Quasi-dyadic CFS signatures

Paulo S. L. M. Barreto, Pierre-Louis Cayrel, Rafael Misoczki, Niehbur Robert

To cite this version:

Paulo S. L. M. Barreto, Pierre-Louis Cayrel, Rafael Misoczki, Niehbur Robert. Quasi-dyadic CFS signatures. 6th International Conference Information Security and Cryptology, Oct 2010, Shangai, China. hal-00870939

HAL Id: hal-00870939

https://inria.hal.science/hal-00870939

Submitted on 8 Oct 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Quasi-dyadic CFS signatures

Paulo S. L. M. Barreto[1]*, Pierre-Louis Cayrel[2],
Rafael Misoczki[1], and Robert Niebuhr[3]

[1] Departamento de Engenharia de Computação e Sistemas Digitais (PCS),
Escola Politécnica, Universidade de São Paulo, Brazil.
{pbarreto,rmisoczki}@larc.usp.br
[2] CASED – Center for Advanced Security Research Darmstadt,
Mornewegstrasse, 32
64293 Darmstadt
Germany
pierre-louis.cayrel@cased.de
[3] Technische Universität Darmstadt
Fachbereich Informatik
Kryptographie und Computeralgebra,
Hochschulstraße 10
64289 Darmstadt
Germany
rniebuhr@cdc.informatik.tu-darmstadt.de

Abstract. Courtois-Finiasz-Sendrier (CFS) digital signatures critically depend on the ability to efficiently find a decodable syndrome by random sampling the syndrome space, previously restricting the class of codes upon which they could be instantiated to generic binary Goppa codes. In this paper we show how to construct t-error correcting quasi-dyadic codes where the density of decodable syndromes is high, while also allowing for a reduction by a factor up to t in the key size.

Keywords: post-quantum cryptography, coding-based cryptography, digital signatures, efficient parameters and algorithms.

1 Introduction

Digital signatures are among the most useful and pervasive cryptographic primitives, either *per se* or as part of more elaborate, derived protocols. Yet the overwhelming majority of actually deployed signature schemes seem to rely on the hardness of certain computational problems that are efficiently solvable by quantum computers [19]. Should quantum computers become a technological reality, the task of ensuring that suitable quantum-resistant signatures are available for deployment becomes critical.

The signature algorithm proposed by Courtois, Finiasz and Sendrier, or CFS for short [4], is one of the few and most promising schemes known based on the

* Supported by the Brazilian National Council for Scientific and Technological Development (CNPq) under research productivity grant 303163/2009-7.

difficulty of decoding linear error-correcting codes. However, it has the drawback that public keys tend to be exceedingly large [9], all the more so due to an attack due to Bleichenbacher (unpublished, but described in [9]).

Part of the difficulty resides in obtaining codes with high density of decodable syndromes, since the CFS signing mechanism involves sampling random syndromes until a decodable one is found. Essentially the only family of suitable codes for this purpose is that of binary Goppa codes, for which one can actually correct all t design errors, leading to a signing complexity of $O(t!)$. In comparison, for other classes of codes, no decoding method is known that is capable of efficiently correcting more than about half as many errors; since one has then to design the error correcting capacity twice as high, the CFS signing complexity becomes $O((2t)!) \approx O((t!)^2 \cdot 4^t/\sqrt{t})$, far too much for any secure parameter set.

Quasi-dyadic (QD) codes [14], which constitute a proper subfamily of Goppa codes, have been proposed to address the problem of key reduction in the related McEliece and Niederreiter cryptosystems [13, 15]. However, the original QD construction only yields codes with a fairly low density of decodable syndromes, comparable to generic alternant codes rather than to other Goppa codes.

Our contribution: In this paper we modify the construction algorithm for t-error correcting quasi-dyadic codes [14], where the density of decodable syndromes is high, while also allowing for a reduction by a factor up to t in the key size. This yields dense binary Goppa codes as needed for practical instantiation of CFS signatures.

Recently, in an independent unpublished work Kobara [12] proposed another construction (dubbed flexible quasi-dyadic, or FQD for short) for the same problem, based on selecting distinct linear combinations from the rows of a certain nonsingular matrix, with the associated computational effort of this kind of operation[4]. In contrast, our proposed algorithm is more accurately seen as a natural extension of the original quasi-dyadic construction, whereby a stringent condition on the length of private codes is dropped and replaced by a straightforward consistency validation for the resulting parity-check matrix. It is also computationally simpler, since no linear combinations of rows from the parity-check matrix have to be generated and compared. Besides, contrary to [12] we provide a security assessment of binary QD codes against certain recent structural attacks [7, 20] against this and other families of error-correcting codes. In particular, we argue that, in spite of those attacks being successful against non-binary QD codes (and quasi-cyclic codes as well), binary QD codes remain unscathed and are hence suitable for cryptographic applications.

The remainder of this paper is organized as follows. Section 2 introduces some basic concepts of coding theory. We proceed by describing the CFS signature scheme and its security in Section 3. In Section 4 we review the class of quasi-dyadic codes and propose a modification of the generation algorithm, enlarging

[4] We note *en passant* that, although [12] claims that the FQD construction further reduces key sizes, this does not hold since that method does not produce any code that is not defined by [14, Theorem 2].

that class with codes where the density of decodable syndromes increases by an exponential factor in the number of errors. We discuss security issues of the resulting quasi-dyadic CFS scheme in Section 5. We conclude in Section 6.

2 Preliminaries

In what follows all vector and matrix indices are numbered from zero onwards.

Definition 1. *Given a ring $\mathcal{R}$ and a vector $h = (h_0, \ldots, h_{n-1}) \in \mathcal{R}^n$, the* dyadic matrix *$\Delta(h) \in \mathcal{R}^{n \times n}$ is the symmetric matrix with components $\Delta_{ij} = h_{i \oplus j}$, where $\oplus$ stands for bitwise exclusive-or on the binary representations of the indices. The sequence h is called its* signature. *The set of dyadic $n \times n$ matrices over $\mathcal{R}$ is denoted $\Delta(\mathcal{R}^n)$. Given $t > 0$, $\Delta(t, h)$ denotes $\Delta(h)$ truncated to its first t rows.*

One can recursively characterize a dyadic matrix when n is a power of 2: any 1×1 matrix is dyadic, and for $k > 0$ any $2^k \times 2^k$ dyadic matrix M has the form

$$M = \begin{bmatrix} A & B \\ B & A \end{bmatrix},$$

where A and B are $2^{k-1} \times 2^{k-1}$ dyadic matrices. It is not hard to see that the signature of a dyadic matrix coincides with its first row. Dyadic matrices form a commutative subring of $\mathcal{R}^{n \times n}$ as long as $\mathcal{R}$ is commutative [11]. We will consider here only the case where $\mathcal{R} = \mathbb{F}_q$, the finite field with q (a power of 2) elements.

Definition 2. *A* dyadic permutation *is a dyadic matrix $\Pi^i \in \Delta(\{0,1\}^n)$ whose signature is the i-th row of the identity matrix.*

Definition 3. *A* quasi-dyadic *matrix is a (possibly non-dyadic) block matrix whose component blocks are dyadic submatrices. A* quasi-dyadic (QD) code *is a linear error-correcting code that admits a quasi-dyadic parity-check matrix.*

Definition 4. *Given two disjoint sequences $z = (z_0, \ldots, z_{t-1}) \in \mathbb{F}_q^t$ and $L = (L_0, \ldots, L_{n-1}) \in \mathbb{F}_q^n$ of distinct elements, the* Cauchy matrix *$C(z, L)$ is the $t \times n$ matrix with elements $C_{ij} = 1/(z_i - L_j)$, i.e.*

$$C(z, L) = \begin{bmatrix} \dfrac{1}{z_0 - L_0} & \cdots & \dfrac{1}{z_0 - L_{n-1}} \\ \vdots & \ddots & \vdots \\ \dfrac{1}{z_{t-1} - L_0} & \cdots & \dfrac{1}{z_{t-1} - L_{n-1}} \end{bmatrix}.$$

Cauchy matrices have the property that all of their submatrices are nonsingular [18]. Notice that, in general, Cauchy matrices are not dyadic and vice-versa, although the intersection of these two classes is non-empty in characteristic 2.

Definition 5. *Given $t > 0$ and a sequence $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$, the* Vandermonde matrix $\mathrm{vdm}(t, L)$ *is the $t \times n$ matrix with elements $V_{ij} = L_j^i$.*

Definition 6. *Given a sequence $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$ of distinct elements and a sequence $D = (D_0, \dots, D_{n-1}) \in \mathbb{F}_q^n$ of nonzero elements, the* Generalized Reed-Solomon code $GRS_t(L, D)$ *is the $[n, k, t]$ linear error-correcting code defined by the parity-check matrix*

$$H = \mathrm{vdm}(t-1, L) \cdot \mathrm{diag}(D).$$

An alternant code *is a subfield subcode of a Generalized Reed-Solomon code.*

Let p be a prime power, let $q = p^d$ for some d, and let $\mathbb{F}_q = \mathbb{F}_p[x]/b(x)$ for some irreducible polynomial $b(x) \in \mathbb{F}_p[x]$ of degree d. Given a code specified by a parity-check matrix $H \in \mathbb{F}_q^{t \times n}$, the *trace construction* derives from it an $\mathbb{F}_p$-subfield subcode by fixing a basis of $\mathbb{F}_q$ over $\mathbb{F}_p$, writing the $\mathbb{F}_p$-coefficients of each $\mathbb{F}_q$-component of H onto d successive rows of a parity-check matrix $T_d(H) \in \mathbb{F}_p^{dt \times n}$ for the subcode. The related *co-trace* parity-check matrix $T'_d(H) \in \mathbb{F}_p^{dt \times n}$, equivalent to $T_d(H)$ by a left permutation, is obtained from H by writing the $\mathbb{F}_p$-coefficients of terms of equal degree from all components from a column of H onto successive rows of $T'_d(H)$.

Thus, given $\mathbb{F}_q$ elements $u_i(x) = u_{i,0} + \cdots + u_{i,d-1}x^{d-1}$, the (co-)trace construction maps a column $(u_0, \dots, u_{t-1})^\mathsf{T}$ from H to the column $(u_{0,0}, \dots, u_{0,d-1}; \dots; u_{t-1,0}, \dots, u_{t-1,d-1})^\mathsf{T}$ on the trace matrix $T_d(H)$, and to the column $(u_{0,0}, \dots, u_{t-1,0}; \dots; u_{0,d-1}, \dots, u_{t-1,d-1})^\mathsf{T}$ on the co-trace matrix $T'_d(H)$.

Definition 7. *Given a prime power p, $q = p^d$ for some d, a sequence $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$ of distinct elements, and a polynomial $g(x) \in \mathbb{F}_q[x]$ of degree t such that $g(L_i) \neq 0$ for $0 \leqslant i < n$, the* Goppa code $\mathit{\Gamma}(L, g)$ *over $\mathbb{F}_p$ is the alternant code over $\mathbb{F}_p$ corresponding to $GRS_t(L, D)$, where $D = (g(L_0)^{-1}, \dots, g(L_{n-1})^{-1})$.*

A binary Goppa code can correct up to t errors, sometimes slightly more [17, 2], regardless of whether the generator $g(x)$ is irreducible or not. For all other cases, no method is generally known to correct more than about $t/2$ errors.

Consider a t-error correcting $\mathbb{F}_p$-alternant code of length n derived from a code over $\mathbb{F}_{p^m}$. The syndrome space has size p^{mt}. However, the decodable syndromes are only those that correspond to error vectors of weight not exceeding t. In other words, only $\sum_{w=1}^{t} \binom{n}{w}(p-1)^w$ nonzero syndromes are decodable, and hence their density is

$$\delta = \frac{1}{p^{mt}} \sum_{w=1}^{t} \binom{n}{w} (p-1)^w.$$

If the code length is a fraction $1/p^c$ for some $c \geqslant 0$ of the full length, i.e. $n = p^{m-c}$, the density can be approximated as

$$\delta \approx (n^t/t!)(p-1)^t/p^{mt} = (p^{m-c})^t(p-1)^t/(p^{mt}t!) = (p-1)^t/(p^{ct}t!).$$

A particularly good case is therefore $\delta \leqslant 1/t!$, which occurs when $(p^c/(p-1))^t \leqslant 1$, i.e. $c \leqslant \log_p(p-1)$, or $n \geqslant p^m/(p-1)$. Unfortunately this also means that for binary codes the highest densities are attained only by full or nearly full length codes, otherwise the density is reduced by a factor 2^{ct}. For full length binary codes ($p = 2$, $n = 2^m$) the density simplifies to

$$\delta \approx \frac{1}{2^{mt}} \frac{n^t}{t!} = \frac{1}{t!}.$$

3 CFS signature scheme

Courtois, Finiasz and Sendrier proposed in [4] the first practical signature scheme based on coding theory. The Full Domain Hash (FDH) approach assumes that all the hash values can be inverted by decryption.

3.1 Description

The CFS signature scheme is based on the Niederreiter cryptosystem: signing a document requires hashing it to a syndrome and then decoding it to an error vector of a certain weight t. Since not all syndromes are decodable, a counter is hashed with the message, and the signer tries successive counter values until a decodable syndrome is found. The signature consists of both the error pattern of weight t corresponding to the syndrome, and the counter value yielding this syndrome.

Let $\mathcal{H} : \{0,1\}^* \times \mathbb{N} \to \mathbb{F}_q^k$ be a random oracle for a given vector space $\mathbb{F}_q^k$ over a finite field $\mathbb{F}_q$. Formally, the CFS signature scheme consists of the following algorithms:

- Keygen: For the desired security level expressed by suitable integers q, n, k, t, choose a linear t-error correcting $[n,k]$-code over $\mathbb{F}_q$ defined by a public parity-check matrix H with a private decoding trapdoor $\mathcal{T}$. The private-public key pair is $(\mathcal{T}, H)$.
- Sign: Let $m \in \{0,1\}^*$ be the message to sign. Find $c \in \mathbb{N}$ (either sequentially or by random sampling) such that $s \leftarrow \mathcal{H}(m,c)$ is a decodable syndrome. Using the decoding trapdoor $\mathcal{T}$, find $e \in \mathbb{F}_q^n$ of weight $\mathsf{wt}(e) \leqslant t$ such that $He^{\mathrm{T}} = s^{\mathrm{T}}$. The signature is the pair (e,c).
- Verify: Let (e,c) be a purported signature for message m. Compute $s \leftarrow \mathcal{H}(m,c)$, and accept iff $\mathsf{wt}(e) \leqslant t$ and $He^{\mathrm{T}} = s^{\mathrm{T}}$.

The original description of the CFS scheme [4] suggests using a binary Goppa code and scanning over the c values sequentially. Random counter sampling (limited to r bits, i.e. from the set $\{0 \dots 2^r - 1\}$) was proposed in [5] to obtain a security proof in the random oracle model, assuming the intractability of the following problems:

Definition 8 (Goppa Parametrized Bounded Decoding (GPBD)). *Given a matrix $H \in \mathbb{F}_2^{r \times n}$ and a syndrome $s \in \mathbb{F}_2^r$, is there a word $e \in \mathbb{F}_2^n$ of weight $\mathsf{wt}(e) \leqslant r/\lg n$ such that $He^T = s^T$?*

Definition 9 (Goppa Code Distinguishing (GD)). *Given $m, t, n \in \mathbb{N}$ and a matrix $H \in \mathbb{F}_2^{mt \times n}$, is H the parity-check matrix of a binary t-error correcting $[n, n - mt]$ Goppa code?*

The main drawback of the CFS scheme is the key size. For the 80-bit security level, the authors of [4] suggest taking $m = 16$ and $t = 9$, leading to 1152 KiB keys. In the next section, we propose a construction that allows for smaller keys (and faster arithmetic), by using quasi-dyadic Goppa codes.

4 Quasi-dyadic codes

We recap the original construction of binary QD Goppa codes [14]. These are characterized by Theorem 1, which in turn suggests Algorithm 1, taken from the same reference.

Theorem 1 ([14]). *Let $H \in \mathbb{F}_q^{n \times n}$ with $n > 1$ be simultaneously a dyadic matrix $H = \Delta(h)$ for some $h \in \mathbb{F}_q^n$ and a Cauchy matrix $H = C(z, L)$ for two disjoint sequences $z \in \mathbb{F}_q^n$ and $L \in \mathbb{F}_q^n$ of distinct elements. Then $\mathbb{F}_q$ is a binary field, h satisfies*

$$\frac{1}{h_{i \oplus j}} = \frac{1}{h_i} + \frac{1}{h_j} + \frac{1}{h_0}, \tag{1}$$

and $z_i = 1/h_i + \omega$, $L_j = 1/h_j + 1/h_0 + \omega$ for some $\omega \in \mathbb{F}_q$.

4.1 Quasi-dyadic codes for CFS signatures

Because the sequences z and L must be disjoint and consist of distinct elements, the length of the codes Algorithm 1 produces are upper bounded by $n \leqslant 2^{m-1}$, and hence the syndrome density is bound by $1/(2^t t!)$. Clearly, if z and L were not disjoint at least one element $H_{ij} = 1/(z_i - L_j)$ of matrix H would be undefined due to division by zero.

However, the CFS signature scheme only needs a very small t (say, $t \lesssim m$), meaning that most elements of the sequence z, and hence the corresponding rows of the largest possible matrix $\Delta(h)$, are left unused anyway when defining the actual code. It is therefore possible to allow matrix $\Delta(h)$ to contain undefined entries, as long as the rows and columns containing those entries are removed afterwards, and that $\Delta(t, h)$ itself contains only well-defined entries. This means the code length can be naturally extended all the way up to $2^m - t$, corresponding to an exact partition of the field elements from $\mathbb{F}_{2^m}$ into two disjoint sequences z and L.

In principle, this strategy can fail, i.e. the first t rows could contain an undefined element. This can be handled by either choosing a different code, or else by carefully rearranging the dyadic signature h into some h' in order to permute the rows of $\Delta(h)$ and eliminate undefined elements from $\Delta(t, h')$. As it turns out, the probability that an improper element will appear on the first t rows of $\Delta(h)$

Algorithm 1 Constructing a purely dyadic binary Goppa code

```
INPUT: $q$ (a power of 2), $n \leqslant q/2$, $t$.
OUTPUT: Support $L$, generator polynomial $g$, dyadic parity-check matrix $H$ for a
    Goppa code $\Gamma(L, g)$ of length $n$ and design distance $2t+1$ over $\mathbb{F}_q$.
$U \leftarrow \mathbb{F}_q \setminus \{0\}$
▷ Choose the dyadic signature $(h_0, \ldots, h_{n-1})$. N.B. Whenever $h_j$ with $j > 0$ is taken
  from $U$, so is $1/(1/h_j + 1/h_0)$ to prevent a potential spurious intersection between
  $z$ and $L$.
$h_0 \xleftarrow{\$} U$, $U \leftarrow U \setminus \{h_0\}$
for $s \leftarrow 0$ to $\lceil \lg n \rceil - 1$ do
    $i \leftarrow 2^s$
    $h_i \xleftarrow{\$} U$, $U \leftarrow U \setminus \{h_i,\ 1/(1/h_i + 1/h_0)\}$
    for $j \leftarrow 1$ to $i - 1$ do
        $h_{i+j} \leftarrow 1/(1/h_i + 1/h_j + 1/h_0)$
        $U \leftarrow U \setminus \{h_{i+j},\ 1/(1/h_{i+j} + 1/h_0)\}$
    end for
end for
$\omega \xleftarrow{\$} \mathbb{F}_q$
▷ Assemble the Goppa generator polynomial:
for $i \leftarrow 0$ to $t - 1$ do
    $z_i \leftarrow 1/h_i + \omega$
end for
$g(x) \leftarrow \prod_{i=0}^{t-1} (x - z_i)$
▷ Compute the support:
for $j \leftarrow 0$ to $n - 1$ do
    $L_j \leftarrow 1/h_j + 1/h_0 + \omega$
end for
$h \leftarrow (h_0, \ldots, h_{n-1})$
$H \leftarrow \Delta(t, h)$
return $L$, $g$, $H$
```

is extremely low. As a consequence, the simpler strategy of just trying another code, if this is ever necessary in practice, is much simpler to implement without any measurable impact on either security or efficiency.

This idea is captured in Algorithm 2, which in practice is as simple to implement and as efficient as Algorithm 1. In a sense it is actually somewhat simpler, since less field elements have to be computed and discarded from the remaining allowed set U. Notice that improper array elements, whose evaluation would cause division by zero, are represented by a zero value, since this cannot ever occur on a proper array entry.

Algorithm 2 Constructing a purely dyadic, CFS-friendly code

```
INPUT: $m$, $n$, $t$.
OUTPUT: A dyadic signature $h$ from which a CFS-friendly $t$-error correcting binary
    Goppa code of length $n$ can be constructed from a code over $\mathbb{F}_{2^m}$, and the sequence
    $b$ of all consistent blocks of columns (i.e. those that can be used to define the code
    support).
$q \leftarrow 2^m$
repeat
    $U \leftarrow \mathbb{F}_q \setminus \{0\}$
    $h_0 \xleftarrow{\$} U,\ U \leftarrow U \setminus \{h_0\}$
    for $s \leftarrow 0$ to $m - 1$ do
        $i \leftarrow 2^s$
        $h_i \xleftarrow{\$} U,\ U \leftarrow U \setminus \{h_i\}$
        for $j \leftarrow 1$ to $i - 1$ do
            if $h_i \neq 0$ and $h_j \neq 0$ and $1/h_i + 1/h_j + 1/h_0 \neq 0$ then
                $h_{i+j} \leftarrow 1/(1/h_i + 1/h_j + 1/h_0)$
            else
                $h_{i+j} \leftarrow 0$ ▷ undefined entry
            end if
            $U \leftarrow U \setminus \{h_{i+j}\}$
        end for
    end for
    $c \leftarrow 0$ ▷ also: $U \leftarrow \mathbb{F}_q$
    if $0 \notin \{h_0, \ldots, h_{t-1}\}$ then ▷ consistent root set
        $b_0 \leftarrow 0,\ c \leftarrow 1$ ▷ also: $U \leftarrow U \setminus \{1/h_i, 1/h_i + 1/h_0 \mid i = 0, \ldots, t-1\}$
        for $j \leftarrow 1$ to $\lfloor q/t \rfloor - 1$ do
            if $0 \notin \{h_{jt}, \ldots, h_{(j+1)t-1}\}$ then ▷ consistent support block
                $b_c \leftarrow j,\ c \leftarrow c+1$ ▷ also: $U \leftarrow U \setminus \{1/h_i + 1/h_0 \mid i = jt, \ldots, (j+1)t-1\}$
            end if
        end for
    end if
until $ct \geqslant n$ ▷ consistent roots and support
$h \leftarrow (h_0, \ldots, h_{q-1}),\ b \leftarrow (b_0, \ldots, h_{c-1})$ ▷ also: $\omega \xleftarrow{\$} U$
return $h$, $b$ ▷ also: $\omega$
```

Algorithm 2 produces a code that is amenable to the same treatment as a generic Goppa code when instantiating the CFS signature scheme, namely, apply the trace construction of a binary alternant code from the code over $\mathbb{F}_{2^m}$, permute the columns of the corresponding parity-check matrix, and put the result in systematic form to get a CFS public key. However, this simple technique does not benefit from a possible reduction in key size since it destroys the quasi-dyadic structure. Algorithm 2 is designed to preserve that structure by removing the entire $t \times t$ block where one (or more) improper column lies.

The strategy to get shorter keys is then to permute the blocks (or a large subset thereof) among themselves, dyadic-permute each block individually, and apply the co-trace construction to get a binary quasi-dyadic alternant code. This has to be done carefully so as to fully hide the code structure. The obvious approach is to delete more blocks and/or to replace them (and also the blocks that contain improper columns) by random dyadic blocks (the latter case corresponds to Wieschebrink's technique). One has to be careful here as well, since if only a fraction $1/2^c$ of the columns remain, the syndrome density effectively decreases by a factor 2^{ct} as seen above. A sensible choice, which we will usually adopt, is to take a fraction $2^{-1/t}$ of full code length (i.e. $c = 1/t$), since this only increases the average signing time by a factor of 2.

Typical parameter combinations are put forward on Table 1. We will later examine some possible parameter choices in the context of, and as a result of, the security discussion in Section 5.

Table 1. Suggested parameters for practical security levels.

level	m	t	$n = \lfloor 2^{m-1/t} \rceil$	$k = n - mt$	key size (KiB)
80	15	12	30924	30744	169
100	20	12	989724	989484	7248
120	25	12	31671168	31670868	289956

5 Security

Most of the time, the most threatening attacks are based on decoding algorithms for generic linear codes. There are two main families of generic algorithms, (Generalized) Birthday Algorithm (GBA) and Information Set Decoding (ISD). However, due to the peculiar nature of QD codes one has to take care of structural attacks as well. We provide an overview of these attacks and their impact on the choice of parameters for a quasi-dyadic CFS instantiation.

5.1 (Generalized) birthday attacks

An attack due to Daniel Bleichenbacher against the CFS scheme is described in [9]. We can shortly describe this attack as follows:

- build 3 lists L_0, L_1, and L_2 of XORs of respectively t_0, t_1 and t_2 columns of H (with $t = t_0 + t_1 + t_2$).
- merge the two lists L_0 and L_1 into a list L'_0 of XORs of $t_0 + t_1$ columns of H, keeping only those starting with λ zeros.
- repeat the following steps:
  - choose a counter and compute the corresponding document hash,
  - XOR this hash with all elements of L_2 matching on the first λ bits,
  - look up each of these XORs in L'_0: any complete match gives a valid signature.

Due to this attack, the values of m and t proposed in the original CFS scheme are not enough to ensure a proper security level. Therefore, instead of $m = 16$ and $t = 9$, the authors of [9] propose $m = 21$ and $t = 10$, or $m = 19$ and $t = 11$, or $m = 15$ and $t = 12$, as new parameters for a security of more than 2^{80} binary operations.

5.2 Decoding attacks

The authors of [9] derive lower bounds on the work factor of idealized versions of the ISD and of the GBA. Table 2 shows the cost of these two attacks against various parameter sets, calculated according to [9]. Table 3 lists for each t the minimum m such that the security level is about 2^{80} or larger, taking both ISD and GBA into account, and the resulting key sizes.

For simplicity, on Table 2 we assume full-length codes with $n = 2^m$. In practice we would adopt slightly shorter punctured codes, taking e.g. $n = 2^{m-1/t}$ since this keeps the signing time within a factor of 2 from the corresponding time for full-length codes; this choice is adopted in Table 3. While the key size may be too large for smaller t, and conversely the signing complexity may be too large for larger t, intermediate combinations like $m = 15$, $t = 12$ may be just right in practice for this security level.

Table 2. Time complexity (given as lg) for the ISD / GBA attack against the CFS scheme using binary codes with various parameter sets.

n t	2^{15}	2^{16}	2^{17}	2^{18}	2^{19}	2^{20}	2^{21}
9	66.4/60.3	72.2/63.3	78.1/66.4	83.9/69.5	89.8/72.5	95.6/75.6	101.5/78.7
10	72.8/63.1	79.5/66.2	86.2/69.3	93.0/72.4	99.8/75.4	106.5/78.5	113.3/81.5
11	79.0/67.2	86.6/71.3	94.3/75.4	102.0/79.5	109.6/83.6	117.4/87.6	125.1/91.7
12	85.2/81.5	93.7/85.6	102.2/89.7	110.8/93.7	119.4/97.8	128.1/101.9	136.7/105.9

5.3 Structural attacks

Structural attacks attempt to benefit from the symmetries existent in the public and private information. As an example of the potential of such attacks, the

Table 3. Minimum m to yield time complexity of at least 2^{80}, expected number of signing attempts, and key sizes.

(t, m)	$(8, 25)$	$(9, 22)$	$(10, 21)$	$(11, 19)$	$(12, 15)$	$(13, 14)$	$(14, 14)$	$(15, 13)$	$(16, 13)$
sec level	$2^{81.7}$	$2^{81.7}$	$2^{81.5}$	$2^{83.6}$	$2^{81.5}$	$2^{80.7}$	$2^{84.1}$	$2^{80.7}$	$2^{84.6}$
avg sign atts	$2^{16.3}$	$2^{19.5}$	$2^{22.8}$	$2^{26.3}$	$2^{29.8}$	$2^{33.5}$	$2^{37.3}$	$2^{41.3}$	$2^{45.3}$
key size (KiB)	93902	93862	25080	12560	169	346	187	187	13

technique described in [16] successfully extracts the private key from the quasi-cyclic codes proposed in [10]. That scheme takes a binary quasi-cyclic subcode of a BCH code of length n as the secret code. The structure is hidden by a heavily constrained permutation in order to produce a quasi-cyclic public code. This implies that the permutation transformation is completely described with n_0^2 binary entries where $n_0 \ll n$ is the quasi-cyclic index. The attack takes advantage of the fact that the secret is a subcode of completely known BCH code. The idea is to construct a system of linear equations by exploiting the public generator matrix and a known parity-check matrix of the BCH code, so as to get an overdefined (and easily solvable) system satisfied by the unknown permutation matrix.

We show how to adapt this attack to our variant. Let H_0 be a private parity-check matrix of the underlying $[n, k, 2t + 1]$ Goppa code, for which a decoding trapdoor is known to exist (or at least revealing that trapdoor, as is the case for the purely dyadic parity-check matrix $\varDelta(t, h)$ constructed in Section 4.1). Consider matrix $G = [G_P \mid O]$ where G_P is a generator matrix of the code defined by the public parity-check matrix H, and O is the zero matrix with $N - n$ columns. Clearly, there exists an $N \times N$ matrix X such that:

$$H_0 X G^{\mathrm{T}} = O. \qquad (2)$$

Writing $N = N_0 t$, X is an $N_0 \times N_0$ block matrix whose blocks are either the $t \times t$ zero matrix or a $t \times t$ dyadic permutation (the actual permutation varying from block to block). Let $n_0 < N_0$ be the number of nonzero blocks in X (all of them on the n_0 leftmost columns of X, , without loss of generality because of the structure of G). There are therefore $\binom{N_0}{n_0} n_0! t^{n_0}$ possibilities for X. The situation is almost the same as for quasi-cyclic alternant codes [1]. The main difference is that, rather than having small powers of a fixed value whose successive powers are on the diagonal, here we have one single element whose position assumes one out of t possibilities (and this is not fixed). Therefore solving the system given by Equation (2) reveals all the private information.

The first obstacle, however, is obtaining H_0. The attack against quasi-cyclic codes simply guesses the private parity-check matrix since there are only $O(2^m)$ possibilities. In the QD case, on the other hand, guessing H_0 would already incur a superpolynomial cost $O(2^{m^2})$. To makes things worse, for each guess of H_0, the attacker would have to mount and solve a linear system over the ring of dyadic $t \times t$ matrices, containing $n_0 \times N_0$ unknowns, or alternatively a system over $\mathbb{F}_2$ directly, increasing the number of unknowns by a factor t^2. In either case the total amount of work is prohibitively high. The attacker might try to guess X instead, or at least the positions of its nonzero dyadic blocks, but this incurs an extra cost factor $\binom{N_0}{n_0}n_0!$, which is too high for practical parameters. A further difficulty is that the systems are highly underdefined (typically containing hundreds of thousands of equations in tens of millions of unknowns).

None of the above ideas seems to lead to any promising strategy for a structural attack based on systems of linear equations. We next examine the possibility of using systems of quadratic equations to reduce the overall attack complexity.

5.4 Attacks based on multivariate quadratic equations

The structural attacks outlined above are based on solving certain systems of linear equations after guessing part of the unknown information, a task that, the attacker hopes, is made easier by the structure of the underlying codes, but as we saw the chances of these ideas ever succeeding are meagre at best. Recently, Faugère *et al.* [7] proposed to reduce the decoding problem for quasi-dyadic codes (and others) to the problem of solving systems of multivariate quadratic equations (MQE) instead. The overall idea is to find an alternant decoder for the public code directly, i.e. to write the public parity-check matrix as $H = VD$ for an unknown Vandermonde matrix V and an unknown diagonal matrix D defined over the public field $\mathbb{F}_{2^d}$, where $d \mid m$. The unknown components of V and D in the defining equation $H = VD$ give rise to an instance of the MQE problem. By making careful use of the structure of H, the authors of [7] are able to reduce the complexity of such instances, since many component equations become linear, and the truly quadratic part involves a reduced number of variables. This way they are able to break all parameters proposed in [1] and [14] over extension fields.

Apart from the fact that the attack complexity increases steeply as the codes are defined over ever smaller extension fields, to the effect that no actual attack was described against any of the published *binary* parameters, we argue that this strategy, at least as it is presented, cannot yield an attack against binary QD codes, even if it succeeds (at an impractically high cost but still faster than other methods) against e.g. quasi-cyclic codes. The reason is that the attack principle is to construct an *alternant* trapdoor directly from the public code defined by H, which is *not* a Goppa code except with overwhelmingly low probability. This trapdoor can be used to correct about $t/2$ errors at most, where t is the design number of errors. For all alternant codes except binary Goppa codes this is exactly the same as the number of errors that can be introduced and then successfully corrected using the *private* trapdoor, which explains why the attack is successful as long as the associated MQE instance can be solved in practice.

This is the case for codes over extension fields, as demonstrated in [7] (see also [20]). Whether or not this is also the case for non-Goppa binary codes is at best unclear for the time being as we pointed out. However, for the specific case of binary Goppa codes, including binary QD codes, this attack can only correct *half* as many errors as can be introduced and then corrected using the private Goppa trapdoor.

If the underlying QD code were used for encryption, the attacker would have to guess the remaining $t/2$ errors before using the obtained alternant trapdoor. This would mean repeating the attempted decoding $\binom{n}{t/2}/\binom{t}{t/2}$ times, which is clearly infeasible for properly chosen practical parameters. For CFS signatures no guessing is possible, since the messages to be signed are hashed directly onto syndromes, not onto words with errors. Thus the attacker faces the difficulty of finding a syndrome that decodes into a $t/2$-error vector. Such syndromes only occur with exceedingly low density.

We conclude that existing attacks based on solving instances of the MQE problem fail against properly chosen, yet still practical, binary QD codes.

Remark 1. A recent paper by Faugère et al. [8] analyzes the problem of distinguishing binary Goppa codes from random codes. The authors show that, under certain conditions (essentially for the parameters used for signatures), this problem is no longer hard (for binary Goppa codes and binary quasi-dyadic Goppa codes).

6 Conclusion

In this paper, we have presented a new way to instantiate CFS-like signature schemes. The adoption of binary quasi-dyadic (QD) codes allows for a reduction of key sizes by a factor of 4 in practice. Although the number of signing attempts increases by a factor of 2, a proper implementation of the more efficient arithmetic enabled by QD codes is likely to make the actual signing time comparable to plain CFS, possibly faster.

The resulting QD-CFS scheme can be adapted to schemes derived from CFS signatures like [3], [21], or [6]. Binary QD codes can also be applied to other code-based primitives like FSB (hash function), Stern (identification and signature scheme) or SYND (stream cipher). We leave these possibilities for further research.

References

1. T. P. Berger, P.-L. Cayrel, P. Gaborit, and A. Otmani. Reducing key length of the McEliece cryptosystem. In *Progress in Cryptology – Africacrypt'2009*, Lecture Notes in Computer Science, pages 77–97. Springer, 2009.
2. D. J. Bernstein. List decoding for binary Goppa codes. Preprint, 2008. `http://cr.yp.to/papers.html#goppalist`.
3. P.-L. Cayrel, P. Gaborit, D. Galindo, and M. Girault. Improved identity-based identification using correcting codes. *CoRR*, abs/0903.0069, 2009.
4. N. Courtois, M. Finiasz, and N. Sendrier. How to achieve a McEliece-based digital signature scheme. In *Advances in Cryptology – Asiacrypt'2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 157–174, Gold Coast, Australia, 2001. Springer.

5. L. Dallot. Towards a concrete security proof of Courtois, Finiasz and Sendrier signature scheme. Proceedings of WEWoRC 2007, Bochum, Germany, 2007. `http://users.info.unicaen.fr/~ldallot/download/articles/CFSProof-dallot.pdf`.
6. L. Dallot and D. Vergnaud. Provably secure code-based threshold ring signatures. In Matthew G. Parker, editor, *IMA Int. Conf.*, volume 5921 of *Lecture Notes in Computer Science*, pages 222–235. Springer, 2009.
7. J.-C. Faugère, A. Otmani, L. Perret, and J.-P. Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In *Advances in Cryptology – Eurocrypt'2010*, volume 6110 of *LNCS*, pages 279–298. Springer, 2010.
8. Jean-Charles Faugère, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high rate mceliece cryptosystems. Cryptology ePrint Archive, Report 2010/331, 2010. `http://eprint.iacr.org/`.
9. M. Finiasz and N. Sendrier. Security bounds for the design of code-based cryptosystems. In *Advances in Cryptology – Asiacrypt'2009*, volume 5912 of *LNCS*, pages 88–105. Springer, 2009.
10. P. Gaborit. Shorter keys for code based cryptography. In *International Workshop on Coding and Cryptography – WCC'2005*, pages 81–91, Bergen, Norway, 2005. ACM Press.
11. M. N. Gulamhusein. Simple matrix-theory proof of the discrete dyadic convolution theorem. *Electronics Letters*, 9(10):238–239, 1973.
12. K. Kobara. Flexible quasi-dyadic code-based public-key encryption and signature. Cryptology ePrint Archive, Report 2009/635, 2009.
13. R. McEliece. A public-key cryptosystem based on algebraic coding theory. The Deep Space Network Progress Report, DSN PR 42–44, 1978. `http://ipnpr.jpl.nasa.gov/progressreport2/42-44/44N.PDF`.
14. R. Misoczki and P. S. L. M. Barreto. Compact McEliece keys from goppa codes. In *Selected Areas in Cryptography – SAC'2009*, volume 5867 of *LNCS*, pages 276–392. Springer, 2009.
15. H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.
16. A. Otmani, J.-P. Tillich, and L. Dallot. Cryptanalysis of two McEliece cryptosystems based on quasi-cyclic codes. *Mathematics in Computer Science*, 3(2):129–140, 2010.
17. N. J. Patterson. The algebraic decoding of Goppa codes. *IEEE Transactions on Information Theory*, 21(2):203–207, 1975.
18. S. Schechter. On the inversion of certain matrices. *Mathematical Tables and Other Aids to Computation*, 13(66):73–77, 1959. `http://www.jstor.org/stable/2001955`.
19. P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26:1484–1509, 1995.
20. V. G. Umana and G. Leander. Practical key recovery attacks on two McEliece variants. In *International Conference on Symbolic Computation and Cryptography – SCC'2010*, 2010. To appear.
21. D. Zheng, X. Li, and K. Chen. Code-based ring signature scheme. *I. J. Network Security*, 5(2):154–157, 2007.