Efficient Digital Signatures From Coding Theory

Edoardo Persichetti

Florida Atlantic University

Abstract. The design of an efficient code-based signature scheme is by all means still an open problem. In this paper, we propose a simple and efficient scheme following the framework detailed by Lyubashevsky in [15] to construct an identification scheme. The scheme is based on quasi-cyclic codes and, while security relies on the ring algebra that is associated with them, the proposal benefits from the quasi-cyclic structure in reducing key and signature sizes.

1 Introduction

Digital signatures are a very important cryptographic protocol 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 [13]. Many schemes based on coding theory have been proposed over the years, that either follow a "direct" approach like CFS (Courtois, Finiasz and Sendrier [8]) and KKS (Kabatianskii, Krouk and Smeets [12]), or rely on the Fiat-Shamir transform [11] to convert a zero-knowledge identification scheme into a signature scheme. The latter schemes are usually built via a 3-pass protocol (Véron [27]) or, more recently, a 5-pass protocol (Cayrel, Véron and El Yousfi [7]), in turn relying on the work of Stern [25,26]. Unfortunately, all of the above proposals suffer from one or more flaws, be that a huge public key, a large signature or a slow signing algorithm, all of 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 this paper, we propose a code-based identification scheme that follows a different approach, inspired by the work of Lyubashevsky [15]. Such a proposal had been attempted before (see [22]) without success, the main issue being the restrictive choice of the setting (random binary codes). Choosing quasi-cyclic codes allows to take advantage of the innate ring metric and makes the scheme viable.

1.1 Identification Schemes and Fiat-Shamir

We briefly recall the notion of *Zero-Knowledge Identification Scheme*. This is a protocol that allows one party, called the Prover, to prove to another party, the Verifier, that he possesses some secret information, without revealing to the verifier what that secret information is. The paradigm works as follows: suppose that the prover $\mathcal{P}$ wants to prove to the verifier $\mathcal{V}$ the knowledge of some secret information s; $\mathcal{V}$ is equipped with a public key pk and the 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 s, c and y and transmits z. Finally, $\mathcal{V}$ checks that z is correctly formed with the help of pk and D. A typical example is the Feige-Fiat-Shamir identification scheme [10], based on the Quadratic Residuosity (QR) hard problem. An accurate description is given in Appendix A.

Security of zero-knowledge identification schemes has to take into account two different aspects. First of all, a protocol should be secure against *impersonators*, that is, a (usually malicious) parties that try to replace the prover and produce a response that is accepted as valid without the knowledge of s. Moreover, the protocol should satisfy *zero-knowledge*, preventing an attacker to extract information about the secret s from the protocol; in this sense, even an honest verifier is considered an adversary for the scheme. Both adversaries are at least (*passive* attacks) allowed to have access not only to the public key and the public data, but also to the information exchanged during any number of interactions between the prover and the verifier. In the *active* model, instead, an adversary is also allowed to interact, playing the role of the verifier in order to extract some information, before attempting to produce a valid response.

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

2 Preliminaries

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 codeword is the number of its nonzero positions, and the *Hamming Distance* between two codewords is the number of positions in which they differ, that is, the weight of their difference.

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* 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 largely relies 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 [4]. 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, [21]). It follows that SDP is meaningful only if the weight w is *small.*

2.1 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.7

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 generator matrix and 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 been then 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 possible to again have both generator matrix and parity-check matrix in a special form, composed of n_0 circulant $p \times p$ blocks. Then, 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 will focus only on the binary case, thus we set $\mathcal{R} = \mathbb{F}_2[x]/(x^p - 1)$. We then have the following ring-based formulation of a quasi-cyclic version 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 $e_0 + e_1 h = S$.

This was shown to be NP-complete in [3].

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 c is simply $cx+y$. The one-time signature can then be turned into a zero-knowledge identification scheme by replacing c with a challenge 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 (see Appendix B) or discrete logarithm (see Okamoto [19]) are easy and intuitive to design.

Lyubashevsky in [15] showed for the first time how to translate the framework to the lattice case. 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 [14], called *aborting*. In short, this consists of refusing to answer to 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. Details are given also in Appendix B. To further clarify the scheme, we present below the proposed parameters for the scheme ([15, Fig. 2]).

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 [22]. The idea is for the scheme to rely on SDP, hence featuring a public matrix H, a secret s having weight below the GV bound and the public key $S = Hs^T$. Similarly to the lattice case, the final verification should include not only an algebraic formula consisting of H, the commitment Y and S, 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. 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 s and y must satisfy $\mathsf{wt}(s) = \gamma_1 w, \mathsf{wt}(y) = \gamma_2 w$, for certain constants $\gamma_1, \gamma_2 \leq 1$ such that $\gamma_1 + \gamma_2 = 1$. We report below the sample instantiation proposed in [22] (where $\gamma_1 = \gamma_2 = 1/2$).

Table 1: SDP-based Identification Scheme.

Public Data	The parameters $q, n, k, w \in \mathbb{N}$ and an $(n-k) \times n$ parity-check matrix H over $\mathbb{F}_q$.
Private Key	$s \in \mathbb{W}_{q,n,w/2}$.
Public Key	$S = Hs^T$.

PROVER		VERIFIER
Choose $y \xleftarrow{\$} \mathbb{W}_{q,n,w/2}$ and set $Y = Hy^T$.	$\xrightarrow{Y}$	
	$\xleftarrow{c}$	$c \xleftarrow{\$} \mathbb{F}_q \setminus \{0\}$.
Compute $z = y + cs$.	$\xrightarrow{z}$	Accept if $Hz^T = Y + cS$ and $\mathsf{wt}(z) \leq w$.

Unfortunately, this simple proposal is vulnerable to an attacker who tries to learn the secret, even in the passive model. In fact, such an attacker could simply run the protocol several times, storing the values of z and c at every run of the protocol, then computing $z' = c^{-1}y + s$: 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 s. The problem seems to come from the scheme setting itself. In fact, c is constrained to be a field element (to fit the verification equation) but doesn't alter the weight of s, 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 instead of generic codes. The use of quasi-cyclic codes in cryptography is not a novelty: these have been proposed before in the context of encryption (e.g. [3]), mainly with the aim of reducing public-key length. Their originally suggested use (i.e. with GRS codes) was cryptanalyzed in [9] and it is thus not recommended, but other variants based on LDPC and 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. Our scheme uses the simplest and most popular instantiation of quasi-cyclic codes ($n_0 = 2$ so just two blocks), and we present it below.

Table 2: Cyclic Identification Scheme.

Public Data	Parameters $p, w, w_1, w_2 \in \mathbb{N}$, a vector $h \in \{0,1\}^p$ and a hash function $\mathcal{H}$.
Private Key	$s = (s_0, s_1) \in \mathcal{R} \times \mathcal{R}$ of weight w_1.
Public Key	$S = s_0 + s_1 h$.

PROVER		VERIFIER
Choose $y = (y_0, y_1) \xleftarrow{\$} \mathcal{R} \times \mathcal{R}$ of weight w_2 set $Y = y_0 + y_1 h$ and $K = \mathcal{H}(Y)$.	$\xrightarrow{K}$	
	$\xleftarrow{c}$	$c \xleftarrow{\$} \mathcal{R}$ of weight $\hat{\delta} \leq \delta$.
If c is not invertible set $z = \perp$ (abort). Compute $z = y + cs$.	$\xrightarrow{z}$	Accept if $\mathcal{H}(z_0 + z_1 h + cS) = K$ and $\mathsf{wt}(z) \leq w$.

In the verification step we have $z_0 = y_0 + cs_0$ and $z_1 = y_1 + cs_1$ thus $z_0 + z_1 h = y_0 + cs_0 + (y_1 + cs_1)h = y_0 + y_1 h + c(s_0 + s_1 h) = Y + cS$ from which it is easy to see that an honest prover will always succeed.

The final weight $w = w_2 + \hat{\delta} w_1$ should be below the GV bound to ensure that the response z is unique, as usual in SDP instances. The restriction on c being invertible and the use of the hash function $\mathcal{H}$ are both connected to the security of the scheme and their role will be clarified in the next section.

4.1 Security

The security of our identification scheme is directly connected to the QC-SDP problem. In fact, in the following theorem, we show how a successful forger against the scheme can be used to solve an instance of QC-SDP.

Theorem 1. *If the identification scheme described in Table 2 is insecure against active attacks, then there exists a polynomial-time algorithm that can solve QC-SDP.*

Proof. Let $\mathcal{A}$ be a successful forger against the scheme, and let (h^*, S^*, w^*) be the target instance of QC-SDP that we want to solve. We will use $\mathcal{A}$ to solve QC-SDP as follows. We start by setting up an instance of the identification scheme.

Key Generation: On input the instance (h^*, S^*, w^*) of QC-SDP, set $h = h^*$, $w_1 = w^*$, and $S = S^*$.

We then play the attack game with $\mathcal{A}$, in which we replace the hash function $\mathcal{H}$ with a random oracle. In the first part of the game, $\mathcal{A}$ impersonates the verifier and we simulate the behavior of an honest prover (since we don't know the private key), including a simulation of the random oracle; we will need to use two tables T_1 and T_2 to ensure our simulation is consistent.

Commitment Phase:

1. Generate a random string y of weight w_2.
2. Look up y in T_1. If (y, Y, r) is found (i.e. the commitment y had been previously used), return r.
3. Else compute $Y = y_0 + y_1 h$, then generate a random "hash" value r and place (y, Y, r) in T_1.
4. Return r.

Response Phase: On input a challenge c of weight $\hat{\delta} \leq \delta$:

1. Look up c in T_2. If (c, u) is found, use the corresponding value for u.
2. Else generate a random vector u of weight $\leq \hat{\delta} w^*$.
3. Set $z = y + u$ and place (c, u) in T_2.
4. Return z.

Random Oracle Queries: Upon the query $Y' = z_0 + z_1 h + cS$:

1. Look up $Y' = z_0 + z_1 h + cS$ in T_1. If (y', Y', r') is found, output $s = c^{-1}(z + y')$ and the game ends.
2. Else return r.

In fact if the value $Y' = z_0 + z_1 h + cS$ had been previously used, then $z_0 + z_1 h = Y' + cS$, which means we have found a vector of weight $w \leq w_2 + \hat{\delta} w_1$ that solves the QC-SDP instance $(h, Y' + cS, w)$. But $y' + cs$ is one such vector, and since the solution of QC-SDP is unique when w is below the GV bound, it must be that $z = y' + cs$. Then $s = c^{-1}(z + y')$ is the solution to (h^*, S^*, w^*).

If the game is not interrupted as above, we move on to the second part. Now, $\mathcal{A}$ impersonates the prover and aims at being successfully verified.

Random Oracle Queries: Upon the query Y:

1. Look up Y in T_1. If (y, Y, r) is found, return r.
2. Else generate a random "hash" value r and place $(\,\cdot\,, Y, r)$ in T_1.
3. Return r.

Challenge Phase: On input a commitment Y:

1. Generate a random c of weight $\leq \hat{\delta}$.
2. Return c.

In the second part of the game, we proceed as follows: after sending the first challenge $c^{(1)}$, we store this value and the value $z^{(1)}$ that we receive from $\mathcal{A}$. Now, since $\mathcal{A}$ can successfully be verified, either it found a preimage for the hash function, or it correctly guessed the value of the response. We can rule out the former since we are correctly simulating the behavior of the hash function, which is meant to be preimage-resistant;

therefore it must be that $z^{(1)} = y + c^{(1)}s$. At this point, we rewind and we repeat the challenge phase, sending a new, distinct value $c^{(2)}$ and receiving another response $z^{(2)}$ which, as before, will correspond to $y + c^{(2)}s$. Then, $z^{(1)} + z^{(2)} = (c^{(1)} + c^{(2)})s$, which will reveal[1] s, and we win the game. □

4.2 About Zero-Knowledge

Note that our proposals differs substantially from the "naïve" SDP-based proposal, and in fact it 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, yet provides more security. In fact, polynomial multiplication simultaneously increases and scrambles the error positions, and in so doing prevents an attack that is based on a statistical analysis as the one that affected the previous proposal. This is because in general the inverse of c is not guaranteed to have low weight (in fact its weight is in general close to average) and so the vector $y' = c^{-1}y$ effectively behaves as a random vector and thus hides s.

5 Performance and Comparison

To evaluate the performance of our scheme, we start by recalling that the full-fledged signature scheme is obtained via the Fiat-Shamir transform[11]. 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 $\mathcal{F}$ (usually a hash function). The signature is then accepted according to the validity of the response in the identification scheme.

Table 3: The Fiat-Shamir Signature Scheme.

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

[1] Note that $c^{(1)}$ and $c^{(2)}$ can be chosen *ad hoc* such that their sum is surely invertible.

Thus the signature is given by a commitment string and the response string z. In our scheme, this corresponds to a hash value (of length $\ell_{\mathcal{F}}$, say 128 or 256 bits), and a vector of length $2p$. The public data, on the other hand, consists of the vector h (of length p) and the syndrome S (also of length p), for a total of $2p$ bits. Let's look at some parameters for the codes in our scheme. These are normally evaluated with respect to general decoding algorithms such as Information-Set Decoding [23,24,5,16,17]: this is indicated in the column "Security".

Table 4: Parameters.

p	w_1	w_2	δ	Security (log)	Signature Size (bits)	Public Data (bits)
4801	90	100	10	80	$9602 + \ell_{\mathcal{F}}$	9602
9857	150	200	12	128	$19714 + \ell_{\mathcal{F}}$	19714
3072	85	85	7	80	$6144 + \ell_{\mathcal{F}}$	6144
6272	125	125	10	128	$12544 + \ell_{\mathcal{F}}$	12544

The first two rows recall "well-known" parameters suggested in the literature for QC-MDPC codes; because our codes do not need to be decodable, we are able to slightly increase the number of errors. The last two rows, instead, are parameters chosen in a similar way to those above, while trying to optimize performance.

5.1 Comparison with other Code-Based Schemes

Due to the highly diverse nature of code-based schemes, it would prove difficult to reunite them all in just one table. Instead, we are going to (briefly) discuss individually the three main proposals, related variants and improvements.

CFS This 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 [2] reduce this size drastically and are therefore able to complete with our proposal, 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). The advantage of our scheme is then evident, as the signing process only requires the computation of two hash values and two polynomial multiplications.

KKS The KKS approach 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 their security. In fact, it has been shown in [6] 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. [1]) is secure, as shown by Otmani and Tillich [20], 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.

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 2 or 3 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 [25] (see Appendix C), it is possible to choose any 2 out of 3 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 more recent schemes, but even so, communication costs prove to be very high, leading to a very large signature size. Below, we report a comparison of parameters for different variants of the scheme, where the column Véron refers to [27], CVE to [7] and AGS to [18]. Note that all of these parameters refer to a cheating probability of 2^{-16}, a weak authentication level.

Table 5: Comparison of the most popular zero-knowledge identification schemes. All the sizes are expressed in bits.

	Stern 3	Stern 5	Véron	CVE	AGS
Rounds	28	16	28	16	18
Public Data[2]	122500	122500	122500	32768	350
Private Key	700	4900	1050	1024	700
Public Key	350	2450	700	512	700
Total Communication Cost	42019	62272	35486	31888	20080

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 93Kb). Moreover, for signatures, 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 [18], to multiply all the above data by 5. In comparison, our scheme achieves the same security level with just 6Kb.

6 Conclusions

In this paper, we have presented a new construction for an identification 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. To the best of our knowledge, it is the first time that this ring metric is employed in code-based signature schemes. This allows to create a protocol that resembles the lattice-based proposal by Lyubashevsky, and succeeds where previous code-based proposals had failed (e.g. [22]). Thanks to the simplicity of its design, our protocol ends up being very competitive: it features a compact public key, fast signing and verification algorithms, and the signature size is much shorter than other identification-based schemes. Thus, our scheme is a strong candidate to solve the long time open problem of designing an efficient code-based signature scheme.

References

1. P. S. L. M. Barreto, R. Misoczki, and M. A. Simplício Jr. One-time signature scheme from syndrome decoding over generic error-correcting codes. *Journal of Systems and Software*, 84(2):198–204, 2011.

[2] The public matrix H.

2. Paulo S. L. M. Barreto, Pierre-Louis Cayrel, Rafael Misoczki, and Robert Niebuhr. *Quasi-Dyadic CFS Signatures*, pages 336–349. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
3. T. P. Berger, P.-L. Cayrel, P. Gaborit, and A. Otmani. Reducing Key Length of the McEliece Cryptosystem. In B. Preneel, editor, *AFRICACRYPT*, volume 5580 of *Lecture Notes in Computer Science*, pages 77–97. Springer, 2009.
4. E. Berlekamp, R. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384 – 386, may 1978.
5. D. J. Bernstein, T. Lange, and C. Peters. Attacking and Defending the McEliece Cryptosystem. In J. Buchmann and J. Ding, editors, *PQCrypto*, volume 5299 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2008.
6. P.-L. Cayrel, A. Otmani, and D. Vergnaud. On Kabatianskii-Krouk-Smeets Signatures. In C. Carlet and B. Sunar, editors, *WAIFI*, volume 4547 of *Lecture Notes in Computer Science*, pages 237–251. Springer, 2007.
7. P.-L. Cayrel, P. Véron, and S. M. El Yousfi Alaoui. A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem. In A. Biryukov, G. Gong, and D. R. Stinson, editors, *Selected Areas in Cryptography*, volume 6544 of *Lecture Notes in Computer Science*, pages 171–186. Springer, 2010.
8. N. Courtois, M. Finiasz, and N. Sendrier. How to Achieve a McEliece-Based Digital Signature Scheme. In C. Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 157–174. Springer, 2001.
9. J.-C. Faugère, A. Otmani, L. Perret, and J.-P. Tillich. Algebraic Cryptanalysis of McEliece Variants with Compact Keys. In H. Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 279–298. Springer, 2010.
10. U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1:77–94, 1988.
11. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In A. M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.
12. G. Kabatianskii, E. Krouk, and B. J. M. Smeets. A Digital Signature Scheme Based on Random Error-Correcting Codes. In M. Darnell, editor, *IMA Int. Conf.*, volume 1355 of *Lecture Notes in Computer Science*, pages 161–167. Springer, 1997.
13. G. Locke and P. Gallagher. FIPS PUB 186-3: Digital Signature Standard (DSS). *National Institute of Standards and Technology*, 2009.
14. V. Lyubashevsky. Lattice-Based Identification Schemes Secure Under Active Attacks. In R. Cramer, editor, *Public Key Cryptography*, volume 4939 of *Lecture Notes in Computer Science*, pages 162–179. Springer, 2008.
15. V. Lyubashevsky. Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures. In M. Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616. Springer, 2009.
16. A. May, A. Meurer, and E. Thomae. Decoding Random Linear Codes in $\mathcal{O}(2^{0.054n})$. In D. H. Lee and X. Wang, editors, *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 107–124. Springer, 2011.
17. A. May and I. Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In *Advances in Cryptology - EUROCRYPT* 2015, Lecture Notes in Computer Science. Springer, 2015.
18. C. Aguilar Melchor, P. Gaborit, and J. Schrek. A new zero-knowledge code based identification scheme with reduced communication. *CoRR*, abs/1111.1644, 2011.

19. T. Okamoto. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. In E. F. Brickell, editor, *CRYPTO*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53. Springer, 1992.
20. A. Otmani and J.-P. Tillich. An Efficient Attack on All Concrete KKS Proposals. In B.-Y. Yang, editor, *PQCrypto*, volume 7071 of *Lecture Notes in Computer Science*, pages 98–116. Springer, 2011.
21. R. Overbeck and N. Sendrier. Code-based cryptography. In D. J. Bernstein, J. Buchmann, and E. Dahmen, editors, *Post-Quantum Cryptography*, pages 95–145. Springer Berlin Heidelberg, 2009.
22. E. Persichetti. *Improving the Efficiency of Code-Based Cryptography.* PhD thesis, University of Auckland, 2012.
23. E. Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, pages 5–9, 1962.
24. J. Stern. A method for finding codewords of small weight. In G. D. Cohen and J. Wolfmann, editors, *Coding Theory and Applications*, volume 388 of *Lecture Notes in Computer Science*, pages 106–113. Springer, 1988.
25. J. Stern. A New Identification Scheme Based on Syndrome Decoding. In D. R. Stinson, editor, *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 13–21. Springer, 1993.
26. J. Stern. Designing Identification Schemes with Keys of Short Size. In Y. Desmedt, editor, *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 164–173. Springer, 1994.
27. P. Véron. Improved identification schemes based on error-correcting codes. *Appl. Algebra Eng. Commun. Comput.*, 8(1):57–69, 1996.

A Feige-Fiat-Shamir

Table 6: Feige-Fiat-Shamir Identification Scheme.

Public Data	An RSA modulus $N = pq$.
Private Key	$s_1, \dots, s_k$ with $(s_i, N) = 1$.
Public Key	$v_1, \dots, v_k$ with $v_i = s_i^2 \pmod N$.

PROVER		VERIFIER
Choose $y \xleftarrow{\$} \mathbb{Z}$ and $r \xleftarrow{\$} \{-1, 1\}$, then set $Y = ry^2 \pmod N$.	$\xrightarrow{Y}$	
	$\xleftarrow{c}$	$c = (c_1, \dots, c_k)$ with $c_i \xleftarrow{\$} \{0, 1\}$.
Compute $z = ys_1^{c_1} s_2^{c_2} \dots s_k^{c_k} \pmod N$.	$\xrightarrow{z}$	Accept if $z^2 = \pm Y v_1^{c_1} v_2^{c_2} \dots v_k^{c_k} \pmod N$.

B Other Identification Schemes built using the Framework

Table 7: Factoring-based Identification Scheme.

Public Data	An RSA modulus $N = pq$ and a group element g.
Private Key	$s \in D_s$.
Public Key	$S = g^s \pmod{N}$.

PROVER		VERIFIER
Choose $y \xleftarrow{\$} D_y$ and set $Y = g^y \pmod{N}$.	$\xrightarrow{Y}$	
	$\xleftarrow{c}$	$c \xleftarrow{\$} D_c$.
Compute $z = y + cs$.	$\xrightarrow{z}$	Accept if $g^z = YS^c \pmod{N}$.

Table 8: Lattice-based Identification Scheme.

Public Data	A hash function $f \xleftarrow{\$} \mathbb{H}(R, D, m)$.
Private Key	$s \in D_s^m$.
Public Key	$S = f(s)$.

PROVER		VERIFIER
Choose $y \xleftarrow{\$} D_y^m$ and set $Y = f(y)$.	$\xrightarrow{Y}$	
	$\xleftarrow{c}$	$c \xleftarrow{\$} D_c$.
Compute $z = y + cs$. If $z \notin G^m$ set $z = \perp$.	$\xrightarrow{z}$	Accept if $z \in G^m$ and $f(z) = Y + cS$.

The hash functions in this case are sampled from the family $\mathbb{H}(R, D, m) = \{f_a : a \in R^m\}$ where R is the ring $\mathbb{Z}_p[x]/(x^n + 1)$, $D \subseteq R$ and, for every $z \in D^m$, we define $f_a(z) = a \cdot z$. Alternatively, we can see $f_a(z)$ as Az^T, where A is the circulant matrix over $\mathbb{F}_p$ whose first row is a. The subset G is exactly the "safe" subset described above.

C Stern's Identification Scheme

Table 9: Stern Identification Scheme.

Public Data	The parameters $n, k, w \in \mathbb{N}$, an $(n-k) \times n$ parity-check matrix H over $\mathbb{F}_2$ and a hash function $\mathcal{H}$.
Private Key	$s \in \mathbb{W}_{2,n,w}$.
Public Key	$S = Hs^T$.

PROVER		VERIFIER
Choose $y \xleftarrow{\$} \mathbb{F}_2^n$ and a permutation $\pi \xleftarrow{\$} \mathrm{Sym}(n)$, then set $c_1 = \mathcal{H}(\pi, Hy^T)$, $c_2 = \mathcal{H}(\pi(y)), c_3 = \mathcal{H}(\pi(y+s))$.	$\xrightarrow{c_1, c_2, c_3}$	
	$\xleftarrow{b}$	$b \xleftarrow{\$} \{0,1,2\}$.
If $b = 0$ set $z = (y, \pi)$.		Accept if c_1 and c_2 are correct.
If $b = 1$ set $z = (y+s, \pi)$.	$\xrightarrow{z}$	Accept if c_1 and c_3 are correct.
If $b = 2$ set $z = (\pi(y), \pi(\boldsymbol{s}))$.		Accept if c_2 and c_3 are correct and $\mathtt{wt}(\pi(s)) = w$.