Code-Based Identification and Signature Schemes in Software

Sidi Mohamed El Yousfi Alaoui[1], Pierre-Louis Cayrel[2], Rachid El Bansarkhani[3], and Gerhard Hoffmann[3]

[1] CASED-Center for Advanced Security Research Darmstadt, Mornewegstraße, 32 64293 Darmstadt, Germany
elyousfi@cased.de
[2] Laboratoire Hubert Curien, UMR CNRS 5516, Bâtiment F 18 rue du professeur Benoît Lauras, 42000 Saint-Etienne
pierre.louis.cayrel@univ-st-etienne.fr
[3] Technische Universität Darmstadt, Fachbereich Informatik Kryptographie und Computeralgebra, Hochschulstraße 10 64289 Darmstadt, Germany

Abstract. In this paper we present efficient implementations of several code-based identification schemes, namely the Stern scheme, the Véron scheme and the Cayrel-Véron-El Yousfi scheme. We also explain how to derive and implement signature schemes from the previous identification schemes using the Fiat-Shamir transformation. For a security of 80 bits and a document to be signed of size 1 kByte, we reach a signature in about 4 ms on a standard CPU.

Keywords: Cryptography, zero-knowledge identification, signatures, coding theory, efficient implementation.

1 Introduction

Identification schemes are very useful and fundamental tools in many applications such as electronic fund transfer and online systems for preventing data access by invalid users. Such schemes are typical applications of zero-knowledge interactive proofs [15], which are two-party protocols allowing a party called a prover to convince another party called a verifier, that it knows some secret piece of information, without the verifier being able to learn anything about the secret value except for what is revealed by the prover itself. Zero-knowledge identification schemes are of particular interest because it is possible to convert them into secure signature schemes through the very famous Fiat-Shamir paradigm [13].

Quantum computation arises much interest in cryptography, since Peter Shor found a polynomial-time algorithm to solve the factoring and discrete logarithm problems using quantum computers [21]. Therefore, it is of extreme importance to come up with cryptosystems that remain secure even when the adversary has access to a quantum computer; such systems are called post-quantum cryptosystems. One promising candidate is based on codes, since no quantum attack

A. Cuzzocrea et al. (Eds.): CD-ARES 2013 Workshops, LNCS 8128, pp. 122–136, 2013.

exists so far to solve the syndrome decoding problem on which the code-based cryptosystems are based.

Besides the fact that designing code-based identification schemes offer security against quantum attacks, these schemes have other good features. First, they are usually very fast and easy to implement compared to schemes based on number-theoretic problems as they use only matrix-vector multiplications. Second, their security is directly related to the syndrome decoding problem. Finally, the complexity of attacks against code-based identification schemes can be given in the expected number of binary operations and not only through asymptotic estimations, as in the case of lattice-based cryptosystems for example.

In 1993, Stern proposed in [23] the first efficient zero-knowledge identification scheme based on the hardness of the binary syndrome decoding problem. A few years later, Véron in [25] has designed a scheme with a lower communication cost. Recently, Cayrel-Véron-El Yousfi in [11] have designed a scheme which reduces this communication cost even more.

Code-based cryptosystems suffer from a major drawback: they require a very large public key which makes them very difficult to use in many practical situations. Using quasi-cyclic and quasi-dyadic constructions, several new constructions like [4,17] permits to reduce the size of the public matrices. Recently, there have been several structural attacks against such constructions, the first attack presented by Gauthier et al. in [24] and the second attack is due to Faugère et al. [12]; these attacks extract the private key of some parameters of these variants. We should mention that schemes using binary codes are so far unaffected by such attacks.

Our Contribution. In this paper we provide efficient implementations of the Stern, the Véron and the Cayrel-Véron-El Yousfi schemes. We also explain how to derive signature schemes from the previous identification schemes using the Fiat-Shamir paradigm. In a previous work [9], we have used Keccak [14] for the generation orandom vectors and hash values. Now we use RFSB [8] for the same purpose and juxtapose the results. In [10], the authors presented a smart implementation of the Stern scheme, but it was more a proof of concept than an efficient implementation.

Organization of the Paper. First, we give in Section 2 a general overview of code-based cryptography. Section 3 describes the Stern, Véron and Cayrel-Véron-El Yousfi (CVE) schemes. The results of our implementations will be described in Section 4. Finally, we conclude the paper in Section 5.

2 Background of Coding Theory

In this section, we recall basic facts about code-based cryptography. We refer to [6] for a general introduction to these issues.

2.1 Definitions

Linear codes are k-dimensional subspaces of an n-dimensional vector space over a finite field $\mathbb{F}_q$, where k and n are positive integers with $k < n$, and q a prime power. The theoretical error-correcting capability of such a code is the maximum number ω of errors that the code is able to decode. In short, linear codes with these parameters are denoted (n, k)-codes or $(n, n-r)$-codes, where r is the codimension of a code with $r = n - k$.

Definition 1 (Hamming weight). *The (Hamming) weight of an arbitrary vector $x \in \mathbb{F}_q^n$ is the number of its non-zero entries. We use $\mathsf{wt}(x)$ to denote the Hamming weight of x.*

The distance of vectors $x, y \in \mathbb{F}_q^n$ is defined as $\mathsf{wt}(x-y)$. The weight of $x \in \mathbb{F}_q^n$ is therefore just its distance from the null-vector $0 \in \mathbb{F}_q^n$. The minimal distance of a linear code $\mathcal{C}$ is defined as $d := \min_{x \in \mathcal{C}, x \neq 0} \mathsf{wt}(x)$. The error-correcting capability of a linear code $\mathcal{C}$ can be expressed as $\omega = \lfloor \frac{d-1}{2} \rfloor$.

Definition 2 (Generator and Parity Check Matrix). *Let $\mathcal{C}$ be a linear (n, k)-code over $\mathbb{F}_q$. A matrix $G \in \mathbb{F}_q^{k \times n}$ is called a generator matrix of $\mathcal{C}$ if its rows form a basis of $\mathcal{C}$:*

$$\mathcal{C} = \{xG : x \in \mathbb{F}_q^k\}.$$

Vectors $x \in \mathcal{C}$ are called codewords. A matrix $H \in \mathbb{F}_q^{r \times n}$ is called a parity-check matrix of $\mathcal{C}$ if

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

In other words, H is a parity-check matrix, if $GH^T = 0$ holds. A parity-check matrix H generates the dual space $\mathcal{C}^\perp$ of $\mathcal{C}$, the space perpendicular to $\mathcal{C}$.

As we have already mentioned, there have been some proposals to use quasi-cyclic or quasi-dyadic codes in order to reduce the public key size of code-based cryptosystems. The idea is to replace codes having a random parity-check matrix H by particular type of codes with a very compact representation, namely quasi-cyclic or quasi-dyadic codes. In both variants, the matrix has the form $H = (I_r|R)$, where I_r denotes the $r \times r$ identity matrix and $R \in \mathbb{F}_q^{r \times k}$ is a quasi-circulant respectively quasi-dyadic matrix. A quasi-cyclic matrix (resp. quasi-dyadic matrix) is a block matrix whose component blocks are circulant (resp. dyadic) submatrices.

A circulant matrix is defined by a vector $(a_1, a_2, \ldots, a_r) \in \mathbb{F}_q^r$ and has the following form:

$$R = \begin{pmatrix} a_1 & a_2 & a_3 & \ldots & a_r \\ a_r & a_1 & a_2 & \ldots & a_{r-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_2 & a_3 & a_4 & \ldots & a_1 \end{pmatrix}.$$

A dyadic matrix is recursively defined: any 1×1 matrix is dyadic and for $p > 1$, a $2^p \times 2^p$ dyadic matrix has the form:

$$R = \begin{pmatrix} B & C \\ C & B \end{pmatrix},$$

where B and C are $2^{p-1} \times 2^{p-1}$ dyadic matrices. To give an example, an 4×4 dyadic matrix has the following form:

$$R = \begin{pmatrix} a & b & c & d \\ b & a & d & c \\ c & d & a & b \\ d & c & b & a \end{pmatrix},$$

where $a, b, c, d \in \mathbb{F}_q$.

The advantage of a circulant resp. dyadic matrix is the fact that the whole matrix can be reconstructed from the knowledge of its first row alone. This is the trick to reduce a public key element.

We describe in the following the main hard problems on which the security of code-based schemes presented in this paper relies. We denote by $x \xleftarrow{\$} A$ the uniform random choice of x among the elements of a set A, and "$\oplus$" the exclusive disjunction (XOR) operation.

Definition 3 (Binary Syndrome Decoding Problem (SD)).
Input : $H \xleftarrow{\$} \mathbb{F}_2^{r \times n}$, $y \xleftarrow{\$} \mathbb{F}_2^r$, *and an integer* $\omega > 0$.
Find : *a word* $s \in \mathbb{F}_2^n$ *such that* $\mathsf{wt}(s) \leq \omega$ *and* $Hs^T = y$.

This problem was proven to be NP-hard in 1978 [5]. A dual version of the previous problem, using the generator matrix G instead of the parity-check matrix H of the code $\mathcal{C}$, can be defined as follows.

Definition 4 (General Decoding Problem (GD)).
Input : $G \xleftarrow{\$} \mathbb{F}_2^{k \times n}$, $y \xleftarrow{\$} \mathbb{F}_2^n$, *and an integer* $\omega > 0$.
Find : *A pair* $(m, e) \in \mathbb{F}_2^k \times \mathbb{F}_2^n$, *where* $\mathsf{wt}(e) \leq \omega$ *s.t* $mG \oplus e = y$.

Note that $x := mG \in \mathbb{F}_2^n$ for $m \in \mathbb{F}_2^k$ is by definition a codeword. In other words, GD states that given a vector $y \in \mathbb{F}_2^n$, find the (unique) codeword $x \in \mathcal{C}$, such that $\mathsf{wt}(x - y)$ is minimal. GD is also proven to be NP-hard. Moreover, it is assumed that it is hard not only for some worst-case instances, but hard on average.

An extension of the binary syndrome decoding (SD) problem over an arbitrary finite field can be formulated as well. It was proven to be NP-hard by A. Barg in 1994 [2, in russian].

Definition 5 (q-ary Syndrome Decoding (qSD) problem).
Input : $H \xleftarrow{\$} \mathbb{F}_q^{r\times n}$, $y \xleftarrow{\$} \mathbb{F}_q^r$, *and an integer* $\omega > 0$.
Find : *a word* $s \in \mathbb{F}_q^n$ *such that* $\mathsf{wt}(s) \leq \omega$ *and* $Hs^T = y$.

Best known attack. The most efficient known algorithm to attack code-based schemes is the Information Set Decoding (ISD) algorithm. Some improvement of this algorithm have been developed by Peters [19], Niebuhr et al. [18], and Bernstein et al. [7], and recently in [16] and [3]. The main idea of the ISD algorithm consists in recovering the $n - r$ information symbols as follows: the first step is to pick r of the n coordinates randomly in the hope that most of them are error-free, then try to recover the message by solving an $r \times r$ linear system (binary or over $\mathbb{F}_q$). The recent results of this attack are taken into account when choosing our parameters in order to determine the security level needed. We denote the workfactor of the Information Set Decoding algorithm by $\mathrm{WF}_{\mathrm{ISD}}$.

3 Code-Based Zero-Knowledge Identification Schemes

In code-based cryptography, there have been many attempts to design identification schemes. In such constructions, there are two main goals: on the one hand, a prover $\mathcal{P}$ wants to convince a verifier $\mathcal{V}$ of its identity. On the other hand, $\mathcal{P}$ does not want to reveal any additional information that might be used by an impersonator.

For a fixed positive integer n; let S_n denote the symmetric group of $n!$ permutations on n symbols, and let h be a public hash function. In the following, we will give an overview of three proposals in this area. The symbol "||" denotes the concatenating operator.

3.1 Stern Scheme

The first code-based zero-knowledge identification scheme was presented by Stern [23] at Crypto'93, its security is based on the syndrome decoding (SD) problem.

Description. The Stern scheme has two parts: a key generation algorithm, shown in Fig. 1, and an identification protocol as given in Fig. 2. It uses a public parity-check matrix H of the code over the binary field $\mathbb{F}_2$.

The scheme is a multiple-rounds identification protocol, where each round is a three-pass interaction between the prover and the verifier. A cheater has a probability of 2/3 per round to succeed in the protocol without the knowledge of the secret key (sk). The number of rounds depends on the impersonation resistance required. For instance to achieve the weak and strong authentication probabilities of 2^{-16} and 2^{-32} according the norm ISO/IEC-9798-5, one needs respectively 28 and 56 rounds. Stern proposed another identification protocol with five-pass [23] (like CVE in section 3.3), but it is inefficient.

KeyGen:
Let κ be the security parameter
Choose n, r, ω, such that $\mathrm{WF}_{\mathrm{ISD}}(n, r, \omega, 2) \geq 2^{\kappa}$
$H \xleftarrow{\$} \mathbb{F}_2^{r \times n}$
$s \xleftarrow{\$} \mathbb{F}_2^n$, s.t. $\mathsf{wt}(s) = \omega$.
$y \leftarrow Hs^T$
Output $(\mathrm{sk}, \mathrm{pk}) = (s, (y, H, \omega))$

Fig. 1. Stern key generation algorithm

Prover $\mathcal{P}$		**Verifier** $\mathcal{V}$
	$(\mathrm{sk}, \mathrm{pk}) = (s, (y, H, \omega)) \longleftarrow$ KeyGen (h public hash function)	
$u \xleftarrow{\$} \mathbb{F}_2^n, \sigma \xleftarrow{\$} S_n$ $c_1 \leftarrow h\left(\sigma \| Hu^T\right)$ $c_2 \leftarrow h\left(\sigma(u)\right)$ $c_3 \leftarrow h\left(\sigma(u \oplus s)\right)$	$\xrightarrow{c_1, c_2, c_3}$	
	$\xleftarrow{\text{Challenge } b}$	$b \rightarrow \{0, 1, 2\}$
If $b = 0$:	$\xrightarrow{\sigma, u}$	**Check** c_1 and c_2
if $b = 1$:	$\xrightarrow{\sigma, u \oplus s}$	**Check** c_1 and c_3,
if $b = 2$:	$\xrightarrow{\sigma(u), \sigma(s)}$	**Check** c_2 and c_3, $\mathsf{wt}(s) \stackrel{?}{=} \omega$
Note that $Hu^T = H(u \oplus s)^T + y$		

Fig. 2. Stern identification protocol

3.2 Véron Scheme

In 1996, Véron proposed in [25] a dual version of Stern's scheme, its security is based on general decoding problem (GD).

Description. The scheme uses a generator matrix instead of a parity-check matrix of the code, which has the advantage to reduce slightly the communication costs. The Véron scheme, as the Stern's one, is a multiple rounds zero-knowledge protocol, where each round is a three-pass interaction between the prover and the verifier, for which the success probability for a cheater is 2/3 in one round. The key generation algorithm part Fig. 3 and the identification protocol part Fig. 4 of the Véron's scheme are given as follows.

KeyGen:
Let κ be the security parameter
Choose n, k, and ω such that $\mathrm{WF}_{\mathrm{ISD}}(n, k, \omega, 2) \geq 2^{\kappa}$
$G \leftarrow \mathbb{F}_2^{k \times n}$
$(m, e) \leftarrow \mathbb{F}_2^k \times \mathbb{F}_2^n$, s.t. $\mathsf{wt}(e) = \omega$ ((m, e) secret key)
$y \leftarrow mG \oplus e$ (y public key)
Output $(\mathrm{sk}, \mathrm{pk}) = ((m, e), (y, G, \omega))$

Fig. 3. Véron key generation algorithm

Prover $\mathcal{P}$		**Verifier** $\mathcal{V}$
	$(\mathrm{sk}, \mathrm{pk}) = ((m, e), (y, G, \omega)) \longleftarrow$ KeyGen (h public hash function)	
$u \xleftarrow{\$} \mathbb{F}_2^k, \sigma \xleftarrow{\$} S_n$ $c_1 \leftarrow h(\sigma)$ $c_2 \leftarrow h(\sigma((u \oplus m)G))$ $c_3 \leftarrow h(\sigma(uG \oplus y))$	$\xrightarrow{c_1, c_2, c_3}$	
	$\xleftarrow{\text{Challenge } b}$	$b \rightarrow \{0, 1, 2\}$
If $b = 0$:	$\xrightarrow{\sigma, (u \oplus m)}$	**Check** c_1 and c_2
if $b = 1$:	$\xrightarrow{\sigma((u \oplus m)G), \sigma(e)}$	**Check** c_2 and c_3, $\mathsf{wt}(\sigma(e)) \stackrel{?}{=} \omega$
if $b = 2$: Note that $\sigma((u \oplus m)G) \oplus \sigma(e) = \sigma(uG \oplus y)$	$\xrightarrow{\sigma, u}$	**Check** c_1 and c_3,

Fig. 4. Véron identification protocol

3.3 CVE Identification Scheme

In 2010, Cayrel, Véron, and El Yousfi presented in [11] a five-pass identification protocol using q-ary codes instead of binary codes.

In addition to the new way of computing the commitments, the idea of this protocol uses another improvement which is inspired by [20,22]. The main achievement of this proposal is to decrease the cheating probability of each round from 2/3 for the Stern and Véron schemes to 1/2. This allows to decrease the communication complexity by obtaining the same impersonation probability in fewer rounds compared to Stern and Véron constructions.

Furthermore, this scheme offers a small public key size, about 4 kBytes, whereas that of Stern and Véron scheme is almost 15 kBytes for the same level of security. It is proven in [11] that this scheme verifies the zero-knowledge proof and its security is based on the hardness of the q-ary Syndrome Decoding (qSD) problem.

Before presenting the CVE identification scheme, we first introduce a special transformation that will be used in the protocol.

Definition 6. *Let $\Sigma \in S_n$ and $\gamma = (\gamma_1, \ldots, \gamma_n) \in (\mathbb{F}_q^*)^n$ such that $\gamma_i \neq 0$ for all i. The transformation $\Pi_{\gamma,\Sigma}$ is defined as follows:*

$$\begin{aligned} \Pi_{\gamma,\Sigma} : \mathbb{F}_q^n &\longrightarrow \mathbb{F}_q^n \\ v &\mapsto \left(\gamma_{\Sigma(1)} v_{\Sigma(1)}, \ldots, \gamma_{\Sigma(n)} v_{\Sigma(n)}\right) \end{aligned}$$

Notice that $\forall \alpha \in \mathbb{F}_q$, $\forall v \in \mathbb{F}_q^n$, $\Pi_{\gamma,\Sigma}(\alpha v) = \alpha \Pi_{\gamma,\Sigma}(v)$, and $\mathsf{wt}(\Pi_{\gamma,\Sigma}(v)) = \mathsf{wt}(v)$.

Description. The key generation algorithm is as follows: in a first step choose randomly a parity-check matrix $H \in \mathbb{F}_q^{r \times n}$ and a vector $s \in \mathbb{F}_q^n$ with weight $\mathsf{wt}(s) = \omega$. s identifies the secret key. Finally, perform Hs^T to get the vector $y \in \mathbb{F}_q^r$. The public key consists of y, H and ω (see Figure 5).

KeyGen:
Choose n, r, ω, and q such that $\mathrm{WF}_{\mathrm{ISD}}(n, r, \omega, q) \geq 2^\kappa$
$H \xleftarrow{\$} \mathbb{F}_q^{r \times n}$
$s \xleftarrow{\$} \mathbb{F}_q^n$, s.t. $\mathsf{wt}(s) = \omega$.
$y \leftarrow Hs^T$
Output $(\mathrm{sk}, \mathrm{pk}) = (s, (y, H, \omega))$

Fig. 5. CVE key generation algorithm

Prover $\mathcal{P}$		**Verifier** $\mathcal{V}$
	$(\mathrm{sk}, \mathrm{pk}) = (s, (y, H, \omega)) \longleftarrow$ KeyGen (h public hash function)	
$u \xleftarrow{\$} \mathbb{F}_q^n, \Sigma \xleftarrow{\$} S_n$		
$\gamma \xleftarrow{\$} (\mathbb{F}_q^*)^n$		
$c_1 \leftarrow h\left(\Sigma, \gamma, Hu^T\right)$		
$c_2 \leftarrow h\left(\Pi_{\gamma,\Sigma}(u), \Pi_{\gamma,\Sigma}(s)\right)$	$\xrightarrow{c_1, c_2}$	
	$\xleftarrow{\alpha}$	$\alpha \xleftarrow{\$} \mathbb{F}_q^*$
$\beta \longleftarrow \Pi_{\gamma,\Sigma}(u + \alpha s)$	$\xrightarrow{\beta}$	
	$\xleftarrow{\text{Challenge } b}$	$b \xleftarrow{\$} \{0, 1\}$
If $b = 0$:	$\xrightarrow{\Sigma, \gamma}$	**Check** $c_1 \stackrel{?}{=} h(\Sigma, \gamma, H\Pi_{\gamma,\Sigma}^{-1}(\beta)^T - \alpha y)$
Else:	$\xrightarrow{\Pi_{\gamma,\Sigma}(s)}$	**Check** $c_2 \stackrel{?}{=} h(\beta - \alpha \Pi_{\gamma,\Sigma}(s), \Pi_{\gamma,\Sigma}(s))$, $\mathsf{wt}(\Pi_{\gamma,\Sigma}(s)) \stackrel{?}{=} \omega$

Fig. 6. CVE identification protocol

3.4 Signature Schemes via the Fiat-Shamir Transform

Using the Fiat-Shamir transform [13], respectively its extended version [1], it is possible to transform the Stern and Véron schemes, respectively the CVE scheme, given above into signature schemes. The idea of this transformation is to split the identification scheme in two parts.

In the first part, the signer runs the identification scheme as before, but without any verifier involved. Instead, the signer has to generate the challenges on his own, for instance using a stream cipher with a predefined start value, which includes the message to sign. On the one hand, the signer can not predict the next challenge bits, and on the other hand, the procedure must be repeatable by the verifier. In other words, concerning the challenges, the signer is simulating the role of the verifier, and recording the responses without any checks.

In the second part, the verifier uses the same stream cipher and starting value and replays the protocol with the saved responses and performs the necessary checks. This also explains the relatively big signature sizes of schemes based on the Fiat-Shamir transform as the signer is recording a history of the actions involved. This history is used by the verifier in the verification process. It also shows the varying sizes of the signatures, as the given responses change from run to run with high probability.

4 Implementation

In total, six different schemes have been implemented in **C**: the Stern, Véron and CVE identification schemes and the corresponding signature schemes based on the Fiat-Shamir transform [13,1].

The implementation assumes that the dimensions of the matrices are a multiple of 64. The public keys G and H are given in systematic form, i.e. $G = [I_k|R]$ and $H = [I_{n-k}|R]$ respectively, where only the redundant part R is used. In the quasi-cyclic and quasi-dyadic cases, the matrices G and H consist of cyclic and dyadic submatrices of size 64×64, because 64 is the natural number to use on a 64-bit machine.

For the generation of random vectors and hash values, we deployed the code-based RFSB-509 hash function presented by Bernstein et al. [8]. This choice is driven by the intension to base the security of the schemes on only one hardness assumption, namely the hardness of solving the syndrome decoding problem. But note that it can be replaced by any other suitable scheme providing the necessary functionality: for comparison, we also implemented the signature schemes using Keccak [14].

The experiments were performed on an Intel Xeon E5-1602 running at 2.80 GHz, having 8 GB of RAM and running a 64bit version of Debian 6.0.6.

4.1 Identification Schemes

Stern Scheme. This scheme uses a binary parity check matrix $H = [I_{n-k}|R]$ of size $r \times n$, where $r = n - k$ and $k = n/2$. For the implementation we used

$n = 768$ and $k = 384$. Due to the row-major order of $\mathbf{C}$, the product sH^T is more efficient as Hs^T ($s \in \mathbb{F}_2^n$). Hence, the implementation uses the transposed matrix H^T instead of H.

Table 1. Stern timing results for 28 rounds when the impersonation probability is bounded by 2^{-16}

Matrix Type	**Dimension** $[n \times r]$	**Weight**	**Time** $[ms]$	**Sec. Level**$_{[bits]}$
Random	768×384	76	2.79	80
Quasi-cyclic	768×384	76	2.57	80
Quasi-dyadic	768×384	76	3.78	80

Véron Scheme: This scheme uses a binary generator matrix $G = [I_k|R]$ of dimensions $k \times n$, where $k = n/2$. Again, in the quasi-cyclic and quasi-dyadic case, the cyclic and dyadic submatrices have a size of 64×64 bits, $n = 768$ and $k = 384$. As in Stern, if G is quasi-cyclic or quasi-dyadic, then the submatrix R would consist of 36 cyclic or dyadic submatrices of size 64×64 bits.

Table 2. Véron timing results for 28 rounds when the impersonation probability is bounded by 2^{-16}

Matrix Type	**Dimension** $[k \times n]$	**Weight**	**Time** $[ms]$	**Sec. Level**$_{[bits]}$
Random	768×384	76	2.65	80
Quasi-cyclic	768×384	76	2.47	80
Quasi-dyadic	768×384	76	3.57	80

Memory Requirements. The memory requirements for the Stern and Véron scheme are as follows: using a random matrix $384 \times 384 = 147.456$ bits are necessary to store the redundancy part R of H resp. G. Using quasi-cyclic (quasi-dyadic) matrices, the memory footprint for the matrices drops by a factor of 64. Only $6 \times 6 \times 64 = 2.304 = 147.456/64$ bits are needed. Hence, although the timings using quasi-cyclic (quasi-dyadic) matrices are worse than for random matrices, in some environments the smaller memory footprint might compensate for the loss in performance.

CVE Scheme. It uses a parity check matrix H of size $r \times n$ over $\mathbb{F}_q$, where $q = 2^m$, $1 \leq m \leq 16$, $r = n - k$ and $k = n/2$. As in the Stern scheme, the implementation uses the transposed matrix H^T instead of H. If H is quasi-cyclic or quasi-dyadic, then the submatrix R would consist of 81 cyclic or dyadic submatrices of 8×8 field elements.

The matrix size is always measured in numbers of field elements. Each field element occupies invariably 2 bytes of memory. Strictly speaking, this would be necessary only in the case $m = 16$. However, using only the necessary bits would

complicate the code and slow down the computation. In environments in which memory is a very valuable resource, this fact had to be taken into account.

For the measurements we used $m = 8$.

Table 3. CVE timing results for 16 rounds when the impersonation probability is bounded by 2^{-16}

Matrix Type	**Dimension** $[n \times r]$	**Degree** $q = 2^m$	**Weight**	**Time** $[ms]$	**Sec. Level**$_{[bits]}$
Random	144×72	256	55	1.40	80
Quasi-cyclic	144×72	256	55	1.38	80
Quasi-dyadic	144×72	256	55	1.67	80

Memory Requirements for the CVE Scheme. Using a random matrix, $72 \times 72 \times 2 = 10.368$ bytes are necessary to store the redundancy part R of H resp. G. Using quasi-cyclic (quasi-dyadic) matrices, the memory footprint for the matrices drops by a factor of 8, because in this case only $9 \times 9 \times 8 \times 2 = 1.296 = 10.368/8$ bytes are needed. Again, as with the Stern and Véron scheme, memory savings using the structured matrix types might be more important than the loss in runtime.

4.2 Signature Schemes Based on Fiat-Shamir Transform

Using the Fiat-Shamir transform [13], one can transform identification schemes to signature schemes. We describe the process in detail for the Stern scheme (respectively Véron scheme). It is straightforward to adapt it to the non canonical (more than three-pass) CVE case, see [1] for more details.

Note that the signer and verifier parts are always located in the same executable, thus the two parts can communicate in almost no time. In reality, they would reside on different machines, such that additional costs over some communication link had to be taken into account.

4.3 The Signing Procedure

Let δ the number of rounds needed to achieve the required cheating probability. In a first step, a commitment CMT is computed as

$$\text{CMT} = (c_{01}, c_{02}, c_{03}) \mid\mid (c_{11}, c_{12}, c_{13}) \mid\mid \ \ldots \ \mid\mid (c_{\delta-1,1}, c_{\delta-1,2}, c_{\delta-1,3}).$$

More precisely, in each round we run one of the above identification schemes to generate a corresponding commitment (c_{i1}, c_{i2}, c_{i3}), where $0 \leq i \leq \delta - 1$. Note that the c_{i1}, c_{i2} and c_{i3} are hashed values (using RFSB-509) and that each such triple has $3 \times 160 = 480$ bits. The number of rounds δ is a predefined value (e.g. 141 for Stern and Véron schemes or 80 for the CVE scheme to achieve a impersonation resistance of $1/2^{80}$). All round-triples together form the compound commitment CMT.

In a second step, we compute the challenge CH $= h(\text{CMT} \parallel M)$, where h denotes the RFSB-509 hash function and M is the message, typically the content of some file. CH has a length such that it consists of twice as many bits as there are rounds, because for each round the signer needs a new challenge. Each two bits of CH give a partial challenge, where the bit pattern 11 is mapped to $b \in \{0, 1, 2\}$ in a cyclic fashion.

Finally, compute for each partial challenge b the response according to the deployed identification scheme. Note that the response size for each b varies depending on its actual value of 0, 1 or 2. Denote all responses by RSP $= (r_0 \parallel r_1 \parallel \ldots \parallel r_{\delta-1})$. The final signature is (CMT $\parallel$ RSP).

4.4 The Verification Procedure

Upon receiving the signature, the verifier extracts CMT and computes CH $= h(\text{CMT} \parallel M)$. As in the signing step, the verifier uses the individual bytes of CH modulo 3 to obtain δ many challenges $b \in \{0, 1, 2\}$. Using b, the verifier extracts the corresponding response contained in RSP and calculates the commitment c_{ij}, where $j = b$ and i denotes the current round. Finally, the verifier computes $h(c_{ij})$ of CMT and compares this value with the c_{ij} contained in the triple (c_{i1}, c_{i2}, c_{i3}). We identify here the value $h(c_{ij})$ and c_{ij}. In case the values of c_{ij} match for all rounds, the signature is considered valid.

In the following, tables are given for runtime measurement of the three signature schemes derived from the corresponding identification schemes using the Fiat-Shamir transform.

4.5 Signature Scheme Timings

Table 4. Stern timing results: separate signing and verification time (s/v) for 141 rounds

Matrix Type	**Dimension**$_{[n \times r]}$	**Weight**	**Time**$_{[ms]}$				**Msg.**$_{[kBytes]}$	**Sec.**$_{[bits]}$
			Keccak		**RFSB**			
Random	768 × 384	76	7.18	3.57	7.67	4.88	1	80
	768 × 384	76	7.32	3.92	7.88	3.70	10	80
	768 × 384	76	7.49	4.02	8.11	3.93	25	80
Quasi-cyclic	768 × 384	76	6.59	3.25	7.01	4.59	1	80
	768 × 384	76	6.70	3.35	7.18	4.16	10	80
	768 × 384	76	6.84	3.49	7.44	4.03	25	80
Quasi-dyadic	768 × 384	76	10.13	5.61	10.46	6.07	1	80
	768 × 384	76	10.25	5.64	10.87	4.71	10	80
	768 × 384	76	10.49	5.65	10.97	7.77	25	80

4.6 Remarks

The signature size in the Stern and Véron schemes is about 25 kBytes and respectively 19 kBytes in the CVE scheme for 80-bit security. The numbers mean an average value of several runs over 141 resp. 80 rounds. Note that the signature size is independent from the message to be signed.

Table 5. Véron timing results: separate signing and verification time (s/v) for 141 rounds

Matrix Type	**Dimension**$_{[n\times r]}$	**Weight**	**Time**$_{[ms]}$				**Msg.**$_{[kBytes]}$	**Sec.**$_{[bits]}$
			Keccak		**RFSB**			
Random	768×384	76	10.26	4.86	7.98	3.76	1	80
	768×384	76	10.37	5.53	7.99	4.22	10	80
	768×384	76	10.55	5.26	8.33	4.14	25	80
Quasi-cyclic	768×384	76	9.37	4.80	6.87	3.89	1	80
	768×384	76	9.47	4.92	7.25	3.33	10	80
	768×384	76	9.64	5.07	7.35	3.89	25	80
Quasi-dyadic	768×384	76	13.79	6.23	11.66	4.17	1	80
	768×384	76	14.23	7.01	11.99	3.88	10	80
	768×384	76	14.02	6.13	12.40	3.65	25	80

Table 6. CVE timing results: separate signing and verification time (s/v) for 80 rounds

Matrix Type	**Dimension**$_{[n\times r]}$	**Weight**	**Time**$_{[ms]}$				**Msg.**$_{[kBytes]}$	**Sec.**$_{[bits]}$
			Keccak		**RFSB**			
Random	144×72	55	4.25	1.90	4.21	3.73	1	80
	144×72	55	4.38	2.03	4.36	2.40	10	80
	144×72	55	4.59	2.97	4.59	2.30	25	80
Quasi-cyclic	144×72	55	5.21	2.42	5.20	2.62	1	80
	144×72	55	5.32	2.51	5.31	3.02	10	80
	144×72	55	5.56	3.70	5.55	2.80	25	80
Quasi-dyadic	144×72	55	4.44	2.07	4.41	2.18	1	80
	144×72	55	4.58	2.13	4.56	2.53	10	80
	144×72	55	4.79	3.13	4.77	2.42	25	80

The runtime is dominated by RFSB creating random vectors $u[0], \ldots, u[\delta - 1]$ before entering the loop of 141 resp. 80 rounds, which could also be confirmed profiling the implementation directly with gprof, the profiler contained in gcc.

5 Conclusion

In this paper, we have described three existing code-based identification and their corresponding signature schemes and provided running times of their implementation. As a result, we obtain three very fast signature schemes. Depending on the message size it is possible to sign and verify in the order of milliseconds, but at the cost of very long signature sizes: typically 19 kBytes for CVE and 25 kBytes bytes for Stern resp. Véron.

The source code of the **C** implementation is available under the following link: `http://cayrel.net/research/code-based-cryptography/code-based-cryptosystems/article/implementation-of-code-based-zero`.

References

1. El Yousfi Alaoui, S.M., Dagdelen, Ö., Véron, P., Galindo, D., Cayrel, P.-L.: Extended security arguments for signature schemes. In: Mitrokotsa, A., Vaudenay, S. (eds.) AFRICACRYPT 2012. LNCS, vol. 7374, pp. 19–34. Springer, Heidelberg (2012)
2. Barg, S.: Some new NP-complete coding problems. Probl. Peredachi Inf. 30, 23–28 (1994)
3. Becker, A., Joux, A., May, A., Meurer, A.: Decoding random binary linear codes in $2^{(n/20)}$: How 1+1=0 improves information set decoding. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 520–536. Springer, Heidelberg (2012)
4. Berger, T.P., Cayrel, P.-L., Gaborit, P., Otmani, A.: Reducing Key Length of the McEliece Cryptosystem. In: Preneel, B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 77–97. Springer, Heidelberg (2009)
5. Berlekamp, E., McEliece, R., van Tilborg, H.: On the Inherent Intractability of Certain Coding Problems. IEEE Transactions on Information Theory 24(3), 384–386 (1978)
6. Bernstein, D.J., Buchmann, J., Dahmen, E.: Post-Quantum Cryptography. Springer (2008)
7. Bernstein, D.J., Lange, T., Peters, C.: Smaller decoding exponents: ball-collision decoding. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 743–760. Springer, Heidelberg (2011)
8. Bernstein, D.J., Lange, T., Peters, C., Schwabe, P.: Really fast syndrome-based hashing. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 134–152. Springer, Heidelberg (2011)
9. Cayrel, P.-L., El Yousfi Alaoui, S.M., Günther, F., Hoffmann, G., Rother, H.: Efficient implementation of code-based identification schemes (2011), `http://cayrel.net/PublicationsCayrel/2011-Efficient%implementation%of%code-based%identification%schemes.pdf`
10. Cayrel, P.-L., Gaborit, P., Prouff, E.: Secure Implementation of the Stern Authentication and Signature Schemes for Low-Resource Devices. In: Grimaud, G., Standaert, F.-X. (eds.) CARDIS 2008. LNCS, vol. 5189, pp. 191–205. Springer, Heidelberg (2008)
11. Cayrel, P.-L., Véron, P., El Yousfi Alaoui, S.M.: A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 171–186. Springer, Heidelberg (2011)

12. Faugère, J.-C., Otmani, A., Perret, L., Tillich, J.-P.: Algebraic cryptanalysis of mceliece variants with compact keys. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 279–298. Springer, Heidelberg (2010)
13. Fiat, A., Shamir, A.: How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
14. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: The Keccak sponge function family, `http://keccak.noekeon.org/`
15. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems. In: STOC 1985. acmid, vol. 22178, pp. 291–304. ACM (1985)
16. May, A., Meurer, A., Thomae, E.: Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 107–124. Springer, Heidelberg (2011)
17. Misoczki, R., Barreto, P.S.L.M.: Compact McEliece keys from Goppa Codes. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009. LNCS, vol. 5867, pp. 376–392. Springer, Heidelberg (2009)
18. Niebuhr, R., Cayrel, P.-L., Bulygin, S., Buchmann, J.: On Lower Bounds for Information Set Decoding over $\mathbb{F}_q$. In: SCC 2010, Rhul, London, UK (2010)
19. Peters, C.: Information-Set Decoding for Linear Codes over $\mathbb{F}_q$. In: Sendrier, N. (ed.) PQCrypto 2010. LNCS, vol. 6061, pp. 81–94. Springer, Heidelberg (2010)
20. Shamir, A.: An efficient identification scheme based on permuted kernels (extended abstract). In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 606–609. Springer, Heidelberg (1990)
21. Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM Journal on Computing 26, 1484–1509 (1995)
22. Stern, J.: Designing Identification Schemes with Keys of Short Size. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 164–173. Springer, Heidelberg (1994)
23. Stern, J.: A New Identification Scheme Based on Syndrome Decoding. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 13–21. Springer, Heidelberg (1994)
24. Gauthier Umana, V., Leander, G.: Practical key recovery attacks on two McEliece variants. IACR Cryptology ePrint Archive, Report 2009/509 (2009)
25. Véron, P.: Improved Identification Schemes Based on Error-Correcting Codes. Appl. Algebra Eng. Commun. Comput. 8(1), 57–69 (1996)