cryptography

Article

Efficient One-Time Signatures from Quasi-Cyclic Codes: A Full Treatment

Edoardo Persichetti

Department of Mathematical Sciences, Florida Atlantic University, Boca Raton, FL 33431, USA; epersichetti@fau.edu

Received: 11 September 2018; Accepted: 9 October 2018; Published: 12 October 2018

Abstract: The design of a practical code-based signature scheme is an open problem in post-quantum cryptography. This paper is the full version of a work appeared at SIN'18 as a short paper, which introduced a simple and efficient one-time secure signature scheme based on quasi-cyclic codes. As such, this paper features, in a fully self-contained way, an accurate description of the scheme setting and related previous work, a detailed security analysis, and an extensive comparison and performance discussion.

Keywords: post-quantum cryptography; code-based cryptography; digital signatures

1. Introduction

Digital signatures are a very important cryptographic primitive in the modern world. Among the most popular, there are, for instance, schemes based on the RSA assumptions, discrete logarithm (DSA) and its elliptic curves version (ECDSA), all included in the FIPS standard 186-3 [1]. Many schemes based on coding theory have been proposed over the years that either follow a "direct" hash-and-sign approach like the Courtois-Finiasz-Sendrier scheme (CFS) [2] and the Kabatianskii-Krouk-Smeets scheme (KKS) [3], or rely on the Fiat–Shamir transform [4] to convert an identification scheme into a signature scheme. The latter schemes are usually built via a 3-pass protocol [5] or, more recently, a 5-pass protocol [6], in turn relying on the work of Stern [7,8]. Unfortunately, many of the various proposals have been broken, and all those that are still considered secure suffer from one or more flaws, be that a huge public key, a large signature or a slow signing algorithm, which make them highly inefficient in practical situations. This is particularly evident in the identification schemes, where it is usually necessary to repeat the protocol many times in order to guarantee correctness or security.

In [9], we introduced a code-based signature scheme following a different approach, inspired by the work of Lyubashevsky [10,11]. Such a proposal had been attempted before (see [12]) without success, the main issue being the choice of the setting (random binary codes), which proved to be too restrictive. Choosing quasi-cyclic codes allows for taking advantage of the innate ring metric and makes the scheme viable in practice.

Our Contribution

This full version features a detailed security analysis, including a proof of security that guarantees one-time existential unforgeability against chosen-message attacks, i.e., 1-EUF-CMA. While one-time signatures are not used directly in most applications, they are still relevant since they can be embedded in a Merkle tree structure to obtain a full-fledged signature scheme, which allows for signing up to a predetermined number of times. Our scheme compares very well to other one-time code-based proposals, obtaining what are, to date, the smallest sizes for both signature and public data in the code-based setting.

The paper is organized as follows: in the next section, we give some preliminary notions about codes and code-based cryptography, as well as identification schemes. In Section 3, we describe the framework on which our scheme will be based, including the previous code-based proposal by Persichetti. Our scheme is presented in Section 4, together with a detailed security analysis (Section 5), and its performance and comparison with other code-base schemes are discussed in Section 6. We conclude in Section 7.

2. Preliminaries

2.1. Coding Theory

Let $\mathbb{F}_q$ be the finite field with q elements. An $[n,k]$ *linear code* $\mathcal{C}$ is a subspace of dimension k of the vector space $\mathbb{F}_q^n$. Codewords are usually measured in the Hamming metric: the *Hamming weight* of a word $x \in \mathbb{F}_q^n$ is the number of its non-zero positions, and the *Hamming distance* between two words $x, y \in \mathbb{F}_q^n$ is the number of positions in which they differ, that is, the weight of their difference. We denote those respectively by $\mathsf{wt}(x)$ and $\mathsf{d}(x,y)$.

Linear codes can be efficiently described by matrices. The first way of doing this is essentially choosing a basis for the vector subspace. A *generator matrix* is a matrix G that generates the code as a linear map: for each message $x \in \mathbb{F}_q^k$, we obtain the corresponding codeword xG. Of course, since the choice of basis is not unique, so is the choice of generator matrix. It is possible to do this in a particular way, so that $G = (I_k|M)$. This is called *systematic form* of the generator matrix. Alternatively, a code can be described by its *parity-check matrix*: this is nothing but a generator for the *dual code* of $\mathcal{C}$, i.e., the code comprised of all the codewords that are "orthogonal" to those of $\mathcal{C}$. The parity-check matrix describes the code as follows:

$$\forall x \in \mathbb{F}_q^n,\ x \in \mathcal{C} \Longleftrightarrow Hx^T = 0.$$

The product Hx^T is known as *syndrome* of the vector x. Note that, if $G = (I_k|M)$ is a generator matrix in systematic form for $\mathcal{C}$, then $H = (-M^{\intercal}|I_{n-k})$ is a systematic parity-check matrix for $\mathcal{C}$.

Code-based cryptography usually relies more or less directly on the following problem, connected to the parity-check matrix of a code.

Problem 1 (Syndrome Decoding Problem (SDP))**.**
Given: $H \in \mathbb{F}_q^{(n-k)\times n}$, $s \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
Goal: find $e \in \mathbb{F}_q^n$ *with* $\mathsf{wt}(e) \leq w$ *such that* $He^T = s$.

This problem is well known and was proved to be NP-complete by Berlekamp, McEliece and van Tilborg in [13]. Moreover, it is proved that there exists a unique solution to SDP if the weight w is below the so-called *GV Bound*.

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

$$\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i \leq q^{n-k}.$$

If this is not the case, multiple solutions exist (see, for example, Overbeck and Sendrier, [14]). It follows that SDP is of particular interest when the weight w is "small".

Quasi-Cyclic Codes

A special subfamily of linear codes is that of cyclic codes.

Definition 2. *Let* $\mathcal{C}$ *be an* $[n,k]$ *linear code over* $\mathbb{F}_q$. *We call* $\mathcal{C}$ cyclic *if*

$$\forall a = (a_0, a_1 \ldots, a_{n-1}),\ a \in \mathcal{C} \Longrightarrow a' = (a_{n-1}, a_0 \ldots, a_{n-2}) \in \mathcal{C}.$$

Clearly, if the code is cyclic, then all the right shifts of any codeword have to belong to $\mathcal{C}$ as well. An algebraic characterization can be given in terms of polynomial rings. In fact, it is natural to build a bijection between cyclic codes and ideals of the polynomial ring $\mathbb{F}_q[X]/(X^n-1)$. We identify the vector $(a_0, a_1 \ldots, a_{n-1})$ with the polynomial $a_0 + a_1X + \cdots + a_{n-1}X^{n-1}$, and then the right shift operation corresponds to the multiplication by X in the ring.

Because of this correspondence, it is possible to see that both the generator matrix and the parity-check matrix of a cyclic code have a special form, namely *circulant* form, where the i-th row corresponds to the cyclic right shift by i positions of the first row.

Cyclic codes have been shown to be insecure in the context of cryptography, as they introduce too much recognizable structure. A subfamily, known as *quasi-cyclic* codes, has then been proposed with some success, mostly in the context of encryption.

Definition 3. *Let $\mathcal{C}$ be an $[n,k]$ linear code over $\mathbb{F}_q$. We call $\mathcal{C}$ Quasi-Cyclic if there exists n_0 such that, for any codeword a, all the right shifts of a by n_0 positions are also codewords.*

When $n = n_0p$, it is again possible to have both matrices in a special form, composed of n_0 circulant $p \times p$ blocks. The algebra of quasi-cyclic codes can be connected to that of the polynomial ring $\mathbb{F}_q[X]/(X^p-1)$, where each codeword is a length-n_0 vector of elements of the ring.

For the remainder of the paper, we consider only binary codes; thus, we set $\mathcal{R} = \mathbb{F}_2[X]/(X^p-1)$, and we restrict our attention to the case $n_0 = 2$. We have the following ring-based formulation of SDP.

Problem 2 (Quasi-Cyclic Syndrome Decoding Problem (QC-SDP))**.**
Given: $h, s \in \mathcal{R}$ and $w \in \mathbb{N}$.
Goal: find $e_0, e_1 \in \mathcal{R}$ with $\mathsf{wt}(e_0) + \mathsf{wt}(e_1) \leq w$ such that $e_0 + e_1h = s$.

This was shown to be NP-complete in [15]. When $n_0 = 2$, it has been proved in [16] that random quasi-cyclic codes lie on the GV bound with overwhelming probability. Moreover, the impact of cyclicity on SDP has been studied, for example in [17], revealing no substantial gain.

2.2. Identification Schemes and Signatures

An identification scheme is a protocol that allows a party $\mathcal{P}$, the Prover, to prove to another party $\mathcal{V}$, the Verifier, that he possesses some secret information x, usually called *witness*, without revealing to the verifier what that secret information is. The paradigm works as follows: $\mathcal{V}$ is equipped with a public key pk and some public data D. To start, $\mathcal{P}$ chooses some random data y and commits to it by sending $Y = f(y)$ to $\mathcal{V}$, where f is usually a trapdoor one-way function or a hash function. $\mathcal{V}$ then chooses a random challenge c and sends it to $\mathcal{P}$. After receiving c, $\mathcal{P}$ computes a response z as a function of c, x and y and transmits z. Finally, $\mathcal{V}$ checks that z is correctly formed using pk and D.

A signature scheme is defined by a triple $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver})$, respectively the *key generation algorithm*, the *signing algorithm* and the *verification algorithm*. The key generation algorithm KeyGen takes as input a security parameter λ and outputs a signing key sgk and a verification key vk. The private signing algorithm Sign receives as input a signing key sgk and a message m and returns a signature σ. Finally, the public verification algorithm Ver uses a verification key vk to verify a signature σ that is transmitted together with the message m: it outputs 1, if the signature is recognized as valid, or 0 otherwise.

The standard notion of security for digital signatures schemes is Existential Unforgeability under Chosen-Message Attacks (EUF-CMA), as described, for example, in [18]. In this scenario, the goal of an attacker is to produce a valid message/signature pair, and the attack model allows the attacker to obtain a certain, predetermined, number of signatures on arbitrarily chosen messages (signing queries). In particular, if the attacker is only allowed to obtain a single signature, we talk about 1-EUF-CMA security. Since this is the security target of this work, we give a precise definition below.

Definition 4. *An adversary $\mathcal{A}$ is a polynomial-time algorithm that acts as follows:*

1. *Query a key generation oracle to obtain a verification key* vk.
2. *Choose a message m and submit it to a signing oracle. The oracle will reply with $\sigma = \mathsf{Sign}_{\mathsf{sgk}}(m)$.*
3. *Output a pair (m^*, σ^*).*

The adversary succeeds if $\mathsf{Ver}_{\mathsf{vk}}(m^*, \sigma^*) = 1$ and $(m^*, \sigma^*) \neq (m, \sigma)$. We say that a signature scheme is 1-EUF-CMA secure if the probability of success of any adversary $\mathcal{A}$ is negligible in the security parameter, i.e.,

$$\Pr[\mathsf{vk} \overset{\$}{\leftarrow} \mathsf{KeyGen} : \mathsf{Ver}_{\mathsf{vk}}(\mathcal{A}(\mathsf{vk}, \mathsf{Sign}_{\mathsf{sgk}}(m))) = 1] \in negl(\lambda). \tag{1}$$

Fiat and Shamir in [4] showed how to obtain a full-fledged signature scheme from an identification scheme. With this paradigm, the signer simply runs the identification protocol, where, for the purpose of generating the challenge, the verifier is replaced by a random oracle $\mathcal{H}$ (usually a cryptographic hash function). The signature is then accepted according to the validity of the response in the identification scheme. We report this in Table 1.

Table 1. The Fiat–Shamir signature scheme.

Setup	Select an identification scheme $\mathcal{I}$.
Sign	On input the private key of $\mathcal{I}$ and a message m, commit Y, set $c = \mathcal{H}(Y, m)$, compute a response z and return the signature $\sigma = (Y, z)$.
Ver	On input the public key of $\mathcal{I}$, a message m and a signature σ, set $c = \mathcal{H}(Y, m)$ then output 1 if z is accepted in $\mathcal{I}$, else return 0.

Note that several signature schemes, including [11] and this work, use a slightly modified version of the above paradigm, where the signature is (c, z) instead of (Y, z). The verifier can then calculate Y from z and the public key, and check the equality between c and the hash digest obtained using this newly-generated Y and m.

3. A Framework for Signatures

3.1. Number Theory and Lattices

There is a relatively recent approach that provides an easy way to construct efficient signature schemes based on any hard problem. The approach consists of successive reductions building on the original hard problem, first deriving a collision-resistant hash function f, then converting it into a one-time signature where the private key is a pair of integers (x, y), the public key is the pair $(f(x), f(y))$, and the signature of a message m is simply $mx + y$. The one-time signature can then be turned into an identification scheme by replacing m with a challenge c chosen by the verifier and letting y be the commitment (a distinct y is used in every run of the protocol). Finally, the identification scheme is transformed into a full-fledged signature scheme using the Fiat–Shamir transform. Proposals based on classical number theory problems such as RSA or discrete logarithm (see Okamoto [19]) are easy and intuitive to design.

Lyubashevsky showed for the first time how to translate the framework to the lattice case, presenting in [10] an identification scheme which was then refined and updated in [11]. The translation is rather direct, except for an issue which is inherent to the nature of the lattice schemes: unlike factoring or discrete logarithm, in fact, the hardness of lattice problems comes from finding elements that live in a specific *subset* of a ring, namely elements with small Euclidean norm. Transmitting several elements of this nature can leak some parts of the private key. To overcome this limitation, the author makes use of a technique, already introduced in [20], called *aborting*. In short, this consists of rejecting the challenge if in doing so the security of the scheme would be compromised. In practice, this is realized by limiting the set of possible answers to a smaller "safe" subset, consisting of elements whose norm satisfies a certain bound.

3.2. A Coding Theory Scenario

A first, direct translation of the framework to the case of code-based cryptography was proposed by Persichetti in [12]. The idea is for the scheme to rely on SDP, hence featuring a public matrix H, a secret x having weight below the GV bound and the public key $s_x = Hx^T$. Similarly to the lattice case, the final verification should include not only an algebraic formula consisting of H, the commitment Y and s_x, but also a check on the weight of the response z.

Formally, one can see the syndrome computation as a hash function $f(x) = Hx^T$, which is is preimage-resistant provided that the weight of x is small. From now on, we will denote this function as $synd_H(x)$. It follows that the scheme is subject to the additional constraint that the random element y and the challenge c should be chosen such that $\mathsf{wt}(z) \leq w$, where w is the value of the GV distance. This means that c can only be an element of $\mathbb{F}_q$ and that x and y must satisfy $\mathsf{wt}(x) = \gamma_1 w, \mathsf{wt}(y) = \gamma_2 w$, for certain constants $\gamma_1, \gamma_2 \leq 1$ such that $\gamma_1 + \gamma_2 = 1$. In the sample instantiation that we are about to present, we have chosen $\gamma_1 = \gamma_2 = 1/2$ for simplicity. We will also use the notation $\mathcal{D}_a$ to indicate the distribution that samples uniformly at random a vector of $\mathbb{F}_q^n$ of weight less or equal to a. The scheme uses a cryptographic hash function $\mathcal{H}$ as per the Fiat–Shamir paradigm.

KeyGen

Input: parameters $q, n, k, w \in \mathbb{N}$ and an $(n-k) \times n$ parity-check matrix H over $\mathbb{F}_q$.

1. Sample $x \xleftarrow{\$} \mathcal{D}_{w/2}$.
2. The signing key is x.
3. The verification key is $s_x = synd_H(x)$.

Sign

Input: a message m and the signing key x.

1. Sample $y \xleftarrow{\$} \mathcal{D}_{w/2}$.
2. Compute $s_y = synd_H(y)$.
3. Compute $c = \mathcal{H}(m, s_y)$.
4. Compute $z = cx + y$.
5. The signature is $\sigma = (c, z)$.

Ver

Input: a message m, a signature σ and the verification key s_x.

1. Compute $s_z = synd_H(z)$.
2. Use the verification key to compute $v = cs_x + s_z$.
3. Compute $c' = \mathcal{H}(m, v)$.
4. Accept if $c' = c$ and $\mathsf{wt}(z) \leq w$.
5. Else, reject.

Vulnerability from Multiple Signatures

Unfortunately, if used to sign multiple messages, this simple proposal is vulnerable to an attacker who tries to learn the secret. In fact, if an attacker can obtain a polynomial number of signatures, it could store the corresponding values of z and c and then compute $z' = c^{-1}y + x$: this is always possible, since c is a field element and is non-zero. Now, the vector $y' = c^{-1}y$ is randomly generated and has low weight, so each of its coordinates is biased towards 0. Therefore, a simple statistical analysis will eventually reveal all the positions of x. The problem seems to come from the scheme metric itself. In fact, c is constrained to be a field element (to fit the verification equation) but doesn't alter the weight of x, and so the low-weight vector y that is added is not enough to properly hide the secret support.

4. The New Scheme

The core of our idea is to use quasi-cyclic codes in the framework that we have described above. The use of quasi-cyclic codes in cryptography is not a novelty: these have been proposed before in the context of encryption (e.g., [15]). Their originally suggested use (i.e., with GRS codes) was cryptanalyzed in [21] and it is thus not recommended, but other variants based on Low-Density Parity-Check (LDPC) and Moderate-Density Parity-Check (MDPC) codes are still considered safe. In both cases, the issue is that introducing the extra algebraic structure can compromise the secrecy of the private matrix used for decoding.

A big advantage of our proposal is that this issue does not apply. In fact, since there is no decoding involved, an entirely random code can be used, and the code itself is public, so there is no private matrix to hide. In this sense, our scheme is closer, to an extent, to the work of [22], which is centered on *random* quasi-cyclic codes.

As far as signature schemes go, Gaborit and Girault in [23] propose a variant of Stern's ID scheme that uses quasi-cyclic codes (called "double-circulant" by the authors). While this proves to be more efficient than the classical Stern scheme, the protocol still features the same flaw, i.e., a non-trivial cheating probability. This leads to the necessity of repeating the protocol several times, with an obvious impact on the efficiency of the scheme.

In our setting, we use 2-quasi-cyclic codes where words are vectors in $\mathcal{R} \times \mathcal{R}$. For a word $x = (x_0, x_1)$, the syndrome function associated to $h \in \mathcal{R}$ is defined as $synd_h(x) = x_0 + x_1 h$, following the notation that takes a parity-check matrix in systematic form (and hence defined by h) as in Problem 2. For a more general formulation, we also adapt the notation from the previous section, indicating with $\mathcal{D}_1$ and $\mathcal{D}_2$ the distributions that sample uniformly at random vectors of $\mathcal{R} \times \mathcal{R}$ having weight respectively less or equal to $w_1 = \gamma_1 w$ and $w_2 = \gamma_2 w$. Our signature scheme is presented below. The scheme uses a hash function $\mathcal{H}$ that outputs bit strings of fixed weight δ, which is one of the system parameters.

KeyGen

Input: parameters $p, \delta, w_1, w_2 \in \mathbb{N}$ and a vector $h \in \mathcal{R}$.

1. Sample $x \xleftarrow{\$} \mathcal{D}_1$.
2. The signing key is x.
3. The verification key is $s_x = synd_h(x)$.

Sign

Input: a message m and the signing key x.

1. Sample $y \xleftarrow{\$} \mathcal{D}_2$.
2. Compute $s_y = synd_h(y)$.
3. Compute $c = \mathcal{H}(m, s_y)$.
4. Compute $z = cx + y$.
5. The signature is $\sigma = (c, z)$.

Ver

Input: a message m, a signature σ and the verification key s_x.

1. Compute $s_z = synd_h(z)$.
2. Use the verification key to compute $v = cs_x + s_z$.
3. Compute $c' = \mathcal{H}(m, v)$.
4. Accept if $c' = c$ and $\mathrm{wt}(z) \leq w$.
5. Else, reject.

Like before, we have a constraint on the weight of the response vector z: in this case, $w \leq \delta w_1 + w_2$, since c is no longer a constant. Then, w is required to be below the GV bound to ensure that the response

z is the unique solution to the corresponding QC-SDP instance. This is a consequence of the security requirements, as we will see next.

To conclude, note that it is easy to check that an honest verifier always gets accepted. In fact, in an honest run of the protocol, then $v = cs_x + s_z = c \cdot synd_h(x) + synd_h(z)$. Due to the transitivity of the syndrome computation, this is the same as $synd_h(cx+z) = synd_h(y) = s_y$. Therefore, $c' = \mathcal{H}(m, v) = \mathcal{H}(m, s_y) = c$ and the verification is passed.

5. Security

The change of metric in our proposal means that our scheme is substantially different from the "naïve" SDP-based proposal of Section 3.2, and in fact resembles the lattice setting much more. In fact, as in the lattice case, our objects are "vectors of vectors", namely in this case a length-2 vector of length-p binary vectors. Due to the inherent arithmetic associated to the ring $\mathcal{R}$, this allows us to choose c in the same realm, and perform an operation (ring multiplication) that is still compatible with the verification operation, but does affect the weight of the response vector. Polynomial multiplication simultaneously increases and scrambles the error positions, and in so doing prevents the simple attack based on statistical analysis that affected the previous proposal. Unfortunately, this is still not enough to hide the private information. The following procedure [24] shows that it is still possible to recover the private key with a polynomial number of signatures.

Procedure 1. *Start by obtaining a polynomial number ℓ of signatures, i.e., pairs $(c^{(i)}, z^{(i)})$ for $i, = 1, \dots, \ell$. For each pair, $c^{(i)}$ is chosen uniformly at random among the vectors of weight δ, and $z^{(i)} = c^{(i)}x + y^{(i)}$ where $y^{(i)}$ is also chosen uniformly at random (sampled from $\mathcal{D}_2$). For each i, write $c^{(i)} = X^{i_1} + \dots + X^{i_\delta}$, that is, as a polynomial of weight δ in $\mathcal{R}$. Then, calculate*

$$\begin{aligned} z^{(i,j)} &= X^{-i_j} z^{(i)} \pmod{X^p - 1} \\ &= X^{-i_j}(c^{(i)}x + y^{(i)}) \pmod{X^p - 1} \\ &= (1 + \sum_{k \neq j, k \in \{1,\dots,\delta\}} X^{i_k - i_j})x + X^{-i_j} y^{(i)} \pmod{X^p - 1} \\ &= x + \sum_{k \neq j, k \in \{1,\dots,\delta\}} x^{(i,j)} + y^{(i,j)} \pmod{X^p - 1}, \end{aligned}$$

where $x^{(i,j)} = X^{i_k - i_j} x \pmod{X^p - 1}$ and $y^{(i,j)} = X^{-i_j} y^{(i)} \pmod{X^p - 1}$.

Since $x^{(i,j)}$ is just a shift of x and $y^{(i,j)}$ is just a shift of $y^{(i)}$, and their support will likely have little to no intersection with the support of x (due to the weight of the vectors), it is possible to reveal the support of x simply by looking at the bits that belong to the support of a large enough number of $z^{(i,j)}$.

Note that the above procedure is in fact a refinement of the simple statistical analysis attack encountered before: in both cases, the problem is that the weight of the vectors is simply too low to properly mask the private vector. It is then clear that it is impossible to sign multiple times and preserve security. It follows that our scheme only achieves one-time security. To prove the one-time security of our scheme, we follow the paradigm for a generic one-time signature scheme of Pointcheval and Stern, which was already employed in the code-based setting in [25]. In this paradigm, signature schemes are treated in a unified way, as a protocol that outputs triples of the form (σ_1, h, σ_2), where σ_1 represents the commitment, or a sequence of commitments, if the protocol needs to be repeated multiple times. σ_2 is the response, or a sequence of responses, and h is the hash value, as in the Fiat–Shamir scheme. To obtain security, it is necessary that σ_1 is sampled uniformly at random from a large set and that σ_2 only depends on σ_1, the message m and the hash value h.

In our scheme, the first element $\sigma_1 = s_y$ is sampled uniformly at random from $\mathcal{D}_2$, which has size $\binom{n}{w_2}$. Note that, even though this value is not explicitly output as part of the signature, it is immediate to recover it from the signature, as shown in Step 2 of the verification algorithm. The vector c is exactly the hash value obtained from the message m and σ_1, i.e., the element h in the Pointcheval–Stern notation. We clearly use

c from now on, to avoid confusion as h is used to denote the vector defining the parity-check matrix in a QC code. Finally, we show that $\sigma_2 = z$ indeed only depends on the message m, σ_1 and c. The dependence is obvious, given that z is computed using only the private key, c itself and y, which is in a one-to-one correspondence with s_y (due to w_2 being below the GV bound). Furthermore, z is uniquely determined by those values. In fact, suppose there existed a distinct valid triple (s_y, c, z') with $z' \neq z$. Since the triple is valid, it needs to satisfy the verification equation, thus $synd_h(z') = cs_x + s_y = s_z$. This is clearly not possible because both z and z' have weight below the GV bound, which implies that there exists only one vector having syndrome s_z, i.e., $z' = z$.

The next step is to show that, in our signature scheme, it is possible to simulate the target triples without knowing the private key, unbeknownst to the adversary.

Lemma 1. *It is possible to obtain artificially-generated triples of the form (s_y, c, z) which are indistinguishable from honestly-generated triples, unless the adversary is able to solve an instance of QC-SDP.*

Proof. To begin, notice that any valid triple is required to satisfy two constraints. First, the weight of z has to be below the GV bound; in fact, $\mathsf{wt}(z)$ is expected to be statistically close to the bound $w \leq w_2 + \delta w_1$. Second, the triple needs to pass the verification equation, and so $s_y = cs_x + s_z$. Then, to simulate a valid triple, it is enough to sample two elements at random and set the third to match. More precisely, one would sample $c \overset{\$}{\leftarrow} \mathcal{D}_c$ and $z \overset{\$}{\leftarrow} \mathcal{R}^2$, the second one chosen such that $\mathsf{wt}(z) \approx w$. Then, one would proceed by setting s_y to be exactly $cs_x + s_z$, which is possible since the public key s_x is known.

Now, it is easy to see that all honestly-generated triples correspond to syndromes $s_y = synd_h(y)$ where y has weight w_2 below the GV bound, while, for simulated triples, the syndrome s_y is obtained from a vector $y = cx + z$ which has expected weight *above* the GV bound with overwhelming probability. This is because both c and z are generated independently and at random, and so the expected weight is simply $\delta w_1 + \mathsf{wt}(z)$, which is bigger than the bound with overwhelming probability.

In conclusion, distinguishing a simulated triple from an honest one corresponds to solving a QC-SDP instance as claimed. □

The last piece necessary for our proof is the well-known *forking lemma*. We report it below, as formulated in [26].

Theorem 1 (General Forking Lemma). *Let $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme with security parameter λ. Let $\mathcal{A}$ be an adversary, running in time T and performing at most q random oracle queries and ℓ signing queries. Suppose that $\mathcal{A}$ is able to produce a valid signature $(m, \sigma_1, h, \sigma_2)$ with probability $\varepsilon \geq 10(\ell + 1)(\ell + q)/2^\lambda$. If the triples (σ_1, h, σ_2) can be simulated without knowing the private key with only a negligible advantage for $\mathcal{A}$, then there exists a polynomial-time algorithm $\mathcal{B}$ that can simulate the interaction with $\mathcal{A}$ and is able to produce two valid signatures $(m, \sigma_1, h, \sigma_2)$ and $(m, \sigma_1, h', \sigma_2')$, for $h' \neq h$, in time $T' \leq 120686qT/\varepsilon$.*

We are now ready for our security result.

Theorem 2. *Let $\mathcal{A}$ be a polynomial-time 1-EUF-CMA adversary for the signature scheme with parameters p, δ, w_1, w_2, running in time T and performing at most q random oracle queries. Let the probability of success of $\mathcal{A}$ be $\varepsilon \geq 20(q + 1)/2^\lambda$. Then, the QC-SDP problem with parameters $n = 2p, w = \delta w_1 + w_2$ can be solved in time $T' \leq 120,686\ell qT/\varepsilon$.*

Proof. We have seen in Procedure 1 that it is possible to recover the private key using a polynomial number ℓ of signatures. The forking lemma can be iterated so that it is guaranteed to produce ℓ distinct, valid signatures in time less or equal to $T' \leq 120686\ell qT/\varepsilon$. The thesis naturally follows from the combination of these two facts. □

6. Performance and Comparison

To properly evaluate the performance, we start by recalling the main components of our scheme. First of all, the public data consists of the vector h (of length p) and the syndrome s_x (also of length p), for a total of $2p$ bits. The signature, on the other hand, is given by the challenge string c and the response z. In our scheme, this corresponds respectively to a vector of length p and a vector of length $2p$. It is possible to greatly reduce this size thanks to a storing technique [27] which allows for representing low-weight vectors in a compact manner. Namely, a binary vector of length n and weight w is represented as an index, plus an indication of the actual vector weight, for a total of $\log\binom{n}{w} + \log(w)$. Note that in our case this applies to both c and z.

We now provide (Table 2) some parameters for the codes in our scheme. These are normally evaluated with respect to general decoding algorithms such as Information-Set Decoding [28–32]: the amount of security bits is indicated in the column "Security".

Table 2. Parameters (all sizes in bits).

p	w_1	w_2	δ	Security (λ)	Public Data	Signature Size
4801	90	100	10	80	9602	4736
9857	150	200	12	128	19714	9475
3072	85	85	7	80	6144	3160
6272	125	125	10	128	12544	6368

The first two rows report well-known parameters suggested in the literature for QC-MDPC codes; however, since our codes do not need to be decodable, we are able to slightly increase the number of errors introduced. The last two rows, instead, are parameters chosen ad hoc, in order to optimize performance.

6.1. Existing Code-Based Solutions

We are now going to briefly discuss the three main approaches to obtain code-based signatures, and related variants. This will give an insight into why designing an efficient code-based signature scheme is still an open problem.

6.1.1. CFS

The CFS scheme [2] follows the "hash and sign" paradigm, which is a very natural approach for code-based cryptography, and thus it retains most of its traits, both good and bad. For instance, the verification consists of a single matrix-vector multiplication and so it is usually very fast. On the other hand, the scheme features a very large public key (the whole parity-check matrix). Structured instances as proposed for example in [33] reduce this size drastically and are therefore able to deal with this issue, although with a potential few security concerns. However, the main downfall of CFS is the extremely slow signing time. This is a consequence of the well-known fact that a random word is in general *not* decodable, thus finding a decodable syndrome requires an incredibly high number of attempts (at least 2^{15} in the simplest instances). To lower this number, the common solution is to use codes with very high rate, which in itself could lead to potential insecurities (e.g., the distinguisher of). Thus, it seems unrealistic to obtain an efficient signature scheme in this way.

6.1.2. KKS

The KKS approach [3] still creates signatures in a "direct" way, but without decoding. Instead, the scheme relies on certain aspects of the codes such as a carefully chosen distance between the codewords, and uses a secret support. Unfortunately, the main drawback of KKS-like schemes is the security. In fact, it has been shown in [34] that most of the original proposals can be broken after recovering just a few signatures. Furthermore, not even a one-time version of the scheme (e.g., [25]) is secure, as shown by Otmani and Tillich [35], who are able to break all proposals in the literature without needing to know any message/signature pair. It is therefore unlikely that the KKS approach could be suitable for a credible code-based signature scheme.

6.1.3. Identification Schemes

All of the code-based identification schemes proposed so far are 3-pass (or 5-pass) schemes with multiple challenges. Thus, the prover sends two or three entirely different responses depending on the value of the challenge (usually a bit or {0,1,2}). In this sense, our proposal represents a big novelty. In fact, multiple challenges allow for a malicious user to be able to cheat in some instances. For example, in the original proposal by Stern [7], it is possible to choose any two out of three possible responses and pass verification for those even without knowing the private key, thus leading to a cheating probability of 2/3. This cheating probability is subsequently lowered in most recently proposals, approaching 1/2. Nevertheless, this causes a huge issue, since the protocol needs to be repeated several times in order for an honest prover to be accepted. The 35 repetitions of the original scheme can be lowered to approximately 16 repetitions in recent variants, but, even so, communication costs prove to be very high, leading to a very large signature size. In Table 3 below, we report a comparison of parameters for different variants of the scheme, where the column Véron refers to [5], CVE to [6] and AGS to [36]. Note that all of these parameters refer to a cheating probability of 2^{-16}, a weak authentication level.

Table 3. Comparison of the most popular code-based identification schemes. All the sizes are expressed in bits.

	Stern 3	**Stern 5**	**Véron**	**CVE**	**AGS**
Rounds	28	16	28	16	18
Public Data	122,500	122,500	122,500	32,768	350
Private Key	700	4900	1050	1024	700
Public Key	350	2450	700	512	700
Total Communication Cost	42,019	62,272	35,486	31,888	20,080

In the latest proposal (column AGS), the size of the public matrix is considerably smaller thanks to the use of double-circulant codes. However, the signature size is still very large (about 93 Kb). Moreover, for a signature to be considered secure, one would expect computational costs to produce a forgery to be no less than 2^{80}; this would require, as claimed by the authors in [36], to multiply all the above data by 5, producing even larger sizes.

6.2. Comparison

A comparison of our scheme with the full-fledged schemes described above would not be entirely accurate. We can however compare (Table 4) our scheme to other code-based proposals that are one-time secure, such as [25,37]. Both of these schemes follow the KKS approach, and therefore come with some potential security concerns, as mentioned in the previous section. For simplicity, we will refer to [25] as BMS and to [37] as GS. Note that the latter comes in two variants, which use respectively quasi-cyclic codes, and a newly-introduced class of codes called "quadratic double-circulant" by the authors. All the parameters and sizes (in bits) are reported in the following table, and correspond to a security level of 2^{80}.

Table 4. Comparison of code-based one-time signature schemes. All the sizes are expressed in bits.

	BMS	**GS 1**	**GS 2**	**Our Scheme**
Public Data	930,080	75,000	17,000	6144
Signature Size	3739	18,900	7000	3160

It is immediate to notice that our scheme presents the smallest amount of public data (which groups together public key and any additional public information) and the smallest signature size. To be fair, the BMS scheme employs the same indexing trick used in this work, while this is not the case for the other scheme. Since the signature of the GS scheme (in both variants) also includes a low-weight vector, we expect that it would be possible to apply the same technique to the GS scheme as well, with the obvious reduction in size. We did not compute this explicitly, but it is plausible to assume it would be very close to that of our scheme. Nevertheless, the size of the public data remains much larger even in the most aggressive of the two variants (GS 2).

6.3. Implementation

To confirm the practicality of our scheme, we have developed a simple implementation in C. The implementation is a straightforward translation to C with the addition of the steps for generating public and private keys. The hash function used was SHA-256. We ran the protocol on a small microprocessor, namely a 580 MHz single-core MIPS 24KEc. The choice of this microprocessor was made based on the usage of it, since this type of microprocessor is commonly used in the Internet of Things (IoT) applications. The measurements are reported in Table 5 below.

Table 5. Implementation results.

p	w_1	w_2	δ	**Key Generation (sgk/vk)**	**Signing**	**Verification**
4801	90	100	10	0.061 ms/22.754 ms	89.665 ms	22.569 ms
9857	150	200	12	0.169 ms/104.655 ms	374.206 ms	99.492 ms
3072	85	85	7	0.052 ms/14.017 ms	35.150 ms	14.271 ms
6272	125	125	10	0.116 ms/67.972 ms	150.063 ms	42.957 ms

Note that key generation is dominated by the syndrome computation necessary to obtain the verification key, while sampling the signing key has a negligible cost. The signing operation is the most expensive, which makes sense, while the verification is of the same order of magnitude as the key generation. Both signing and verification algorithm are relatively fast but could be sped up even further, since the hash function used was, at the time the measurements were taken, not optimized to run in such a small device.

7. Conclusions

In this paper, we have presented a new construction for a one-time signature scheme based on coding theory assumptions. In particular, our scheme uses quasi-cyclic codes and relies on the hardness of the quasi-cyclic version of the syndrome decoding problem (QC-SDP), while making use of the inherent ring structure for its arithmetic properties. Quasi-cyclic codes allow for a compact description, and a drastic reduction in the public key size, resulting in a very lightweight scheme. In addition, the ring arithmetic, similar to Lyubashevsky's lattice-based proposal, is very efficient, and we expect to obtain extremely fast and practical implementations. Thanks to all these features, as well as the simplicity of its design, our protocol is very competitive: it features a compact public key, fast signing and verification algorithms, and the signature size is much shorter than other one-time secure code-based protocols. In particular, the protocol is naturally very appealing in lightweight applications, where resources are limited and aspects such as execution time and memory requirements are of crucial importance. Examples could be embedded devices such as microprocessors, or the Internet-of-Things (IoT). Moreover, our scheme could be a very efficient solution for protocols that require only one-time signatures as building blocks, such as the work of [38] based on the *k*-repetition paradigm.

In summary, we believe that our proposal represents a very interesting solution per se, as well as an important step forward in the long quest for an efficient code-based signature scheme.

Funding: This research received no external funding.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Locke, G.; Gallagher, P. *FIPS PUB 186-3: Digital Signature Standard (DSS)*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2009.
2. Courtois, N.; Finiasz, M.; Sendrier, N. How to Achieve a McEliece-Based Digital Signature Scheme. In *ASIACRYPT*; Boyd, C., Ed.; Springer: Berlin, Germany, 2001; Volume 2248, pp. 157–174.
3. Kabatianskii, G.; Krouk, E.; Smeets, B.J.M. *A Digital Signature Scheme Based on Random Error-Correcting Codes*; Darnell, M., Ed.; Springer: Berlin, Germany, 1997; Volume 1355, pp. 161–167.
4. Fiat, A.; Shamir, A. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *CRYPTO*; Odlyzko, A.M., Ed.; Springer: Berlin, Germany, 1986; Volume 263, pp. 186–194.
5. Véron, P. Improved identification schemes based on error-correcting codes. *Appl. Algebra Eng. Commun. Comput.* **1996**, *8*, 57–69.
6. Cayrel, P.L.; Véron, P.; Alaoui, S.M.E.Y. A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem. In *Selected Areas in Cryptography*; Biryukov, A., Gong, G., Stinson, D.R., Eds.; Springer: Berlin, Germany, 2010; Volume 6544, pp. 171–186.
7. Stern, J. A New Identification Scheme Based on Syndrome Decoding. In *CRYPTO*; Stinson, D.R., Ed.; Springer: Berlin, Germany, 1993; Volume 773, pp. 13–21.
8. Stern, J. Designing Identification Schemes with Keys of Short Size. In *CRYPTO*; Desmedt, Y., Ed.; Springer: Berlin, Germany, 1994; Volume 839, pp. 164–173.
9. Persichetti, E. *Efficient One-Time Signatures from Quasi-Cyclic Codes*; ACM: New York, NY, USA, 2018.
10. Lyubashevsky, V. Fiat–Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures. In *International Conference on the Theory and Application of Cryptology and Information Security*; Matsui, M., Ed.; Springer: Berlin, Germany, 2009; Volume 5912, pp. 598–616.
11. Lyubashevsky, V. Lattice signatures without trapdoors. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin, Germany, 2012; pp. 738–755.
12. Persichetti, E. Improving the Efficiency of Code-Based Cryptography. Ph.D. Thesis, University of Auckland, Auckland, New Zealand, 2012.
13. Berlekamp, E.; McEliece, R.; van Tilborg, H. On the inherent intractability of certain coding problems. *IEEE Trans. Inf. Theory* **1978**, *24*, 384–386. [CrossRef]
14. Overbeck, R.; Sendrier, N. Code-based cryptography. In *Post-Quantum Cryptography*; Bernstein, D.J., Buchmann, J., Dahmen, E., Eds.; Springer: Berlin, Germany, 2009; pp. 95–145.
15. Berger, T.P.; Cayrel, P.L.; Gaborit, P.; Otmani, A. Reducing Key Length of the McEliece Cryptosystem. In *International Conference on Cryptology in Africa*; Preneel, B., Ed.; Springer: Berlin, Germany, 2009; Volume 5580, pp. 77–97.
16. Chen, C.L.; Peterson, W.W.; Weldon, E.J. Some results on quasi-cyclic codes. *Inf. Control* **1969**, *15*, 407–423. [CrossRef]
17. Chabot, C.; Legeay, M. Using automorphisms group for decoding. In Proceedings of the 12th International workshop on Algebraic and Combinatorial Coding Theory (ACCT 2010), Novosibirsk, Russia, 5–11 September 2010.
18. Goldwasser, S.; Micali, S.; Rivest, R.L. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.* **1988**, *17*, 281–308. [CrossRef]
19. Okamoto, T. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. In *Annual International Cryptology Conference*; Brickell, E.F., Ed.; Springer: Berlin, Germany, 1992; Volume 740, pp. 31–53.
20. Lyubashevsky, V. Lattice-Based Identification Schemes Secure Under Active Attacks. In *Public Key Cryptography*; Cramer, R., Ed.; Springer: Berlin, Germany, 2008; Volume 4939, pp. 162–179.

21. Faugère, J.C.; Otmani, A.; Perret, L.; Tillich, J.P. Algebraic Cryptanalysis of McEliece Variants with Compact Keys. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Gilbert, H., Ed.; Springer: Berlin, Germany, 2010; Volume 6110, pp. 279–298.
22. Aguilar, C.; Blazy, O.; Deneuville, J.C.; Gaborit, P.; Zémor, G. Efficient Encryption from Random Quasi-Cyclic Codes. *arXiv* **2016**, arXiv:1612.05572.
23. Gaborit, P.; Girault, M. Lightweight code-based identification and signature. In Proceedings of the 2007 IEEE International Symposium on Information Theory, Nice, France, 24–29 June 2007; pp. 191–195.
24. Tillich, J.P. (INRIA, Paris, France). Personal communication, 2018.
25. Barreto, P.S.L.M.; Misoczki, R.; Simplicio, M.A., Jr. One-time signature scheme from syndrome decoding over generic error-correcting codes. *J. Syst. Softw.* **2011**, *84*, 198–204. [CrossRef]
26. Pointcheval, D.; Stern, J. Security arguments for digital signatures and blind signatures. *J. Cryptol.* **2000**, *13*, 361–396. [CrossRef]
27. Ruskey, F. Combinatorial generation. In *Preliminary Working Draft*; University of Victoria: Victoria, BC, Canada, 2003; Volume 11, p. 20.
28. Bernstein, D.J.; Lange, T.; Peters, C. Attacking and Defending the McEliece Cryptosystem. In *PQCrypto*; Buchmann, J., Ding, J., Eds.; Springer: Berlin, Germany, 2008; Volume 5299, pp. 31–46.
29. May, A.; Meurer, A.; Thomae, E. Decoding Random Linear Codes in $\mathcal{O}(2^{0.054n})$. In *ASIACRYPT*; Lee, D.H., Wang, X., Eds.; Springer: Berlin, Germany, 2011; Volume 7073, pp. 107–124.
30. May, A.; Ozerov, I. On computing nearest neighbors with applications to decoding of binary linear codes. In *Advances in Cryptology—EUROCRYPT* 2015; Springer: Berlin, Germany, 2015.
31. Prange, E. The use of information sets in decoding cyclic codes. *IRE Trans. Inf. Theory* **1962**, *8*, 5–9. [CrossRef]
32. Stern, J. A method for finding codewords of small weight. In *Coding Theory and Applications*; Cohen, G.D., Wolfmann, J., Eds.; Springer: Berlin, Germany, 1988; Volume 388, pp. 106–113.
33. Barreto, P.S.L.M.; Cayrel, P.L.; Misoczki, R.; Niebuhr, R. Quasi-Dyadic CFS Signatures. In *Information Security and Cryptology: 6th International Conference*; Lai, X., Yung, M., Lin, D., Eds.; Springer: Berlin, Germany, 2011; pp. 336–349.
34. Cayrel, P.L.; Otmani, A.; Vergnaud, D. On Kabatianskii-Krouk-Smeets Signatures. In *International Workshop on the Arithmetic of Finite Fields*; Carlet, C., Sunar, B., Eds.; Springer: Berlin, Germany, 2007; Volume 4547, pp. 237–251.
35. Otmani, A.; Tillich, J.P. An Efficient Attack on All Concrete KKS Proposals. In *International Workshop on Post-Quantum Cryptography*; Yang, B.Y., Ed.; Springer: Berlin, Germany, 2011; pp. 98–116.
36. Melchor, C.A.; Gaborit, P.; Schrek, J. A new zero-knowledge code based identification scheme with reduced communication. In Proceedings of the 2011 IEEE Information Theory Workshop, Paraty, Brazil, 16–20 October 2011.
37. Gaborit, P.; Schrek, J. Efficient code-based one-time signature from automorphism groups with syndrome compatibility. In Proceedings of the 2012 IEEE International Symposium on Information Theory Proceedings, Cambridge, MA, USA, 1–6 July 2012; pp. 1982–1986.
38. Persichetti, E. On a CCA2-secure variant of McEliece in the standard model. *IACR Cryptol. ePrint Arch.* **2012**, *2012*, 268.