_2^n$ only happens in the case of perfect codes (which are banally unusable because of the overmuch leak of information they would disclose) the smartest play to make remains to repeatedly hash the message until one obtains a decodable syndrome. Binary Goppa codes [5] represent the best choice as underlying code for both their efficient decoding method (Patterson algorithm) and their steady resistance against all known attacks. This procedure is nothing more than a "hash-and-sign" routine, which inevitably requires several tries. Concretely, given suitable hash function $h$, one produces a sequence $d_0,\dots,d_\iota$ of elements in $\mathbb{F}_2^{n-k}$ such that

$$d_0 = h(m \| 0),\, d_1 = h(m \| 1),\, \dots,\, d_i = h(m \| i),\, \dots\, d_\iota = h(m \| \iota)$$

where $\iota$ is the smallest integer such that $d_\iota$ is a decodable syndrome. The signature is then composed by the corresponding error pattern $e_\iota$ (that only the signer can compute) and by the counter $\iota$. The first straightforward question that arises is about how many attempts are needed in order to obtain a useful syndrome. The answer can be easily found by comparing the total number of syndromes to the number of (efficiently) correctable syndromes:

$$\frac{\#\text{ Decodable Syndromes}}{\#\text{ Total Syndromes}} = \frac{\sum_{i=0}^{t}\binom{n}{i}}{2^{n-k}} \simeq \frac{\binom{n}{t}}{2^{n-k}} \simeq \frac{\frac{n^t}{t!}}{n^t} = \frac{1}{t!}$$

which represent the probability of a syndrome to be decodable (here the relations among the parameters of a generic binary Goppa code have been used, i.e. $k = n - mt$ and $n = 2^m$).

This scheme bases its security on two assumptions: the hardness of both the Syndrome Decoding Problem [6] and the Goppa Code Distinguisher Problem [11].

Now we present the signature scheme of CFS.

- $\mathrm{KGen}_{\mathrm{CFS}}(1^{\lambda})$: select $n, k, t$ according to the security parameter $\lambda$ then pick a random $[n, k]$ binary Goppa code $\mathcal{C}$ with correcting capacity $t$ and parity-check matrix $H$ and let $\mathcal{D}_H$ be an (efficient) syndrome decoding algorithm for $\mathcal{C}$. Pick a random $(n-k)\times(n-k)$ invertible matrix $S$ and a random $n \times n$ permutation matrix $P$ and set $H_{\mathrm{pub}} = SHP$. Chose a hash function $h$. Output $\mathrm{pk} = (h, t, H_{\mathrm{pub}})$ as public key and $\mathrm{sk} = (S, H, P, \mathcal{D}_H)$ as secret key.
- $\mathrm{Sign}_{\mathrm{CFS}}(m, \mathrm{sk})$: given the message $m$, compute $d_i = h(m \| i)$, starting from $i = 0$ and increase it until $d_\imath$ is a decodable syndrome. Set $e = \mathcal{D}_H(S^{-1}d_\imath)$ and output the signature $\sigma = (\imath, eP)$.
- $\mathrm{Verify}_{\mathrm{CFS}}(m, \sigma, \mathrm{pk})$: let $\sigma = (\imath, u)$, verify that $u$ has Hamming weight less or equal than $t$, then compute $a = h(m \| \imath)$ and $b = H_{\mathrm{pub}}u^{\mathrm{T}}$. The signature $\sigma$ is valid if and only if $a = b$.

We can see that the signature is correct, in fact

$$a = h(m \| \imath) = d_\imath = SH \cdot e^{\mathrm{T}} = SHPP^{-1} \cdot e^{\mathrm{T}} = H_{\mathrm{pub}} \cdot u^{\mathrm{T}} = b$$

where $S^{-1}d_\imath = H \cdot e^{\mathrm{T}}$ comes from the fact that $e$ has syndrome $S^{-1}d_\imath$.

In [12] authors propose a modified version of the CFS signature called mCFS. Here the counter $i$ used in Sign is replaced by a random nonce.

## 2. The mCFS$_c$ signature

In this section we describe and analyze the signature in [25], and in Proposition 2 we explicit an attack.

### 2.1 *Code Based Hash Function*

This signature is build on the protocol mCFS [11, 12] using a particular code based hash function. This hash function is based on the work of [2] and it follows the Merkel-Damgard design [13]. Let $r$ be the length of the digest and let $s$ be an integer. The hash function is the iterative application of a *compression function* $f : (\mathbb{F}_2)^s \to (\mathbb{F}_2)^r$, in fact, given a string $m$ proceed as following:

1. consider $m$ padded such that its length is a multiple of $s$ and split $m$ in $|m|/s$ blocks of length $s$;
2. in the first round, combine the first block of $m$ with a fixed initial vector (IV) obtaining the state $L_1$ of length $s$ and compute $f(L_1)$;

3. in the $i$-th round combine $f(L_{i-1})$ with the $i$-th block of $m$ obtaining the $i$-th state $L_i$ and apply $f$ to it;
4. the output of the hash function is given by $f(L_{|m|/s})$.

The hash function used in mCFS$_c$ uses the scheme above and the following compression function $f$. Let $r$ be the length of the digest. Given a $r \times n$ parity-check matrix $H$ of a $[n, n-r]$ binary code $\mathcal{C}$, let $w$ be an integer dividing $n$ and set $l = n/w$ and $s = w\log(l)$. Now we describe the compression function $f : (\mathbb{F}_2)^s \to (\mathbb{F}_2)^r$ based on $H$:

1. let $H_1, \ldots, H_w$ be $r \times w$ matrices such that $H = (H_1, \ldots, H_w)$;
2. given $x \in (\mathbb{F}_2)^s$, split it in $w$ blocks of length $\log(l)$: $x = (x_1, \ldots, x_n)$. We can see each $x_i$ as an integer between 0 and $l-1$;
3. set $f(x)$ as the sum of the $(x_i+1)$-th column of the matrix $H_i$, for $i = 1, \ldots, w$. In formulas, if $(H_i)_j$ is the $j$-th column of $H_i$, we have

$$f(x) = \sum_{i=1}^{w} (H_i)_{x_i+1}.$$

Observe that $f$ strongly depends on the choice of the parity-check matrix $H$.

For the signature mCFS$_c$, in [25], $H$ is chosen as the parity-check matrix of a $[n, n-r]$ binary Goppa code, and the parameter $w$ is less than the correcting capacity $t$ of the code. This yields to the hash function $h_H : \{0,1\}^* \to (\mathbb{F}_2)^r$ based on $H$. Observe that the computation of $h_H$ implies the knowledge of $H$.

**Proposition 1 :** For every state $L_i$ of the hash function $h_H$, $f(L_i)$ is a syndrome of a vector of Hamming weight $w$.

***Proof:*** By construction, the state $x = L_i$ is splitted in $w$ integers $x_1, \ldots, x_w$ between 0 and $l-1$. Let $c_i$ be the vector of length $n$ having support $(x_1+1)+0\cdot l, (x_2+1)+1\cdot l, \ldots, (x_w+1)+(w-1)l$, it has Hamming weight $w$ and $f(L_i)$ is exactly $Hc_i^{\mathrm{T}}$, the syndrome of $c_i$.

We can summarize the compression function as follows: let $n$ and $w$ be positive integers such that $w$ divides $n$ and set $s = w\log(n/w)$. Consider the bijection

$$\begin{aligned} \mathrm{split} : (\mathbb{F}_2)^s &\to ((\mathbb{F}_2)^{\log(n/w)})^w \\ (u_1, \ldots, u_s) &\mapsto (z_1, \ldots, z_w) \end{aligned}$$

that splits a binary vector of length $s$ into $w$ vectors of length $\log(n/w)$. Now we can see every vector in $(\mathbb{F}_2)^{\log(n/w)}$ as an integer between 0 and $n/w-1$. Define

$$\begin{aligned} \delta_t : (\mathbb{F}_2)^s &\to (\mathbb{F}_2)^n \\ (u_1,\ldots,u_s) &\mapsto (v_1,\ldots,v_n) \end{aligned} \tag{1}$$

where $(v_1,\ldots,v_n)$ is the vector of Hamming weight $w$ whose support is given by $(\text{split}(x)_1+1)+0\cdot\frac{n}{w}, (\text{split}(x)_2+1)+1\cdot\frac{n}{w},\ldots,(\text{split}(x)_w+1)+(w-1)\frac{n}{w}$. Then the compression function $f$ can be written as $f(x)=H\cdot\delta_t(x)$.

### 2.2 *The signature scheme*

Since in [25] it is not specified if the hash function is based on the secret matrix $H$ or the public matrix $H_{\text{pub}}$, we first observe that since the hash function is part of the public key, this discloses the secret matrix $H$ and an attack can be performed confronting columns of $H$ and $H_{\text{pub}}$, finding the permutation in quadratic time. Hence, assuming that the hash is based on the public matrix, the signature is given by the following algorithms. Let $\lambda$ be the security parameter.

- $\text{K Gen}_{\text{mCFS}_c}(1^\lambda)$: select $n,k,t$ according to $\lambda$ then pick a random $[n,k]$ binary Goppa code $\mathcal{C}$ with correcting capacity $t$ and parity-check matrix $H$ and let $\mathcal{D}_H$ be an (efficient) syndrome decoding algorithm for $\mathcal{C}$. Pick a random $n\times n$ permutation matrix $P$ and set $H_{\text{pub}}=HP$. Choose an integer $w$ less than $t$ and such that $w$ divides $n$ and construct the hash function $h_{H_{\text{pub}}}:\{0,1\}^*\to(\mathbb{F}_2)^{n-k}$ based on $H_{\text{pub}}$. Output $\text{pk}=(h_{H_{\text{pub}}},t,H_{\text{pub}})$ as public key and $\text{sk}=(H,P,\mathcal{D}_H)$ as secret key.
- $\text{Sign}_{\text{mCFS}_c}(m,\text{sk})$: given the message $m$, pick a random $R$ in $\{1,\ldots,2^{n-k}\}$ and compute $d=h_{H_{\text{pub}}}(h_{H_{\text{pub}}}(m)\,\|\,R)$ and set $e=\mathcal{D}_H(d)$. Output the signature $\sigma=(R,eP)$.
- $\text{Verify}_{\text{mCFS}_c}(m,\sigma,\text{pk})$: let $\sigma=(R,u)$. Verify that $u$ has Hamming weight less or equal than $t$, then compute $a=h_{H_{\text{pub}}}(h_{H_{\text{pub}}}(m)\,\|\,R)$ and $b=H_{\text{pub}}u^{\text{T}}$. The signature $\sigma$ is valid if and only if $a=b$.

The signature scheme is correct using the same argument for CFS.

**Proposition 2 :** Let $(\mathrm{sk}, \mathrm{pk})$ be the output of $\mathrm{K\,Gen}_{\mathrm{mCFS}_c}(1^\lambda)$. An attacker knowing the public key pk can forge a signature compatible to the private key sk for any message $m$.

***Proof:*** Given a public key $\mathrm{pk} = (h_{H_{\mathrm{pub}}}, t, H_{\mathrm{pub}})$ and a message $m$, the attacker picks a random $R$ in $\{1, \ldots, 2^{n-k}\}$ and computes $d = h_{H_{\mathrm{pub}}}(h_{H_{\mathrm{pub}}}(m) \| R)$ but it stops before the last round of the outer hash function $h_{H_{\mathrm{pub}}}$, obtaining the round state $\overline{L} = L_{|m|/s} \in (\mathbb{F}_2)^s$ instead of the digest $d = f(L_{|m|/s})$. He set $u = \delta_t(\overline{L})$ and outputs as a signature for $m$ the tuple $\sigma = (R, u)$. Anyone can verify that this is a valid signature of $m$ compatible with the secret key $\mathrm{sk} = (H, P, \mathcal{D}_H)$, in fact we can compute $a = h_{H_{\mathrm{pub}}}(h_{H_{\mathrm{pub}}}(m) \| R)$ and $b = H_{\mathrm{pub}} u^{\mathrm{T}}$ and observing that $a$ is equal to $b$ since multiplying $u = \delta_t(\overline{L})$ by $H_{\mathrm{pub}}$ is the last step of the hash function $h_{H_{\mathrm{pub}}}$. Therefore $\sigma$ is a valid signature.

## 3. A generalisation of mCFS$_c$

We now slightly generalise mCFS$_c$ by considering a modification of $h_H$, proving that this new entire family of hash functions is vulnerable to the same attack we described for $h_H$ and therefore is not suitable for secure applications.

Let $B_{n,t}$ be the set of vectors in $(\mathbb{F}_2)^n$ of Hamming weight less or equal than $t$. Let $\gamma_t : (\mathbb{F}_2)^s \to (\mathbb{F}_2)^n$ be such that $\mathrm{Im}(\gamma_t) \subseteq B_{n,t}$, i.e. $\gamma_t$ is a function mapping bitstrings of length $s$ into bitstring of length $n$ with a Hamming weight bounded by $t$:

$$\mathrm{w}(\gamma_t(v)) \leq t \quad \forall v \in (\mathbb{F}_2)^s$$

We denote with $\overline{h}_H : \{0,1\}^* \to (\mathbb{F}_2)^{n-k}$ the function mapping messages into syndromes associated to the parity-check matrix $H$ defined by the formula

$$m \mapsto \overline{h}_H(m) = H \cdot \gamma_t(h(m)), \tag{2}$$

where $h(\cdot)$ is any efficient function $\{0,1\}^* \to (\mathbb{F}_2)^s$. For simplicity of notation, we will call $h$ a hash function, even though we do not require here that $h$ satisfy any security property (even though it would be a good practice to choose a cryptographically-secure hash).

With this definition we can consider the following version of CFS, that we call $\widetilde{\mathrm{CFS}}$ :

- $\mathrm{KGen}_{\widetilde{\mathrm{CFS}}}(1^{\lambda})$: randomly choose a code $\mathcal{C}$ to be used in the CFS algorithm (i.e. $\mathcal{C}$ is a code for which there exists an efficient decoder up to $t$ errors and whose randomly picked equivalent codes are indistinguishable from random) with parity-check matrix $H$ and efficient syndrome decoding algorithm $\mathcal{D}_H$. Then randomly choose an invertible $(n-k)\times(n-k)$ matrix $S$ and a permutation $n\times n$ matrix $P$, and define $H_{\mathrm{pub}} = SHP$. Choose an efficient map $\gamma_t$ and a hash $h$. Output $\mathrm{sk} = (S, H, P, \mathcal{D}_H)$ as the secret key and $\mathrm{pk} = (h, \gamma_t, t, H_{\mathrm{pub}})$ as the public key.
- $\mathrm{Sign}_{\widetilde{\mathrm{CFS}}}(m,)$: given the message $m$, compute $d = \overline{h}_{H_{\mathrm{pub}}}(m)$ according to (2). Decode $S^{-1}d$ with the decoder for $H$ and thus obtaining an error vector $e = \mathcal{D}_H(S^{-1}d)$ of Hamming weight at most $t$ and then compute $\overline{e} = eP$. Output the signature $\sigma = \overline{e}$.
- $\mathrm{Verify}_{\widetilde{\mathrm{CFS}}}(m,\sigma,)$: verify that $\sigma = \overline{e}$ has Hamming weight less or equal than $t$, then compute $a = \overline{h}_{H_{\mathrm{pub}}}(m)$ and $\overline{b} = H_{\mathrm{pub}}\overline{e}$. The signature is valid if $a = b$.

The correctness of the above signature scheme is straightforward and follows directly from the correctness of CFS.

We remark that $\mathrm{mCFS}_c$ is (basically) obtained by adopting the algorithm above where:

- $\mathcal{C}$ is a binary irreducible Goppa code;
- $S = I_{n-k}$ is the identity matrix of order $n-k$;
- $\gamma_t$ is the map $\delta_t$ defined in (1);
- $h$ is the code-based hash function $h_{H_{\mathrm{pub}}}$ stopped before the last application of $\delta_t$ and multiplication by $H_{\mathrm{pub}}$, which we denote momentarily $h^{\mathrm{stopped}}_{H_{\mathrm{pub}}}$;

Indeed, with these choices we have $\overline{h}_{H_{\mathrm{pub}}}(m) = H_{\mathrm{pub}} \cdot \delta_t(h^{(\mathrm{stopped})}_{H_{\mathrm{pub}}}(m)) = h_{H_{\mathrm{pub}}}(m)$. We also remark that in $\mathrm{mCFS}_c$ there are other marginal differences with respect to our generalisation, which however do not impact on the main points of the scheme that we sketched above (e.g. computing $h_{H_{\mathrm{pub}}}(h_{H_{\mathrm{pub}}}(m) \,\|\, R)$ instead of $h_{H_{\mathrm{pub}}}(m)$).

**Theorem 3 :** *Let* $(\mathrm{sk}, \mathrm{pk})$ *be the output of* $\mathrm{KGen}_{\widetilde{\mathrm{CFS}}}(1^{\lambda})$. *An attacker knowing the public key* $\mathrm{pk}$ *can forge a signature compatible to the private key* $\mathrm{sk}$ *for any message m.*

***Proof:*** An attacker $\mathcal{A}$ knowing the public parameters $(h, \gamma_t, t, H_{\text{pub}})$ is able to forge any signature. Instead of performing the steps of the signature algorithm, $\mathcal{A}$ performs the following:

1. Given any message $m$, compute $x = \gamma_t(h(m))$;
2. output $x$ as the signature of $m$.

Indeed, $x$ is a valid error vector in $(\mathbb{F}_2)^n$ of Hamming weight at most $t$ (by definition of $\gamma_t$) whose syndrome with respect to the parity-check matrix $H_{\text{pub}}$ is $s = \overline{h}_{H_{\text{pub}}}(m)$. Therefore, any verifier obtains

$$H_{\text{pub}} x^{\text{T}} = H_{\text{pub}} x^{\text{T}} = H_{\text{pub}} \cdot \gamma_t(h(m)) = \overline{h}_{H_{\text{pub}}}(m)$$

and the signature results valid.

We remark how an attacker does not need to know the private key, and the number of operations performed by $\mathcal{A}$ to successfully obtain a forgery are less than the number of operations performed by a honest user to obtain a valid signature. The key-point of the vulnerability of the scheme is that, to obtain a decodable syndrome, we force the application of a function $\gamma_t$ to the output of the hash function before computing the syndrome. Even though this step allows us to obtain a decodable syndrome without having to rely to the (expensive) re-iteration of the signature steps of the original CFS protocol, during the signature algorithm we are forced to explicitly determine a decodable error compatible with the output syndrome.

## 4. Conclusions

One of the practical issues with the CFS signature scheme is the computational effort required to obtain a decodable syndrome from the hash of the message. In [25] the authors attempt to overcome this problem using a Merkel-Damgard-style code-based hash function from the space of binary strings into the set of decodable syndromes, significantly reducing the cost of signing. This approach has proven unsuccessful, since the protocol allows an attacker who does not know the private key to produce a valid signature.

We showed that a generalization of this approach remains insecure: a hash function that sends arbitrarily long binary strings into the set of decodable syndromes can be constructed and yet there exists an attack on this new variation of the CFS signature. Therefore, other solutions should be found, in order to preserve the original security of the scheme but while also reducing the computational effort used in the signing process. The design of a suitable code-based signature should keep in mind both the provable security of CFS-like signatures and the efficiency of the KKS scheme.

## Acknowledgements

The publication was created with the co-financing of the European Union - FSE-REACT-EU, PON Research and Innovation 2014-2020 DM1062 / 2021. The first author acknowledges support from TIM S.p.A. through the PhD scholarship. The first and the second authors are members of the INdAM Research Group GNSAGA. The core of this work is contained in the third author's M.Sc. thesis.

## References

[1] Gessica Alecci, Simone Dutto, and Nadir Murru. Pell hyperbolas in dlp-based cryptosystems. arXiv preprint arXiv:2111.09632, 2021.

[2] Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. In *International Conference on Cryptology in Malaysia*, pages 64–83. Springer, 2005.

[3] Paulo SLM Barreto, Rafael Misoczki, and Marcos A Simplicio Jr. One-time signature scheme from syndrome decoding over generic error-correcting codes. *Journal of Systems and Software*, 84(2):198–204, 2011.

[4] Michele Battagliola, Riccardo Longo, Alessio Meneghetti, and Massimiliano Sala. Threshold ecdsa with an offline recovery party. *Mediterranean Journal of Mathematics*, 19(1):1–29, 2022.

[5] Elwyn Berlekamp. Goppa codes. *IEEE Transactions on Information Theory*, 19(5):590–592, 1973.

[6] Elwyn Berlekamp, Robert McEliece, and Henk Van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[7] Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-speed high-security signatures. *Journal of Cryptographic Engineering*, 2:77–89, 2012.

[8] Pierre-Louis Cayrel, Ayoub Otmani, and Damien Vergnaud. On kabatianskiikrouk-smeets signatures. In *International Workshop on the Arithmetic of Finite Fields*, pages 237–251. Springer, 2007.

[9] Nicola Di Chiano, Riccardo Longo, Alessio Meneghetti, and Giordano Santilli. A survey on NIST PQ signatures. CoRR, abs/2107.11082, 2021.

[10] Tung Chou, Carlos Cid, Simula UiB, Jan Gilcher, Tanja Lange, Varun Maram, Rafael Misoczki, Ruben Niederhagen, Kenneth G Paterson, Edoardo Persichetti, et. al. Classic mceliece: conservative code-based cryptography 10 october 2020. 2020.

[11] Nicolas T Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a mceliece-based digital signature scheme. In International Conference on the Theory and Application of Cryptology and Information Security, pages 157–174. Springer, 2001.

[12] Léonard Dallot. Towards a concrete security proof of courtois, finiasz and sendrier signature scheme. In *Western European Workshop on Research in Cryptology*, pages 65–77. Springer, 2007.

[13] Ivan Bjerre Damgård. A design principle for hash functions. In Conference on the Theory and Application of Cryptology, pages 416–427. Springer, 1989.

[14] Philippe Gaborit and Julien Schrek. Efficient code-based one-time signature from automorphism groups with syndrome compatibility. In 2012 IEEE International Symposium on Information Theory Proceedings, pages 1982–1986. IEEE, 2012.

[15] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ECDSA). *Int. J. Inf. Sec.*, 1:36–63, 08 2001.

[16] Gregory Kabatianskii, Evgenii Krouk, and Ben Smeets. A digital signature scheme based on random error-correcting codes. In *IMA* International Conference on Cryptography and Coding, pages 161–167. Springer, 1997.

[17] Grigorii Kabatiansky, Evgenii Krouk, and Sergei Semenov. Error correcting coding and security for data networks: Analysis of the superchannel concept. John Wiley & Sons, 2005.

[18] Cameron F Kerry and Charles Romine Director. Fips pub 186-4 federal information processing standards publication digital signature standard (dss). 2013.

[19] Cameron F Kerry and Patrick D Gallagher. Digital signature standard (dss). FIPS PUB, pages 186–4, 2013.

[20] DM Kuryazov. Development of electronic digital signature algorithms with compound modules and their cryptanalysis. *Journal of Discrete Mathematical Sciences and Cryptography*, 24(4):1085–1099, 2021.

[21] Robert J McEliece. A public-key cryptosystem based on algebraic. Coding Thv, 4244:114–116, 1978.

[22] Nissa Mehibel and M'hamed Hamadouche. A new enhancement of elliptic curve digital signature algorithm. *Journal of Discrete Mathematical Sciences and Cryptography*, 23(3):743–757, 2020.

[23] Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Prob. Contr. Inform. Theory*, 15(2):157–166, 1986.

[24] Ayoub Otmani and Jean-Pierre Tillich. An efficient attack on all concrete kks proposals. In *International Workshop on Post-Quantum Cryptography*, pages 98–116. Springer, 2011.

[25] Fang Ren, Dong Zheng, WeiJing Wang, et al. An efficient code based digital signature algorithm. *Int. J. Netw. Secur.*, 19(6):1072–1079, 2017.

[26] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Conference on the Theory and Application of Cryptology, pages 239–252. Springer, 1989.

[27] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In Proceedings 35th Annual Symposium on Foundations of Computer Science, pages 124–134. Ieee, 1994.

*Received December, 2021*

*Revised June, 2022*