

# On Kabatianskii-Krouk-Smeets Signatures

Pierre-Louis Cayrel<sup>1</sup>, Ayoub Otmani<sup>2</sup>, and Damien Vergnaud<sup>3</sup>

<sup>1</sup> DMI/XLIM - Université de Limoges, 123 avenue Albert Thomas, 87060 Limoges, France.

`pierre-louis.cayrel@xlim.fr`

<sup>2</sup> GREYC - Ensicaen, Boulevard Maréchal Juin, 14050 Caen Cedex, France.

`Ayoub.Otmani@info.unicaen.fr`

<sup>3</sup> b-it COSEC - Bonn/Aachen International Center for Information Technology - Computer Security Group, Dahlmannstr. 2, D-53113 Bonn, Germany.

`vergnaud@bit.uni-bonn.de`

**Abstract.** Kabastianskii, Krouk and Smeets proposed in 1997 a digital signature scheme based on random error-correcting codes. In this paper we investigate the security and the efficiency of their proposal. We show that a passive attacker who may intercept just a few signatures can recover the private key. We give precisely the number of signatures required to achieve this goal. This enables us to prove that all the schemes given in the original paper can be broken with at most 20 signatures. We improve the efficiency of these schemes by firstly providing parameters that enable to sign about 40 messages, and secondly, by describing a way to extend these *few-times* signatures into classical *multi-time* signatures. We finally study their key sizes and a mean to reduce them by means of more compact matrices.

**Keywords.** Code-based cryptography, digital signature, random error-correcting codes, Niederreiter cryptosystem.

## 1 Introduction

Kabastianskii, Krouk and Smeets proposed in 1997 a digital signature scheme based on random error-correcting codes. In this paper we investigate the security and the efficiency of their proposal. We show that a passive attacker who may intercept just a few signatures can recover the private key. We give precisely the number of signatures required to achieve this goal. This enables us to prove that all the schemes given in the original paper can be broken with at most 20 signatures. We improve the efficiency of these schemes by firstly providing parameters that enable to sign about 40 messages, and secondly, by describing a way to extend these *few-times* signatures into classical *multi-time* signatures. We finally study their key sizes and a mean to reduce them by means of more compact matrices.

**Related work.** In 1978, McEliece [13] proposed the first public key cryptosystem based on coding theory. His idea is to first select a particular code for which

an efficient decoding algorithm is known, and then disguise it as a general-looking linear code. A description of the original code can serve as the private key, while a description of the transformed code serves as the public key. Several proposals were made to modify McEliece's original scheme, but unfortunately most of them turn out to be insecure or inefficient. However, the original primitive, which uses Goppa codes, has remained unbroken (for appropriate system parameters).

In 1986, Niederreiter [15] proposed another (knapsack-based) scheme which relies on a linear code. The McEliece scheme uses a generator matrix while the Niederreiter scheme uses a parity-check matrix, but they were proved [11] to be equivalent in terms of security for the same parameters<sup>4</sup>.

Compared with other public-key cryptosystems which involve modular exponentiation, these schemes have the advantage<sup>5</sup> of high-speed encryption and decryption. However, they suffer from the fact that the public key is very large. In 2005, Gaborit [6] proposed a method to severely decrease its size (making it almost linear – see below).

Digital signature schemes are the most important cryptographic primitive for providing authentication in an electronic world. They allow a signer with a secret key to sign messages such that anyone with access to the corresponding public key is able to verify authenticity of the message. Parallel to the efforts to build an efficient public key encryption scheme from error correcting codes, several attempts were proposed to design signature schemes based on error-correcting codes. Unfortunately, most of the proposed protocols have been proved insecure (see the survey [5] and the references therein for details).

It is well known that any trapdoor permutation permits to design digital signatures by using the unique capacity of the owner of the public key to invert this permutation. The so-called *Full Domain Hash* (FDH) approach can only be used to sign messages whose hash values lies in the range set of the trapdoor permutation. Therefore, a signature scheme based on trapdoor codes must achieve complete decoding. In 2001, Courtois, Finiasz and Sendrier [4] have presented a practical signature scheme derived from a technique allowing complete decoding of Goppa codes (for some parameter choices).

At Crypto'93, Stern [19] proposed an identification scheme based on the syndrome decoding problem. In this scheme, all users share a parity-check matrix for a binary linear code and each user secretly chooses a vector  $\mathbf{v}$  of *small* Hamming weight (slightly below the expected minimum distance of the code). The public key of identification is the corresponding syndrome. By an interactive zero-knowledge protocol, any user can identify himself to another by proving he knows  $\mathbf{v}$  without revealing it. A dual version of the Stern identification scheme that uses a generator matrix of the code was proposed by Véron [20]. Both protocols can give rise to digital signature schemes (though inefficient), by applying the well-known Fiat-Shamir heuristic.

<sup>4</sup> Niederreiter's original scheme relies on generalized Reed-Solomon codes and the two primitives are equivalent if we substitute these codes by Goppa codes.

<sup>5</sup> For the same parameters, the Niederreiter cryptosystem reveals some advantages, for example, the size of the public key and the number of operations to encrypt.

Finally, Kabatianskii, Krouk and Smeets [7] proposed, 10 years ago, a digital signature scheme based on random linear codes. They exploited the fact that for every linear code the set of its correctable syndrome contains a linear subspace of relatively large dimension. Kabatianskii *et al.* concluded their paper by asking for an analysis of the efficiency and the security of their scheme. The investigation of this issue is the main purpose of the present paper.

**Organisation of the paper.** The rest of this paper is organized as follows. Section 2 begins with background material on coding theory. In section 3 and 4, we review the paradigm of *signing without decoding*, together with the schemes proposed by Kabatianskii *et al.* (KKS-1 to KKS-4). In section 5, we present an analysis of their security under a known message attack. In particular, we show that a passive attacker who may only intercept signatures can recover the private key. For the concrete parameters, just about 20 signatures are sufficient to reveal a large part of the key. Section 6 discusses a way to extend these *few-times* signatures into classical *multi-time* signatures. In section 7, we study the key sizes of the KKS signature schemes and propose a way to reduce them. Section 8 concludes the paper with efficiency considerations.

## 2 Notations and Definitions

Let  $n$  be a non-negative integer and  $q$  be a prime power that usually is 2. The support  $\text{supp}(x)$  of a vector  $x \in GF(q)^n$  is the set of coordinates  $i$  such that  $x_i \neq 0$ . The (*Hamming*) weight  $\text{wt}(x)$  of  $x \in GF(q)^n$  is the cardinality of  $\text{supp}(x)$ . A  $\mathcal{C}[n, n-r, d]$  code over  $GF(q)$  is a linear subspace of  $GF(q)^n$  of dimension  $n-r$  and minimum distance  $d$ . The elements of  $\mathcal{C}$  are *codewords*. A linear code can be defined either by a parity check matrix or a generator matrix. A *parity check matrix*  $H$  for  $\mathcal{C}$  is an  $r \times n$  matrix such that the vectors  $w \in GF(q)^n$  which are solutions to the equation  $Hw^T = 0$  are exactly the codewords of  $\mathcal{C}$ . When the first  $r$  columns of  $H$  form the  $r \times r$  identity matrix, we denote  $H$  by  $(I_r | M)$  where the columns of  $M$  are the last  $n-r$  columns of  $H$ . A generator matrix  $G$  is an  $(n-r) \times n$  matrix formed by a basis of  $\mathcal{C}$ .  $G$  is *systematic* if its first  $n-r$  columns form  $I_{n-r}$ . For a non-negative integer  $t$ , we denote by  $\mathcal{M}_{n,t}$  the set of vectors of  $GF(q)^n$  of weight  $t$  and by  $\mathcal{M}_{n,\leq t}$  the set  $\cup_{i=0}^t \mathcal{M}_{n,i}$ .

A *syndrome decoding algorithm*  $\text{dec}()$  for  $\mathcal{C}$  (defined by a  $r \times n$  parity check matrix  $H$ ) is a process that is able to find for a given vector  $s \in GF(q)^r$  a vector  $e = \text{dec}(s) \in GF(q)^n$  such that:

$$H \cdot e^T = s. \tag{1}$$

The vector  $e$  is seen as an *error* and the element  $s \in GF(q)^r$  is called its *syndrome*. Note that a decoding algorithm does not necessarily succeed in finding an error for *any* syndrome. The algorithm  $\text{dec}()$  achieves a *complete decoding* if it can resolve Equation (1) for *any*  $s \in GF(q)^r$ , and a *t-bounded syndrome decoding algorithm* for  $\mathcal{C}$  is a decoding algorithm that is able to recover any error vector of  $\mathcal{M}_{n,\leq t}$ . More precisely, it is an application  $\text{dec} : H\mathcal{M}_{n,\leq t} \rightarrow \mathcal{M}_{n,\leq t}$  such that

$\text{dec}(0) = 0$  and for any  $s \in H\mathcal{M}_{n,\leq t}$  where  $H\mathcal{M}_{r,\leq t}$  is the set  $\{Hz^T : z \in \mathcal{M}_{n,\leq t}\}$ ,  $\text{dec}(s)$  is solution to Equation (1). The set of *correctable syndromes* for  $\text{dec}$  is therefore  $H\mathcal{M}_{n,\leq t}$ . Note that  $\text{dec}()$  is well-defined when  $2t + 1 < d$  because Equation (1) admits a unique solution. Finally, it has been proved in [1] that the problem of *syndrome decoding* i.e. solving Equation (1) is NP-Hard.

As mentioned in the introduction Niederreiter outlined a public-key cryptosystem based upon the difficulty of the syndrome decoding problem [15] for an arbitrary linear code. It is a modified version of the McEliece cryptosystem [13]. Each user chooses a code  $\mathcal{C}[n, n - r, d]$  over  $GF(q)$  for which a polynomial (in  $n$ ) decoding algorithm is known. The plain text space is the set  $\mathcal{M}_{n,t}$  with  $2t + 1 < d$ . The private key is a parity check matrix  $H$  of  $\mathcal{C}$ , a  $t$ -bounded syndrome decoding algorithm  $\text{dec}()$  for  $\mathcal{C}$ , an  $r \times r$  invertible matrix  $S$  and a  $n \times n$  permutation matrix  $P$ . The public key is the matrix  $H' = SHP$ . The encryption process consists of computing  $c = H'm^T$  for  $m \in \mathcal{M}_{n,t}$ . To decrypt a cipher text  $c' \in GF(q)^r$ , the owner of the private key computes  $\text{dec}(S^{-1} \cdot c') \cdot P$ .

The security of code-based cryptosystems relies upon two kinds of attacks. One type of attacks which are called *structural attacks* aims at totally breaking the cryptosystem by recovering the secret matrices. The other class of attacks try to conceive decoding algorithms for arbitrary linear codes in order to decrypt a given cipher text. Such an attack is called a *decoding attack*. The most efficient algorithms used to decode arbitrary linear codes are based on the information set decoding. A first analysis was done by McEliece in [13], then by Lee and in Brickell in [9] and also by Stern in [18] and Leon in [10] and lastly by Canteaut and Chabaud in [3] which is the best algorithm known up to now with roughly  $O\left(\frac{n^3 \binom{n}{t}}{\binom{r}{t}}\right) = O\left(n^3 2^{nh_2(\frac{t}{n}) - rh_2(\frac{t}{r}) + o(n)}\right)$  operations where  $h_2(x) = -x \log_2(x) - (1-x) \log_2(1-x)$ . Nowadays it is commonly accepted that a system is secure if the best attack known requires more than  $2^{80}$  operations.

### 3 How to Sign without Decoding ?

Let  $k$  be an integer and assume that  $GF(q)^k$  is the set of messages to be signed. In order to sign a message  $m$  by means of a Niederreiter cryptosystem, one has to define an  $k \times n$  parity check matrix  $H$  representing a code  $\mathcal{C}[n, n - k, d]$ . However  $m$  has to be a correctable syndrome or in other words, there must exist  $z \in \mathcal{M}_{n,t}$  such that  $H z^T = m$ . This is not possible for *any* message because the decoding algorithm can only decode the set of correctable syndromes which is different from  $GF(q)^r$ . Thus one needs to find an application  $\chi : GF(q)^k \rightarrow H\mathcal{M}_{n,t}$ , and then decode  $\chi(m)$  to produce  $z \in \mathcal{M}_{n,t}$  that satisfies  $\chi(m) = Hz^T$ .

Kabatianskii *et al.* [7] presented a technique to produce code-based signatures using *any arbitrary linear code*. This means that it is not necessary to design a decoding algorithm to sign messages. The idea is to directly define a *secret* application  $f : GF(q)^k \rightarrow \mathcal{M}_{n,t}$  in order to automatically generate a signature  $f(m)$  for  $m \in GF(q)^k$ . The signer then sends  $(m, f(m))$ . He also publishes an application  $\chi : GF(q)^k \rightarrow H\mathcal{M}_{n,t}$  to be used in the verification step which is

defined for any  $m \in GF(q)^k$  by  $\chi(m) \stackrel{\text{def}}{=} Hf(m)^T$ . A receiver checks the validity of a signed message  $(m, z)$  by verifying that:

$$\text{wt}(z) = t \quad \text{and} \quad \chi(m) = Hz^T. \quad (2)$$

The most important part of a “signing-without-decoding” scheme is to design the application  $f$ . From a practical point of view, the description of  $f$  (and  $\chi$ ) has to be better than an enumeration of its images for which it would need  $q^k \log_2 \binom{n}{t} (q-1)^t$  bits to store  $f$  (and also  $\chi$ ). Thus, a random application  $f$  would be a good choice in terms of security but a bad one for a concrete use. From a security point of view, it is necessary that the public matrix  $H$  and the public application  $\chi$  *do not provide any information* about the secret application  $f$ . If this property is guaranteed then the security of the scheme is equivalent to that of the Niederreiter cryptosystem upon which it is built. Indeed to recover  $f$  from  $H$  (and  $\chi$ ), an opponent has to solve  $\chi(m) = Hz^T$  for a given  $m \in GF(q)^k$  which is actually an instance of the syndrome decoding problem.

Moreover, it should be noted that the only property needed for  $\mathcal{C}$  is that it should be difficult to solve Equation (2). In other words,  $t$  should be large enough or at a pinch  $\mathcal{C}$  can be a *random* linear code provided its minimum distance is large enough. The following proposition given in [7] estimates the minimum distance of a (random) linear code generated by a randomly drawn parity check matrix.

**Proposition 1.** *The probability  $Pr\{d(\mathcal{C}) \geq d\}$  that a random  $r \times n$  parity check matrix  $[I_r|M]$  over  $GF(q)$  defines a code  $\mathcal{C}$  with a minimum distance  $d(\mathcal{C})$  greater or equal to  $d$  satisfies the following inequality:*

$$Pr\{d(\mathcal{C}) \geq d\} \geq 1 - q^{-r+nh_q(\frac{d-1}{n})},$$

where  $h_q(x) = x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x)$ .

## 4 Kabatianskii-Krouk-Smeets Signatures

Kabatianskii *et al.* [7] proposed a signature scheme based on arbitrary linear error-correcting codes. Actually, they proposed to use a *linear* application  $f$ . Three versions are given which are presented in the sequel but all have one point in common: for any  $m \in GF(q)^k$ , the signature  $f(m)$  is a codeword of a linear code  $\mathcal{U}$ . Each version of KKS proposes different linear codes in order to improve the scheme. We now give a full description of their scheme.

Firstly, we suppose that  $\mathcal{C}$  is defined by a random parity check matrix  $H$ . We also assume that we have a very good estimate  $d$  of its minimum distance through Proposition 1 for instance. Next, we consider a linear code  $\mathcal{U}$  of length  $n' \leq n$  and dimension  $k$  defined by a generator matrix  $G = [g_{i,j}]$ . We suppose that there exist two integers  $t_1$  and  $t_2$  such that  $t_1 \leq \text{wt}(u) \leq t_2$  for any non-zero codeword  $u \in \mathcal{U}$ .

Let  $J$  be a subset of  $\{1, \dots, n\}$  of cardinality  $n'$ ,  $H(J)$  be the sub matrix of  $H$  consisting of the columns  $h_i$  where  $i \in J$  and define an  $r \times n'$  matrix

$F \stackrel{\text{def}}{=} H(J)G^T$ . The application  $f : GF(q)^k \rightarrow \mathcal{M}_{n,t}$  is then defined by  $f(m) = mG^*$  for any  $m \in GF(q)^k$  where  $G^* = [g_{i,j}^*]$  is the  $k \times n$  matrix with  $g_{i,j}^* = g_{i,j}$  if  $j \in J$  and  $g_{i,j}^* = 0$  otherwise. The public application  $\chi$  is then  $\chi(m) = Fm^T$  because  $HG^{*T} = H(J)G^T$ . The main difference with Niederreiter signatures resides in the verification step where the receiver checks that:

$$t_1 \leq \text{wt}(z) \leq t_2 \quad \text{and} \quad F \cdot m^T = H \cdot z^T. \quad (3)$$

- **Setup.** The signer chooses a random matrix  $H = [I_r | D]$  that represents the parity check matrix of a code  $\mathcal{C}[n, n-r, \geq d]$ . He also chooses a generator matrix  $G$  that defines a code  $\mathcal{U}[n', k, t_1]$  such that  $\text{wt}(u) \leq t_2$  for any  $u \in \mathcal{U}$ . He chooses a random set  $J \subset \{1, \dots, n\}$  and he forms  $F = H(J)G^T$ .
- **Parameters.**
  - **Private key.** The set  $J \subset \{1, \dots, n\}$  and the  $k \times n'$  matrix  $G$
  - **Public key.** The  $r \times k$  matrix  $F$  and the  $r \times n$  matrix  $H$
- **Signature.** Given  $m \in GF(q)^k$ , the signer sends  $(m, m \cdot G^*)$
- **Verification.** Given  $(m, z)$ , the receiver verifies that:

$$t_1 \leq \text{wt}(z) \leq t_2 \quad \text{and} \quad F \cdot m^T = H \cdot z^T.$$

**Fig. 1.** KKS signature scheme.

Note that it is not so important to have  $d > 2t_2$  because it would mean otherwise that a message  $m$  may have several signatures  $z$  which are all solutions to Equation (3). Recall also that the crucial fact about  $\mathcal{C}$  is that Equation (3) should be difficult to solve when the number of errors (*i.e.* the weight of  $z$ ) belongs to the interval  $[t_1, t_2]$ . Figure 1 sums up the different steps of a KKS signature scheme.

**Definition 1 (KKS-1).** Let  $\mathcal{U}[n', k, t]$  be an equidistant code ( $t_1 = t_2 = t$ ) over  $GF(q)$  such that  $n' \leq n$  defined by a generator matrix  $G = [g_{i,j}]$ . It is known [12] that for such a code,  $n' = \frac{q^k - 1}{q - 1}$  and  $t = q^{k-1}$ .

Unfortunately, KKS-1 is not practicable because it requires a code length too large. For instance in the binary case ( $q = 2$ ) and in the FDH paradigm,  $k$  must be at least 160. It implies that  $n \geq n' = 2^{160} - 1$ . It is necessary to replace the equidistant code by another one for which  $t_1 \neq t_2$ . Two solutions are proposed in [7]: either one chooses the dual code of a binary BCH code or a random linear code thanks to Proposition 2 and Proposition 3.

**Proposition 2 (Carlitz-Uchiyama Bound).** Let  $\mathcal{U}$  be the dual of a binary BCH code of length  $n' = 2^m - 1$  and designed distance  $\delta = 2s + 1$ . Then for any  $u \in \mathcal{U}$ :

$$\left| \text{wt}(u) - \frac{n' + 1}{2} \right| \leq (s - 1)\sqrt{n' + 1}.$$

**Definition 2 (KKS-2).** *The signer chooses randomly: a binary  $r \times (n - r)$  matrix  $D$ , a non singular  $k \times k$  matrix  $A$ , an  $n'$ -subset  $J \subset \{1, \dots, n\}$ . He forms a binary  $r \times n$  parity check matrix  $H = [I_r | D]$ . He chooses a generator matrix  $G$  of a dual binary BCH code with length  $n' = 2^m - 1$  and designed distance  $2s + 1$ . He forms  $F = H(J)(AG)^T$  (he masks matrix  $G$ ). The public key consists of matrices  $H$  and  $F$ , and the secret key is the set  $J$  and the matrix  $A$ .*

The following numeric values are given in [7]:  $m = 10$ ,  $s = 6$ ,  $k = ms = 60$ ,  $n' = 2^{10} - 1 = 1023$ ,  $t_1 = 352$ ,  $t_2 = 672$ ,  $r = 2808$  and  $n = 3000$ . The minimum distance of  $\mathcal{C}$  is at least 1024 with probability  $\geq 1 - 10^{-9}$ .

As for the number of bits to store, we see that the private key consists of  $nh_2(\frac{n'}{n})$  bits for describing  $J$  and  $k^2$  bits for the matrix  $A$ . For the public key, we need to store  $r(n - r)$  bits for the matrix  $H$  and  $rk$  bits for the matrix  $F$ . KKS-2 can be even more improved by taking a random linear code for  $\mathcal{U}$ . Thanks to Proposition 3, it is possible to know the probability that a random linear code have its nonzero codeword weights inside a given interval.

**Proposition 3.** *Let  $\mathcal{U}$  be a code defined by a random  $k \times n'$  systematic generator matrix. Let  $\delta$  be a real such that  $0 < \delta < 1$ . Then the probability that a random binary linear that its nonzero codewords have their weight inside  $[\omega_1; \omega_2]$  is at least:*

$$Pr\{\omega_1 \leq wt(\mathcal{U}) \leq \omega_2\} \geq 1 - 2^{-2(n'-k) + n'h_2(\frac{\omega_1-1}{n'}) + n'h_2(\frac{n'-\omega_2+1}{n'})}.$$

**Definition 3 (KKS-3).** *The signer follows the same steps as KKS-2 but chooses a random  $k \times n'$  systematic matrix  $G = [I_k | B]$ . The public key consists of matrices  $H$  and  $F$ , and the secret key is the set  $J$  and the matrix  $B$ .*

Now the size of the private key of KKS-3 consists again of  $nh_2(\frac{n'}{n})$  bits for  $J$  and  $k(n' - k)$  bits for the matrix  $B$ . The size of the public key is not changed. We give the following numeric values:  $k = 160$ ,  $n' = 900$ . The code  $\mathcal{U}$  generated by  $G$  has all its weights between  $t_1 = 90$  and  $t_2 = 110$  with probability  $\geq 1 - 2^{-749}$ . The signer selects a random  $1100 \times 2000$  parity check matrix  $H$  for  $\mathcal{C}$ . Then  $d(\mathcal{C}) > 220$  with probability  $\geq 1 - 2^{-100}$ . Table 1 gives the parameters given by the authors for KKS-3 called here version #1 updated with our proposition (version #2) that encounters the current security level. Unlike what was done in [7] where the security of the system is evaluated through  $t = \frac{t_1 + t_2}{2}$ , we give a value for  $t_1$  such that the decoding attack [3] can not cope with it.

Finally, the authors proposed a modification that helps someone to construct a KKS scheme from codes that contain codewords of low weight. The idea is to take for the code  $\mathcal{U}$  the direct product of  $P$  codes  $\mathcal{U}_i[n^*, k^*, t_1^*]$  over  $GF(q)$  whose codewords have weight  $\leq t_2^*$ . Of course,  $\mathcal{U}$  has also codewords of low weight. So, one has to find a way to eliminate those codewords. Assume that each code  $\mathcal{U}_i$  is defined by a generator matrix  $G_i$ . The finite field  $GF(q^{k^*})$  is considered as a  $k^*$ -dimensional vector space over  $GF(q)$  defined by a fixed basis. We denote by  $M_\beta$  the matrix representing the linear map  $x \mapsto x\beta$  where  $\beta \in GF(q^{k^*})$ . Let  $Q$

**Table 1.** KKS Parameters.

Scheme	Version	$k$	$n'$	$t_1$	$t_2$	$r$	$n$	$nh_2(\frac{t_1}{n}) - rh_2(\frac{t_1}{r})$
KKS-2		60	1023	352	672	2808	3000	36
KKS-3	#1	60	280	50	230	990	1250	17
	#2	160	1000	90	110	1100	2000	80
KKS-4		48	180	96	96	765	1100	53

be a non-negative integer. For any  $u = (u_1, \dots, u_Q) \in GF(q^{k^*})^Q$  we define for any  $x \in GF(q^{k^*})$

$$u(x) = u_1 + u_2x + \dots + u_Qx^{Q-1}.$$

Let  $\beta_1, \dots, \beta_P$  be non-zero elements of  $GF(q^{k^*})$ ,  $A_1, \dots, A_P$  be non-singular  $k^* \times k^*$  matrices and  $J_1, \dots, J_P$  disjoint subsets of  $\{1, \dots, n\}$  of cardinality  $n^*$  with  $Pn^* < n$ . The application  $f : GF(q^{k^*})^Q \mapsto GF(q)^n$  sends any  $u$  to  $z$  which equals 0 on positions  $\{1, \dots, n\} \setminus \cup_{i=1}^P J_i$  and equals  $u(\beta_i)A_iG_i$  on the positions of  $J_i$ . The signature equation is again  $H z^T = F u^T$  where the public matrix is  $F = (F_1, \dots, F_Q)$  and where  $F_j$  is the  $r \times k^*$  matrix that equals

$$F_j \stackrel{\text{def}}{=} \sum_{i=1}^P H(J_i) \left( M_{\beta_i^{j-1}} A_i G_i \right)^T.$$

The set of messages is now  $GF(q^{k^*})^{Q-1}$  and to sign  $(u_2, \dots, u_Q)$ , the sender chooses  $u_1 \in GF(q^{k^*})$  such that  $u_1 \notin \{\sum_{i=2}^Q u_i \beta_j^{i-1} : j = 1, \dots, P\}$  (this is always possible when  $P \leq q^{k^*}$ ) and so that  $Pt_1^* \leq \text{wt}(f(u)) \leq Pt_2^*$ . We call this scheme KKS-4.

**Definition 4 (KKS-4).** *The signer chooses  $P$  codes  $\mathcal{U}_i[n^*, k^*, t_1]$  over  $GF(q)$  whose codewords have weight  $\leq t_2$ , nonzero elements  $\beta_1, \dots, \beta_P$  in  $GF(q^{k^*})$ , non singular  $k^* \times k^*$  matrices  $A_1, \dots, A_P$  and disjoint subsets  $J_1, \dots, J_P$  of  $\{1, \dots, n\}$ . These quantities form the secret key. He forms matrix  $F$  as described above which constitutes with matrix  $H$  the public key.*

Note that in this modified scheme  $n' = Pn^*$ ,  $k = Pk^*$ ,  $t_1 = Pt_1^*$  and  $t_2 = Pt_2^*$ . The authors gave these values:  $Q = 14$ ,  $P = 12$ ,  $k^* = 4$ ,  $n^* = 15$ . The codes  $\mathcal{U}_1, \dots, \mathcal{U}_P$  are all equal to a binary equidistant code  $\mathcal{W}[15, 4, 8]$ .  $\mathcal{C}$  is a random code of length  $n = 1100$  and dimension 335. The minimum distance  $d(\mathcal{C}) \geq 193$  with probability at least  $1 - 10^{-9}$ . Table 1 recapitulates these values.

## 5 Recovering the Private Key under a Known Message Attack

The security of KKS signatures rests on the quality of  $f$ . We have seen that if  $f$  provides no information then *a priori* KKS scheme is as secure as a Niederreiter

cryptosystem. In reality, with each use of  $f$ , we do obtain a lot of information. Indeed, each signature  $z$  reveals  $|\text{supp}(z)|$  positions of the secret set  $J$ . Therefore an opponent can exploit this fact to recover  $J$ . Note that once the opponent knows  $J$ , he can find secret matrix  $G$  by just solving the linear system  $F = H(J)G^T$  where  $G$  represents the unknown because with high probability  $H(J)$  is a full rank matrix since most of the time  $r > n'$ .

However in the case of KKS-4, the opponent has also to find  $A_1, \dots, A_P$  and the elements  $\beta_1, \dots, \beta_P$  that are roots of the polynomials  $U(X) = \sum_{i=1}^Q u_i X^i$  defined by each message  $(u_1, \dots, u_Q)$ . We do not treat this issue in this paper and we prefer to focus on the first scheme.

We assume that the attacker has  $\ell \geq 1$  signatures  $(m_i, z_i)$  at his disposal. Each signature  $z_i$  can be seen as a result of an independent random choice, and  $\ell$  signatures give  $|\cup_{i=1}^{\ell} \text{supp}(z_i)|$  elements of  $J$ . We define the random variable  $U_\ell \stackrel{\text{def}}{=} |\cup_{i=1}^{\ell} \text{supp}(z_i)|$ . Thus  $\ell$  signatures reveal on average  $\mathbf{E}[U_\ell]$  positions of  $J$  where  $\mathbf{E}[X]$  is the expectation of a random variable  $X$ . For any position  $j \in \{1, \dots, n'\}$ , let  $\chi_j$  be the Bernoulli random variable defined by  $\chi_j = 1$  if  $j \in \cup_{i=1}^{\ell} \text{supp}(z_i)$  and by  $\chi_j = 0$  otherwise. By definition  $U_\ell = \sum_{j=1}^{n'} \chi_j$  and consequently  $\mathbf{E}[U_\ell] = \sum_{j=1}^{n'} \mathbf{E}[\chi_j]$ . Moreover,  $\Pr\{\chi_j = 0\} = \prod_{i=1}^{\ell} \Pr\{j \notin \text{supp}(z_i)\} = \Pr\{j \notin \text{supp}(z_1)\}^\ell$  since the signatures  $z_i$  are considered as independent random variables. Let  $N_w$  be the number of codewords of  $\mathcal{U}$  of weight  $w$  and for any  $j \in \{1, \dots, n'\}$  let  $N_w(j)$  be the number of codewords  $u \in \mathcal{U}$  of weight  $w$  such that  $u_j = 0$ . We have then:

$$\begin{aligned} \Pr\{j \notin \text{supp}(z_i)\} &= \sum_{t=t_1}^{t_2} \Pr\{j \notin \text{supp}(z_i), \text{wt}(z_i) = t\} \\ &= \sum_{t=t_1}^{t_2} \Pr\{j \notin \text{supp}(z_i) | \text{wt}(z_i) = t\} \Pr\{\text{wt}(z_i) = t\}. \end{aligned}$$

But,  $\Pr\{\text{wt}(z_i) = t\} = \frac{N_t}{q^k}$  and  $\Pr\{j \notin \text{supp}(z_i) | \text{wt}(z_i) = t\} = \frac{N_t(j)}{N_t}$ . A way to simplify the computations is to approximate this last equality by putting:

$$\frac{N_t(j)}{N_t} = \Pr\{j \notin \text{supp}(z_i) | \text{wt}(z_i) = t\} \simeq \frac{\binom{n-1}{t}}{\binom{n}{t}} = \left(1 - \frac{t}{n'}\right).$$

This enables us to obtain that:

$$\Pr\{j \notin \text{supp}(z_i)\} = 1 - \frac{q^{-k}}{n'} \sum_{t=t_1}^{t_2} t N_t.$$

This implies that  $\Pr\{\chi_j = 0\} = \left(1 - \frac{q^{-k}}{n'} \sum_{t=t_1}^{t_2} t N_t\right)^\ell$ . Thus if we set:

$$p_\ell \stackrel{\text{def}}{=} 1 - \left(1 - \frac{q^{-k}}{n'} \sum_{t=t_1}^{t_2} t N_t\right)^\ell$$

then we have proved the following proposition.

**Proposition 4.** *The expected number  $E[U_\ell]$  of elements of  $J$  revealed by  $\ell$  signatures is:*

$$E[U_\ell] = n'p_\ell.$$

Note that the explicit formula for  $\Pr\{U_\ell = j\}$  for any integer  $j$  can be found in [2].

It is necessary to know the weight distribution of the code  $\mathcal{U}$  if one wishes to use Proposition 4. This property is in general difficult to calculate for an arbitrary linear code. However, an opponent can easily execute the following attack. Since  $E[U_\ell] = n'p_\ell$  positions of  $J$  are known on average with  $\ell$  signatures, the opponent has to search the  $(n' - n'p_\ell)$  missing elements of  $J$  among the  $(n - n'p_\ell)$  positions left. At each step, he solves  $k$  systems of  $r$  linear equations with  $n'$  unknowns and stops as soon as the system admits a solution. The cost of this attack is therefore  $O(kn'^\omega \binom{n-n'p_\ell}{n'-n'p_\ell})$  where  $n'^\omega$  represents the cost to solve a linear system with  $n'$  unknowns (naively  $\omega = 3$ ). In order to apply Proposition 4 to any linear code, we need to give inequalities for  $p_\ell$ . This can be done by remarking that:

$$1 - \left(1 - \frac{t_1}{n'}\right)^\ell \leq p_\ell \leq 1 - \left(1 - \frac{t_2}{n'}\right)^\ell.$$

Let us define  $a \stackrel{\text{def}}{=} n'(1 - \frac{t_1}{n'})^\ell$  and  $b \stackrel{\text{def}}{=} n - n' + n'(1 - \frac{t_1}{n'})^\ell$ . We have then  $\binom{n-n'p_\ell}{n'-n'p_\ell} \leq \binom{b}{a}$ . We put in Table 2 the number of operations of the attack for different  $\ell$ . These numeric results are obtained by means of Inequality (4) and by putting  $\omega = 3$ :

$$\binom{b}{a} \leq \frac{1}{\sqrt{2\pi a(1 - \frac{a}{b})}} 2^{bh_2(\frac{a}{b})}. \quad (4)$$

We see for instance that we need only  $\ell = 13$  signatures to break KKS-2 with an amount of  $O(2^{78})$  operations, and  $\ell = 20$  signatures to break version #1 of the KKS-3 system with an amount of  $O(2^{77})$  operations.

These numerical results are confirmed by Proposition 5 that gives a very good approximation of the maximum number of signatures allowed without compromising the security of a KKS scheme.

**Proposition 5.** *Assume that  $n$  sufficiently large and let  $n'$  be such that  $2n' \leq n$  and such that the security parameter  $\lambda$  defined by  $\frac{80 - \omega \log_2 n' - \log_2 k}{n - n'}$  satisfies  $0 < \lambda < 1$ . Let  $\gamma$  be the smallest real  $> 0$  such that  $h_2(\gamma) = \lambda$ . Let us define  $\ell_\gamma$  by:*

$$\ell_\gamma \stackrel{\text{def}}{=} \frac{\ln \frac{\gamma}{1-\gamma} + \ln(\frac{n}{n'} - 1)}{\ln(1 - \frac{t_2}{n'})}.$$

*The private key of the KKS system can be recovered with  $\ell$  signatures if  $\ell \geq \ell_\gamma$ .*

**Table 2.** Number of operations to recover  $J$  for different values of  $\ell$  for the schemes proposed in [7].

	$\ell = 15$	$\ell = 14$	$\ell = 13$	$\ell = 12$	$\ell = 11$	$\ell = 10$
KKS-2	$2^{56}$	$2^{65}$	$2^{78}$	$2^{97}$	$2^{122}$	$2^{160}$

	$\ell = 23$	$\ell = 22$	$\ell = 21$	$\ell = 20$	$\ell = 19$	$\ell = 18$
KKS-3 (version #1)	$2^{58}$	$2^{64}$	$2^{70}$	$2^{77}$	$2^{86}$	$2^{96}$

	$\ell = 6$	$\ell = 5$	$\ell = 4$	$\ell = 3$	$\ell = 2$
KKS-4	$2^{46}$	$2^{63}$	$2^{96}$	$2^{155}$	$2^{261}$

*Proof.* Let  $\ell \leq \ell_\gamma$  and let  $\delta_\ell \stackrel{\text{def}}{=} \frac{n' - n' p_\ell}{n - n' p_\ell}$ . Note that  $\delta_\ell \leq \frac{n'}{n} \leq \frac{1}{2}$ . It is well-known that  $\binom{n - n' p_\ell}{n' - n' p_\ell} = 2^{(n - n' p_\ell) h_2(\delta_\ell) + o(n)}$ . One can check that if  $\ell \leq \ell_\gamma$  then  $\delta_\ell \geq \gamma$  and therefore  $h_2(\delta_\ell) \geq h_2(\gamma)$ . Since  $p_\ell \leq 1 - (1 - \frac{t_2}{n'})^\ell$ , we can write that:

$$\begin{aligned} (n - n' p_\ell) h_2(\delta_\ell) &\geq \left( n - n' + n' \left( 1 - \frac{t_2}{n'} \right)^{\ell_\gamma} \right) h_2(\gamma) \\ &\geq \frac{1 + \gamma}{1 - \gamma} (80 - \omega \log_2 n' - \log_2 k). \end{aligned}$$

So we have  $k n' \omega 2^{(n - n' p_\ell) h_2(\delta_\ell)} \geq 2^{80}$  because  $\frac{1 + \gamma}{1 - \gamma} \geq 1$  ( $\gamma \geq 0$ ) and this terminates the proof.

Proposition 5 gives  $\ell_\gamma = 46$  allowed signatures obtained with  $\gamma = 0.00421 \dots$  for our parameters of KKS-3 version #2. Actually, Inequality (4) shows that we can sign at most 40 times.

## 6 Extension to multi-time signatures

**From one-time to multi-time signatures.** Merkle trees were invented in 1979 by Merkle [14]. The original purpose was to make it possible to efficiently handle many Lamport [8] one-time signatures. A Merkle tree is a way to *commit* to  $n$  messages with a single hash value in such a way that revealing any particular message requires revelation of only  $\log n$  hash values.

The underlying idea is to place the  $n$  messages at the leaves of a complete binary tree (assuming  $n$  is a power of 2 for the sake of simplicity) and then to compute the value at each non-leaf node in the tree as the hash of the values of its two children. The value at the root of the tree is the *commitment* to the  $n$  messages. To reveal a value, the user publishes it as well as the values at siblings of each ancestor of it (the so-called *authenticating path*). One can easily verify that the value was correctly revealed by simply computing hashes up the tree and checking that the ultimate hash value matches the root.

Merkle trees have been proposed to extend one-time signatures to multi-time signatures. The idea is to generate  $n$  one-time public keys, and place them in a Merkle tree. The root of the Merkle tree becomes the public key of the signature scheme. For more details, we refer the reader to Merkle's original paper [14].

**From few-time to multi-time signatures.** Following Merkle's idea, it is possible to extend a *few-time* signature scheme into a *multi-time* signature scheme with the same security. If the underlying signature scheme is secure against a  $\ell$ -chosen/known message attacks, the idea is to place the  $n$  messages at the leaves of a complete  $\ell$ -ary tree.

Let  $\Sigma_\ell = (\text{Setup}_\ell, \text{Sign}_\ell, \text{Verify}_\ell)$  be a signature scheme secure against a  $\ell$ -chosen/known message attacks and let  $n$  be an integer (for the ease of explanation, we assume that  $n = \ell^p$  is a power of  $\ell$ ). The scheme  $\Sigma_{\text{multi}} = (\text{Setup}_{\text{multi}}, \text{Sign}_{\text{multi}}, \text{Verify}_{\text{multi}})$  is defined as follows:

- **Setup<sub>multi</sub>**: on input an integer  $\lambda$  (the security parameter), **Setup<sub>multi</sub>** calls  $(\ell^{p-1} - 1)/(\ell - 1)$  times **Setup<sub>ℓ</sub>**( $\lambda$ ) in order to obtain the key pairs:

$$(\text{pk}_{i,j}, \text{sk}_{i,j}) \text{ for } j \in \{0, \dots, p-1\} \text{ and } i \in \{1, \dots, \ell^j\}.$$

The public key is the root  $\text{pk}_{1,0}$  and the private key consists of a concatenation of the key pairs  $(\text{pk}_{i,j}, \text{sk}_{i,j})$ . The user must keep a counter which contains the number of previously created signatures. In the beginning, the counter is set to zero.

- **Sign<sub>multi</sub>**: Let  $i$  be the counter. On input a message  $m$  and the secret key, **Sign<sub>multi</sub>** computes  $\sigma_0 = \text{Sign}_\ell(m, \text{sk}_{r_0, p-1})$  where  $r_0 = \lfloor i/\ell \rfloor$  and then recursively  $\sigma_{t+1} = \text{Sign}_\ell(\text{pk}_{r_t, p-1-t}, \text{sk}_{r_{t+1}, p-2-t}, \sigma_t)$ , where  $r_{t+1} = \lfloor r_t/\ell \rfloor$  for  $t \in \{0, \dots, p-2\}$ . The resulting signature on  $m$  is:

$$\sigma = (\sigma_0, \text{pk}_{r_0, p-1}, \sigma_1, \text{pk}_{r_1, p-2}, \dots, \sigma_{p-2}, \text{pk}_{r_{p-2}, 1}).$$

- **Verify<sub>multi</sub>**: on input a message  $m$ , a signature

$$\sigma = (\sigma_0, \text{pk}_{r_0, p-1}, \sigma_1, \text{pk}_{r_1, p-2}, \dots, \sigma_{p-2}, \text{pk}_{r_{p-2}, 1}),$$

and a public key  $\text{pk}_{0,1}$ , **Verify<sub>multi</sub>** accepts the signature  $\sigma$  if and only if:

$$\text{Verify}_\ell(\text{pk}_{r_t, p-1-t}, \text{sk}_{r_{t+1}, p-2-t}, \sigma_{t+1}) = 1 \text{ for } t \in \{1, \dots, p-2\}$$

and  $\text{Sign}_\ell(m, \text{sk}_{r_0, p-1}, \sigma_0) = 1$ .

The security of  $\Sigma_{\text{multi}}$  against an  $n$ -chosen/known message attack is trivially equivalent to the one of  $\Sigma_\ell$  against a  $\ell$ -chosen/known message attack. In the design of  $\Sigma_{\text{multi}}$ , tradeoffs can be made between size of the signatures and size of the public key.

This construction permits to transform the KKS signature schemes into classical *multi-time* schemes, but the resulting signatures are unfortunately very long, and in order to make the scheme more practical, it is necessary to reduce the size of the public parameters.

## 7 Reduction of Parameters

In this section we study the key sizes of the KKS schemes and the way to reduce them. We restrict ourselves to the binary case. Firstly, we recall the size of the different parameters for each KKS scheme. The private key consists of  $nh_2(\frac{n'}{n}) + k^2$  bits in the case of KKS-2 and  $nh_2(\frac{n'}{n}) + k(n' - k)$  bits in the case of KKS-3. As to the public key, both schemes need to store  $r(n - r) + rk$  bits. Table 3 which gives numeric values obtained shows as such these solutions can not be used practically. However, we can improve the storage of the public key. Indeed,  $H$  can be shared by all the users. Thus each user needs only to provide his own  $F$ .

**Table 3.** Key sizes in bits.

Scheme	Public key			Private key	
	Common ( $H$ )	Personal ( $F$ )	Total public key		
KKS-2	539136	168480	707616	6378	
KKS-3	version #1	257400	59400	316800	14160
	version #2	990000	176000	1166000	120385

Gaborit presented in [6] a method that reduces the key sizes of error-correcting code cryptosystems. The idea relies upon the use of almost quasi-cyclic matrices. Such matrices are completely determined if the first row is known.

**Definition 5 (Almost quasi-cyclic matrix).** *An  $r \times n$  matrix  $M$  with  $r \geq 2$  and  $n \geq 2$  is a almost quasi-cyclic matrix if each row vector is rotated one element to the right relative to the preceding row vector:*

$$M = \begin{pmatrix} c_1 & c_2 & \dots & & \dots & c_n \\ c_n & c_1 & c_2 & & & c_{n-1} \\ c_{n-1} & c_n & c_1 & c_2 & & c_{n-2} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ c_{n-r+2} & \dots & c_{n-1} & c_n & c_1 & c_2 \dots c_{n-r+1} \end{pmatrix}.$$

Our new scheme relies on the use of random almost quasi-cyclic codes rather than pure random linear codes. We modify KKS schemes by replacing each random matrix by a random systematic almost quasi-cyclic matrix. In other words, the parity check matrix  $H = (I_r | D)$  is chosen such that  $D$  is almost quasi-cyclic. The common public key size is now  $(n - r)$  bits and the personal public key still has  $rk$  bits. For KKS-2 the private key does not change but for KKS-3 the random systematic matrix  $G$  can also be a systematic almost quasi-cyclic matrix. In that case the private key has  $nh_2(\frac{n'}{n}) + (n' - k)$ . When applied to our proposed version of KKS-3 (number 2), this methods gives 176000 bits for the personal public key, 900 bits for common public key and only 2726 bits for the private key. The signature length is about  $\log_2 \binom{n}{t_2} = \log_2 \binom{2000}{110} = nh_2(\frac{11}{200}) = 615$  bits.

## 8 Efficiency issues and conclusion

In [16], Perrig proposed a one-time signature scheme called “BiBa” (for *Bins and Balls*) whose main advantages are fast verification and short signatures. In 2002, Reyzin and Reyzin [17] presented a simpler one-time signature scheme, called HORS, that maintains BiBa’s advantages and removes its main disadvantage, namely the costly generation of signatures. As the schemes studied in this paper, the HORS scheme can be used to sign a few number of messages, instead of just once (and the security decreases as this number increases). Therefore it is worth comparing its efficiency with the one of the KKS schemes.

In the table 4, we compare (for the same heuristic security) the performances of KKS-3 version #2 with our parameters and the HORS scheme implemented with the same one-way function and allowing to sign the same number of messages (namely, 40).

Scheme	HORS $(k, t) = (16, 23657)$	HORS $(k, t) = (20, 14766)$	HORS $(k, t) = (32, 8364)$	<b>KKS-3</b>
Public key size	23833000	147836000	854000	<b>176900</b>
Private key size	3785120	2362560	1338240	<b>2726</b>
Signature size	2560	3200	5120	<b>615</b>

**Table 4.** Efficiency comparison of KKS and HORS for 80-bits of heuristic security

KKS compares very favorably in performance with respect to HORS since its key sizes are much smaller and it can be used over a lower bandwidth channel. However, the generation of HORS signatures is faster since it requires only the evaluation of a hash-function. Furthermore, the security of HORS reduces to an ad-hoc (though well-defined) security assumption on the underlying hash function, whereas the scheme KKS has been proposed without any formal security analysis.

In this paper, we have quantified the variation of the security of KKS schemes against a passive attacker who may intercept a few signatures, but an interesting open issue which remains is to study their resistance to forgery in a reductionist approach.

**Acknowledgements.** We thank P. Gaborit for helpful discussions.

## References

1. E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, *On the intractability of certain coding problems*, IEEE Transactions on Information Theory **24** (1978), no. 3, 384–386.
2. M. Barot and J. A. de la Pena, *Estimating the size of a union of random subsets of fixed cardinality*, Elemente der Mathematik **56** (2001), 163–169.

3. A. Canteaut and F. Chabaud, *A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511*, IEEE Transactions on Information Theory **44** (1998), no. 1, 367–378.
4. N. T. Courtois, M. Finiasz, and N. Sendrier, *How to achieve a McEliece-based digital signature scheme*, Lecture Notes in Computer Science **2248** (2001), 157–174.
5. D. Engelbert, R. Overbeck, and A. Schmidt, *A summary of McEliece-type cryptosystems and their security*, Cryptology ePrint Archive, Report 2006/162, 2006, <http://eprint.iacr.org/>.
6. P. Gaborit, *Shorter keys for code based cryptography*, Proceedings of the 2005 International Workshop on Coding and Cryptography (WCC 2005) (Bergen, Norway), March 2005, pp. 81–91.
7. G. Kabatianskii, E. Krouk, and B. J. M. Smeets, *A digital signature scheme based on random error-correcting codes*, IMA Int. Conf., Lecture Notes in Computer Science, vol. 1355, Springer, 1997, pp. 161–167.
8. L. Lamport, *Constructing digital signatures from a one way function*, Tech. Report CSL-98, SRI International, October 1979.
9. P. J. Lee and E. F. Brickell, *An observation on the security of McEliece's public-key cryptosystem*, Advances in Cryptology - EUROCRYPT'88, Lecture Notes in Computer Science, vol. 330/1988, Springer, 1988, pp. 275–280.
10. J. S. Leon, *A probabilistic algorithm for computing minimum weights of large error-correcting codes*, IEEE Transactions on Information Theory **34** (1988), no. 5, 1354–1359.
11. Y. X. Li, R. H. Deng, and X.-M. Wang, *On the equivalence of McEliece's and Niederreiter's public-key cryptosystems*, IEEE Transactions on Information Theory **40** (1994), no. 1, 271–273.
12. F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*, fifth ed., North-Holland, Amsterdam, 1986.
13. R. J. McEliece, *A public-key system based on algebraic coding theory*, pp. 114–116, Jet Propulsion Lab, 1978, DSN Progress Report 44.
14. R. C. Merkle, *A certified digital signature*, Advances in Cryptology - CRYPTO'89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings (G. Brassard, ed.), Lecture Notes in Computer Science, vol. 435, Springer, 1989, pp. 218–238.
15. H. Niederreiter, *Knapsack-type cryptosystems and algebraic coding theory*, Problems Control Inform. Theory **15** (1986), no. 2, 159–166.
16. A. Perrig, *The BiBa one-time signature and broadcast authentication protocol*, Proceedings of the 8th ACM Conference on Computer and Communications Security, ACM Press, 2001, pp. 28–37.
17. L. Reyzin and N. Reyzin, *Better than BiBa: Short One-Time Signatures with Fast Signing and Verifying*, Information Security and Privacy, 7th Australian Conference, ACISP 2002 (L. M. Batten and J. Seberry, eds.), Lecture Notes in Computer Science, vol. 2384, Springer, 2002, pp. 144–153.
18. J. Stern, *A method for finding codewords of small weight*, Coding Theory and Applications (G. D. Cohen and J. Wolfmann, eds.), Lecture Notes in Computer Science, vol. 388, Springer, 1988, pp. 106–113.
19. J. Stern, *A new identification scheme based on syndrome decoding*, Advances in Cryptology - CRYPTO'93 (D.R. Stinson, ed.), Lecture Notes in Computer Science, vol. 773, Springer, 1993, pp. 13–21.

20. P. Véron, *Problème SD, opérateur trace, schémas d'identification et codes de goppa*, Ph.D. thesis, Université Toulon et du Var, Toulon, France, 1995.