On the KKS Scheme

Ayoub Otmani[1,2] Jean-Pierre Tillich[1]

[1] SECRET Project - INRIA Rocquencourt
Domaine de Voluceau, B.P. 105 78153 Le Chesnay Cedex - France
ayoub.otmani@inria.fr, jean-pierre.tillich@inria.fr
[2] GREYC - Université de Caen - Ensicaen
Boulevard Maréchal Juin, 14050 Caen Cedex, France.

1 Terminology and Notation

In the whole paper q denotes some prime power and we denote by $\mathbb{F}_q$ the finite field with q elements. Let n be a non-negative integer. The set of integers i such that $1 \leqslant i \leqslant n$ is denoted by $1n$. The cardinality of a set A is denoted by A. The concatenation of the vectors $\boldsymbol{x} = (x_1, \dots, x_n)$ and $\boldsymbol{y} = (y_1, \dots, y_m)$ is denoted by $(\boldsymbol{x}||\boldsymbol{y}) \stackrel{\text{def}}{=} (x_1, \dots, x_n, y_1, \dots, y_m)$. The support $\mathsf{supp}(\boldsymbol{x})$ of $\boldsymbol{x} \in \mathbb{F}_q^n$ is the set of i's such that $x_i \neq 0$. The *(Hamming) weight* $\mathsf{wt}(\boldsymbol{x})$ is the cardinality of $\mathsf{supp}(\boldsymbol{x})$. For a vector $\boldsymbol{x} = (x_i)$ and a subset I of indices of $\boldsymbol{x}$, we denote by $\boldsymbol{x}_I$ its restriction to the indices of I, that is:

$$\boldsymbol{x}_I \stackrel{\text{def}}{=} (x_i)_{i \in I}.$$

We will also use this notation for matrices, in this case it stands for the submatrix formed by the columns in the index set, i.e. for any $k \times n$ matrix $\boldsymbol{H}$

$$\boldsymbol{H}_J \stackrel{\text{def}}{=} (h_{ij})_{\substack{1 \leqslant i \leqslant k \\ j \in J}}.$$

A linear code $\mathscr{C}$ of type $[n, k, d]$ over $\mathbb{F}_q$ is a linear subspace of $\mathbb{F}_q^n$ of dimension k and minimum distance d where by definition $d \stackrel{\text{def}}{=} \min\{\mathsf{wt}(\boldsymbol{x}) \ : \ \boldsymbol{x} \in \mathscr{C} \text{ and } \boldsymbol{x} \neq \boldsymbol{0}\}$. The elements of $\mathscr{C}$ are *codewords*. A linear code can be defined either by a parity check matrix or a generator matrix. A *parity check matrix* $\boldsymbol{H}$ for $\mathscr{C}$ is an $(n-k) \times n$ matrix such that $\mathscr{C}$ is the right kernel of $\boldsymbol{H}$:

$$\mathscr{C} = \{\boldsymbol{c} \in \mathbb{F}_q^n \ : \ \boldsymbol{H} c^T = 0\}$$

where $\boldsymbol{x}^T$ denotes the *transpose* of $\boldsymbol{x}$. A generator matrix $\boldsymbol{G}$ is a $k \times n$ matrix formed by a basis of $\mathscr{C}$. We say that $\boldsymbol{G}$ is in *systematic* form if there exists a set J such that $\boldsymbol{G}_J = \boldsymbol{I}_k$. The *syndrome* $\boldsymbol{u}$ by $\boldsymbol{H}$ of $\boldsymbol{x} \in \mathbb{F}_q^n$ is defined as $\boldsymbol{u}^T \stackrel{\text{def}}{=} \boldsymbol{H}\boldsymbol{x}^T$. A *decoding* algorithm for $\boldsymbol{H}$ is an algorithm such that, given $\boldsymbol{u}$ in $\mathbb{F}_q^r$, finds a vector $\boldsymbol{e}$ of *minimum* weight whose syndrome is $\boldsymbol{u}$.

2 The McEliece scheme based on convolutional codes

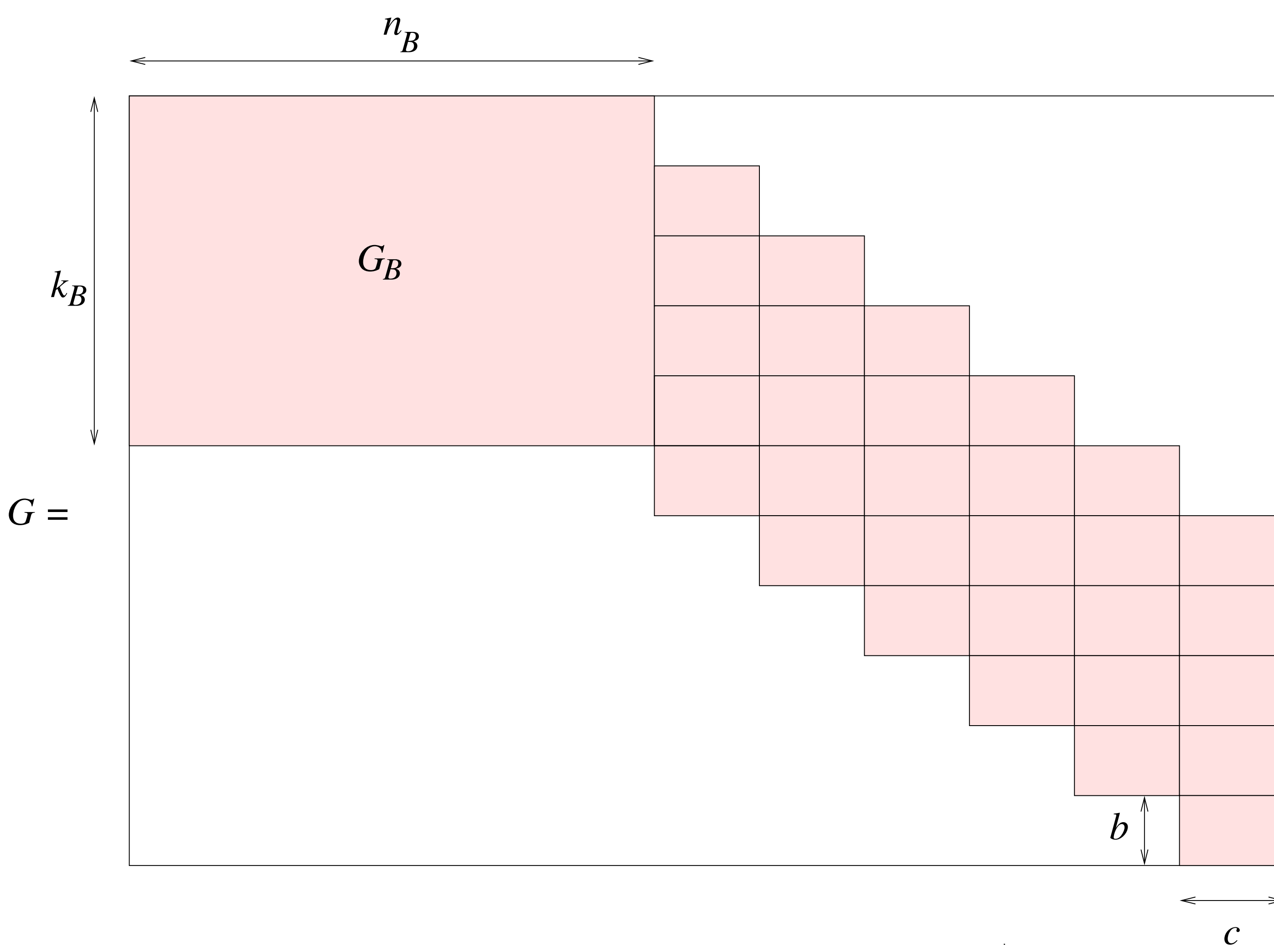
The scheme can be summarized as follows.

Secret key.
- $\boldsymbol{G}$ is a generator matrix which has a block form specified in Figure **??**
- $\stackrel{\text{def}}{=} +$ where is an $n \times n$ permutation matrix,
- is a rank-one matrix over such that is invertible,
- $\boldsymbol{S}$ is a $k \times k$ random invertible matrix over .

Public key. $\stackrel{\text{def}}{=} \boldsymbol{S}^{-1-1}$.

Encryption. The ciphertext $\boldsymbol{c} \in^n$ of a plaintext $\in^k$ is obtained by drawing at random $\boldsymbol{e}$ in ^{n} of weight less than or equal to $\frac{n-k}{2}$ and computing $\boldsymbol{c} \stackrel{\text{def}}{=} +\boldsymbol{e}$.

Decryption. It consists in performing the three following steps:


n_B
k_B
G_B
$G =$
b
c

1. Guessing the value of $\boldsymbol{e}$;
2. Calculating $\boldsymbol{c}' \stackrel{\text{def}}{=} \boldsymbol{c} - \boldsymbol{e} = \boldsymbol{S}^{-1} + \boldsymbol{e} - \boldsymbol{e} = \boldsymbol{S}^{-1} + \boldsymbol{e}$ and using the decoding algorithm of the generalized Reed-Solomon code to recover $\boldsymbol{S}^{-1}$ from the knowledge of $\boldsymbol{c}'$;
3. Multiplying the result of the decoding by $\boldsymbol{S}$ to recover .

This section is devoted to the description of two code-based signature schemes proposed in [KKS97] and more recently in [BMJ11], where the latter can be viewed as a "noisy" version of the former [KKS97]. Our presentation presents the main ideas without giving all the details which can be found in the original papers. We first focus on the scheme of [KKS97] whose construction relies on the following ingredients:

1. a full rank binary matrix $\boldsymbol{H}$ of size $(N-K) \times N$ with entries in a finite field $\mathbb{F}_q$.
2. a subset J of $\{1, \ldots, N\}$ of cardinality n,
3. a linear code over $\mathbb{F}_q$ of length $n \leqslant N$ and dimension k defined by a generator matrix $\boldsymbol{G}$ of size $k \times n$. Let t_1 and t_2 be two integers such that with very high probability, we have that $t_1 \leqslant \mathsf{wt}(\boldsymbol{u}) \leqslant t_2$ for any non-zero codeword $\boldsymbol{u} \in$.

The matrix $\boldsymbol{H}$ is chosen such that the best decoding algorithms cannot solve the following search problem.

Problem 1. Given the knowledge of $\boldsymbol{u} \in \mathbb{F}_q^{N-K}$ which is the syndrome by $\boldsymbol{H}$ of some $\boldsymbol{e} \in \mathbb{F}_q^N$ whose weight lies in $t_1 t_2$, find explicitly $\boldsymbol{e}$, or eventually $\boldsymbol{x}$ in $\mathbb{F}_q^N$ different from $\boldsymbol{e}$ sharing the same properties as $\boldsymbol{e}$.

Finally let $\boldsymbol{F}$ be the $(N-K) \times k$ matrix defined by $\boldsymbol{F} \stackrel{\text{def}}{=} \boldsymbol{H}_J \boldsymbol{G}^T$. The Kabatianskii-Krouk-Smeets (KKS) signature scheme is then described in Figure 1.

Fig. 2. Description of the KKS scheme given in [KKS97].

- Setup.
  1. The signer S chooses N, K n, k, t_1 and t_2 according to the required security level.
  2. S draws a random $(N-K) \times N$ matrix $\boldsymbol{H}$.
  3. S randomly picks a subset J of $\{1, \ldots, N\}$ of cardinality n.
  4. S randomly picks a random $k \times n$ generator matrix $\boldsymbol{G}$ that defines a code such that with high probability $t_1 \leqslant \mathsf{wt}(\boldsymbol{u}) \leqslant t_2$ for any non-zero codeword $\boldsymbol{u} \in$.
  5. $\boldsymbol{F} \stackrel{\text{def}}{=} \boldsymbol{H}_J \boldsymbol{G}^T$ where $\boldsymbol{H}_J$ is the restriction of $\boldsymbol{H}$ to the columns in J.
- Keys.
  - Private key. J and $\boldsymbol{G}$
  - Public key. $\boldsymbol{F}$ and $\boldsymbol{H}$
- Signature. The signature σ of a message $\boldsymbol{x} \in \mathbb{F}_q^k$ is defined as the unique vector σ of $\mathbb{F}_q^N$ such that $\sigma_i = 0$ for any $i \notin J$ and $\sigma_J = \boldsymbol{x}\boldsymbol{G}$.
- Verification. Given $(\boldsymbol{x}, \sigma) \in \mathbb{F}_q^k \times \mathbb{F}_q^N$, the verifier checks that $t_1 \leqslant \mathsf{wt}(\sigma) \leqslant t_2$ and $\boldsymbol{H}\sigma^T = \boldsymbol{F}\boldsymbol{x}^T$.

The scheme was modified in [BMJ11] to propose a one-time signature scheme by introducing two new ingredients, namely a hash function f and adding an error vector $\boldsymbol{e}$ to the signature. It was proved that such a scheme is EUF-1CMA secure in the random oracle model. The description is given in Figure **??**.

3 Description of the Attack

The purpose of this section is to explain the idea underlying our attack which aims at recovering the private key. The attack is divided in two main steps. The first step consists in a (partial) key

Fig. 3. Description of the scheme of [BMJ11].

– Setup.
  1. The signer S chooses N, K n, k, t_1 and t_2 according to the required security level.
  2. S chooses a hash function $f : \{0,1\}^* \times \mathbb{F}_2^{N-K} \longrightarrow \mathbb{F}_2^k$.
  3. S draws a random binary $(N-K) \times N$ matrix $\boldsymbol{H}$.
  4. S randomly picks a subset J of $\{1,\dots,N\}$ of cardinality n.
  5. S randomly picks a $k \times n$ generator matrix $\boldsymbol{G}$ that defines a binary code such that with high probability $t_1 \leqslant \mathsf{wt}(\boldsymbol{u}) \leqslant t_2$ for any non-zero codeword $\boldsymbol{u} \in$.
  6. $\boldsymbol{F} \stackrel{\text{def}}{=} \boldsymbol{H}_J \boldsymbol{G}^T$ where $\boldsymbol{H}_J$ is the restriction of $\boldsymbol{H}$ to the columns in J.
– Keys.
  • Private key. J and $\boldsymbol{G}$
  • Public key. $\boldsymbol{F}$ and $\boldsymbol{H}$
– Signature. The signature of a message $\boldsymbol{x} \in \{0,1\}^*$ is (h,σ) defined as follows:
  • S picks a random $\boldsymbol{e} \in \mathbb{F}_2^N$ such that $\mathsf{wt}(\boldsymbol{e}) = n$.
  • Let $\boldsymbol{h} \stackrel{\text{def}}{=} f(\boldsymbol{x}, \boldsymbol{H}\boldsymbol{e}^T)$ and $\boldsymbol{y}$ be the unique vector of $\mathbb{F}_2^N$ such that (i) $\mathsf{supp}(\boldsymbol{y}) \subset J$, (ii) $\boldsymbol{y}_J = \boldsymbol{h}\boldsymbol{G}$. The second part of the signature σ is then given by $\sigma \stackrel{\text{def}}{=} \boldsymbol{y} + \boldsymbol{e}$.
– Verification. Given a signature $(\boldsymbol{h},\sigma) \in \mathbb{F}_2^k \times \mathbb{F}_2^N$ for $\boldsymbol{x} \in \{0,1\}^*$, the verifier checks that $\mathsf{wt}(\sigma) \leqslant 2n$ and $\boldsymbol{h} = f(\boldsymbol{x}, \boldsymbol{H}\sigma^T + \boldsymbol{F}\boldsymbol{h}^T)$.

recovery attack aiming at unraveling the convolutional structure. The second part consists in a message recovery attack taking advantage of the fact that if the convolutional part is recovered, then an attacker can decrypt a message if he is able to solve the following decoding pro

First, we produce a valid signature for some message using only the public key. To do so, we define a certain code from matrices $\boldsymbol{H}$ and $\boldsymbol{F}$. It turns out that low weight codewords of this code give valid message-signature pairs. Then we just apply Dumer's algorithm [**?**] in order to find these low weight codewords. This attack can even be refined in the following way. Whenever we are able to produce one valid message-signature pair, and since each signature reveals partial information about the private key (especially about J as explained further in this section), we can use it to get another valid message-signature pair revealing more information about J. We repeat this process a few times until we totally recover the whole private key. More details will be given in the following sections.

In what follows, we make the assumption that all the codes are binary because all the concrete proposals are of this kind. The non-binary case will be discussed in the conclusion.

3.1 An auxiliary code

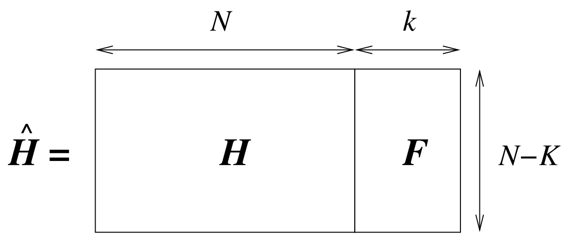
We give here the first ingredient we use to forge a valid message/signature pair for the KKS scheme just from the knowledge of the public pair $\boldsymbol{H}, \boldsymbol{F}$. This attack can also be used for the second scheme given by Figure **??**. In the last case, it is not a valid message/signature pair anymore but an auxiliary quantity which helps in revealing J. This ingredient consists in a linear code $\mathscr{C}_{\text{pub}}$ of length $N+k$ defined as the kernel of $\hat{\boldsymbol{H}}$ which is obtained by the juxtaposition of the two public matrices $\boldsymbol{H}$ and $\boldsymbol{F}$ as given in Figure **??**. The reason behind this definition lies in the following Fact **??**.

Fact 1. *Let $\boldsymbol{x}'$ be in $\mathbb{F}_2^{N+k}$ and set $(\sigma||\boldsymbol{x}) \stackrel{def}{=} \boldsymbol{x}'$ with σ in $\mathbb{F}_2^N$ and $\boldsymbol{x}$ in $\mathbb{F}_2^k$. Then σ is a signature of $\boldsymbol{x}$ if and only if:*

1. $\hat{\boldsymbol{H}}\boldsymbol{x}'^T = 0$
2. $t_1 \leqslant \mathsf{wt}(\sigma) \leqslant t_2$.

The code $\mathscr{C}_{\text{pub}}$ is of dimension $k+K$, and of particular interest is the linear space $\mathscr{C}_{\text{sec}} \subset \mathscr{C}_{\text{pub}}$ that consists in words that satisfy both conditions of Fact **??** and that are obtained by all pairs

Fig. 4. Parity-check matrix $\hat{\boldsymbol{H}}$ of the code $\mathscr{C}_{\text{pub}}$



$(\sigma, \boldsymbol{x})$ of valid message-signature pairs which are obtained by the secret signature algorithm, that is to say:

$$\mathscr{C}_{\text{sec}} \stackrel{\text{def}}{=} \left\{ (\sigma || \boldsymbol{x}) \in \mathbb{F}_2^{N+k} \ : \ \boldsymbol{x} \in \mathbb{F}_2^k, \ \sigma \in \mathbb{F}_2^N, \ \sigma_J = \boldsymbol{x}\boldsymbol{G}, \ \sigma_{1N \setminus J} = 0 \right\}. \quad (1)$$

Clearly, the dimension of $\mathscr{C}_{\text{sec}}$ is k. Additionally, we expect that the weight of σ is of order $n/2$ for any $(\sigma, \boldsymbol{x})$ in $\mathscr{C}_{\text{sec}}$, which is much smaller than the total length N. This strongly suggests to use well-known algorithms for finding low weight codewords to reveal codewords in $\mathscr{C}_{\text{sec}}$ and therefore message-signature pairs. The algorithm we used for that purpose is specified in the following subsection.

3.2 Finding low-weight codewords

We propose to use the following variation on Stern's algorithm due to [**?**] (See also [**?**]). The description of the algorithm is given in Algorithm 1. It consists in searching for low-weight codewords among the candidates that are of very low-weight $2p$ (where p is typically in the range $1 \leqslant p \leqslant 4$) when restricted to a set I of size slightly larger than the dimension $k + K$ of the code $\mathscr{C}_{\text{pub}}$, say $|I| = k + K + l$ for some small integer l. The key point in this approach is to choose I among a set S of test positions. The set S will be appropriately chosen according to the considered context. If no signature pair is known, then a good choice for S is to take:

$$S = [1 \cdots N]. \quad (2)$$

This means that we always choose the test positions among the N first positions of the code $\mathscr{C}_{\text{pub}}$ and never among the k last positions. The reason for this choice will be explained in the following subsection.

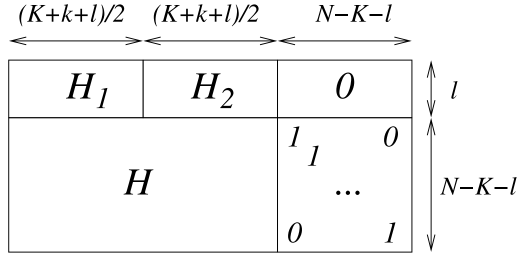


Fig. 5. A parity-check matrix for $\mathscr{C}_{\text{pub}}$ in quasi-systematic form.

Algorithm 1 KKSforge: algorithm that forges a valid KKS signature.

PARAMETERS:

r : number of iterations,
l : small integer ($l \leqslant 40$),
p : very small integer ($1 \leqslant p \leqslant 4$).
S : a subset of $1N$ from which in each iteration a subset of cardinality $K + k + l$ will be randomly chosen.

INPUT: $\hat{\boldsymbol{H}}$
OUTPUT: a list $\mathcal{L}$ containing valid signature/message pairs $(\sigma, \boldsymbol{x}) \in \mathbb{F}_2^N \times \mathbb{F}_2^k$.

$\mathcal{L} \leftarrow \emptyset$.
for $1 \leqslant t \leqslant r$ **do**
  Step 1: Randomly pick $K + k + l$ positions among S to form the set I. This set is partitioned into $I = I_1 \cup I_2$ such that $||I_1| - |I_2|| \leqslant 1$.
  Step 2: Perform Gaussian elimination over the complementary set $\{1, 2, \ldots, N + k\} \setminus I$ to put $\hat{\boldsymbol{H}}$ in quasi-systematic form (as shown in Figure 2).
  Step 3:
  Generate all binary vectors $\boldsymbol{x}_1$ of length $\lfloor (K + k + l)/2 \rfloor$ and weight p and store them in a table at the address $H_1\ \boldsymbol{x}_1^T$
  for all binary vectors $\boldsymbol{x}_2$ of length $\lceil (K + k + l)/2 \rceil$ and weight p **do**
    for all $\boldsymbol{x}_1$ stored at the address $H_2\ \boldsymbol{x}_2^T$ **do**
      Compute $\boldsymbol{x}_3 \stackrel{\text{def}}{=} (\boldsymbol{x}_1 || \boldsymbol{x}_2) \boldsymbol{H}_3^T$ and form the codeword $\boldsymbol{x} \stackrel{\text{def}}{=} (\boldsymbol{x}_1 || \boldsymbol{x}_2 || \boldsymbol{x}_3)$ of $\mathscr{C}_{\text{pub}}$
      if $t_1 \leqslant \mathsf{wt}(\boldsymbol{x}_{1N}) \leqslant t_2$ **then**
        $\mathcal{L} \leftarrow \mathcal{L} \cup \{\boldsymbol{x}\}$
      end if
    end for
  end for
end for
return $\mathcal{L}$

3.3 Explaining the success of the attack

It turns out that this attack works extremely well on all the parameter choices made in the literature, and this even without knowing a single message-signature pair which would make life much easier for the attacker as demonstrated in [COV07]. In a first pass, the attack recovers easily message-signature pairs for all the parameters suggested in [BMJ11,KKS97,KKS05]. Once a signature-message pair is obtained, it can be exploited to bootstrap an attack that recovers the private key as we will explain later.

The reason why the attack works much better here than for general linear codes comes from the fact that $\hat{\boldsymbol{H}}$ *does not behave like a random matrix at all even if the two chosen matrices for the scheme, namely* $\boldsymbol{H}$ *and* $\boldsymbol{G}$ *are chosen at random.* The left part and the right part $\boldsymbol{H}$ and $\boldsymbol{F}$ are namely related by the equation:

$$\boldsymbol{F} = \boldsymbol{H}_J \boldsymbol{G}^T.$$

Indeed, the parity-check matrix $\hat{\boldsymbol{H}}$ displays peculiar properties: $\mathscr{C}_{\text{pub}}$ contains $\mathscr{C}_{\text{sec}}$ as a subcode and its codewords represent valid message-signature pairs. This subcode has actually a very specific structure that helps greatly the attacker:

1. There are many codewords in $\mathscr{C}_{\text{sec}}$, namely 2^k.
2. The support of these codewords is included in a fixed (and rather small) set of size $k + n$.
3. k positions of this set are known to the attacker.
4. These codewords form a linear code (of dimension k).

Because of all these properties, the aforementioned attack will work much better than should be expected from a random code. More precisely, let us bring in:

$$I' \stackrel{\text{def}}{=} I \cap J.$$

Notice that the expectation $\mathbb{E}\left\{I'\right\}$ of the cardinality of the set I' is equal to:

$$\mathbb{E}\left\{|I'|\right\} = \frac{n}{N}(k+K+l) = (R+\alpha\rho+\lambda)n \tag{3}$$

where we introduced the following notation:

$$R \stackrel{\text{def}}{=} \frac{K}{N}, \quad \rho \stackrel{\text{def}}{=} \frac{k}{n}, \quad \alpha \stackrel{\text{def}}{=} \frac{n}{N} \quad \text{and} \quad \lambda \stackrel{\text{def}}{=} \frac{l}{N}.$$

The point is that whenever there is a codeword $\boldsymbol{c}$ in $\mathscr{C}_{\text{sec}}$ which is such that $\mathsf{wt}(\boldsymbol{c}_{I'}) = 2p$ we have a non-negligible chance to find it with Algorithm 1. This does not hold with certainty because the algorithm does not examine all codewords $\boldsymbol{x}$ such that $\mathsf{wt}(\boldsymbol{x}_I) = 2p$, but rather it consists in splitting I in I_1 and I_2 of the same size and looking for codewords $\boldsymbol{x}$ such that $\mathsf{wt}(\boldsymbol{x}_{I_1}) = \mathsf{wt}(\boldsymbol{x}_{I_2}) = p$. In other words, we consider only a fraction δ of such codewords where:

$$\delta = \frac{\binom{(K+k+l)/2}{p}\binom{(K+k+l)/2}{p}}{\binom{K+k+l}{2p}} \approx \sqrt{\frac{(K+k+l)}{\pi p(K+k+l-2p)}}.$$

We will therefore obtain all codewords $\boldsymbol{c}$ in $\mathscr{C}_{\text{sec}}$ which are such that $\mathsf{wt}(\boldsymbol{c}_{I_1}) = \mathsf{wt}(\boldsymbol{c}_{I_2}) = p$. Consider now the restriction $\mathscr{C}'_{\text{sec}}$ of $\mathscr{C}_{\text{sec}}$ to the positions belonging to I', that is:

$$\mathscr{C}'_{\text{sec}} = \Big\{(\boldsymbol{x}_i)_{i\in I'} \ : \ \boldsymbol{x} = (\boldsymbol{x}_i)_{i\in 1N+k} \in \mathscr{C}_{\text{sec}}\Big\}. \tag{4}$$

The crucial issue is now the following question:

Does there exist in $\mathscr{C}'_{sec}$ a codeword of weight $2p$?

The reason for this is explained by the following proposition.

Proposition 1. *Let $I'_s \stackrel{def}{=} I_s \cap J$ for $s \in \{1,2\}$. If there exists a codeword $\boldsymbol{x}'$ in $\mathscr{C}'_{sec}$ such that $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$, then it will be the restriction of a codeword $\boldsymbol{x}$ in $\mathscr{C}_{sec}$ which will belong to the list $\mathcal{L}$ output by Algorithm 1.*

Proof. Consider a codeword $\boldsymbol{x}'$ in $\mathscr{C}'_{\text{sec}}$ such that $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$. For $s \in \{1,2\}$, extend $\boldsymbol{x}_{I'_s}$ with zeros on the other positions of I_s and let $\boldsymbol{x}_s$ be the corresponding word. Notice that $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ will be considered by Algorithm 1 and $\boldsymbol{x}_1$ will be stored at the address $\boldsymbol{H}_1\boldsymbol{x}_1^T$. By definition of $\boldsymbol{x}'$, $(\boldsymbol{x}_1||\boldsymbol{x}_2)$ is the restriction of a codeword $\boldsymbol{x}$ of $\mathscr{C}_{\text{sec}}$ to I, say $\boldsymbol{x} = (\boldsymbol{x}_1||\boldsymbol{x}_2||\boldsymbol{y})$ with $\boldsymbol{y} \in \mathbb{F}_2^{N-K-l}$. Since $\mathscr{C}_{\text{sec}} \subset \mathscr{C}_{\text{pub}}$ we have $\hat{\boldsymbol{H}}\boldsymbol{x}^T = 0$. Let $\hat{\boldsymbol{H}}'$ be the matrix obtained from $\hat{\boldsymbol{H}}$ put in quasi-systematic form through a Gaussian elimination as given in Figure 2. We also have $\hat{\boldsymbol{H}}'\boldsymbol{x}^T = 0$ and hence:

$$\boldsymbol{H}_1\boldsymbol{x}_1^T + \boldsymbol{H}_2\boldsymbol{x}_2^T = 0 \tag{5}$$

and

$$\boldsymbol{H}_3(\boldsymbol{x}_1||\boldsymbol{x}_2)^T + \boldsymbol{y}^T = 0. \tag{6}$$

Equation (5) shows that $\boldsymbol{x}_1$ is stored at address $\boldsymbol{H}_2\boldsymbol{x}_2^T$ and will be considered at Step 8 of the algorithm. In this case, $\boldsymbol{x}$ will be stored in $\mathcal{L}$. □

We expect that the dimension of $\mathscr{C}'_{\text{sec}}$ is still k and that this code behaves like a random code of the same length and dimension. Ignoring the unessential issue whether or not $\boldsymbol{x}'$ satisfies $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$, let us just assume that there exists $\boldsymbol{x}'$ in $\mathscr{C}'_{\text{sec}}$ such that $|\boldsymbol{x}'| = 2p$. There is a non negligible chance that we have $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$ and that this codeword will be found by our algorithm. The issue is therefore whether or not there is a codeword of weight $2p$ in a random code of dimension k and length I'. This holds with a good chance (see [**?**] for instance) as soon as:

$$2p \geqslant d_{\text{GV}}(I',k) \tag{7}$$

where $d_{\mathrm{GV}}(I', k)$ denotes the Gilbert-Varshamov distance of a code of length I' and dimension k. Recall that [?]:

$$d_{\mathrm{GV}}(I', k) \approx h^{-1}(1 - k/I')\, I'$$

where $h^{-1}(x)$ is the inverse function defined over $[0, \frac{1}{2}]$ of the binary entropy function $h(x) \stackrel{\mathrm{def}}{=} -x\log_2 x - (1-x)\log_2(1-x)$. Recall that we expect to have:

$$I' \approx (R + \alpha\rho + \lambda)n,$$

which implies

$$\frac{k}{I'} \approx \frac{\rho}{R + \alpha\rho + \lambda} \approx \frac{\rho}{R}$$

when α and λ are small. Roughly speaking, to avoid such an attack, several conditions have to be met:

1. ρ has to be significantly smaller than R,
2. n has to be large enough.

This phenomenon was clearly not taken into account in the parameters suggested in [KKS97,KKS05,BMJ11] as shown in Table 1. The values of $d_{\mathrm{GV}}(I', k)$ are extremely low (in the range $1-6$). In other words, taking $p = 1$ is already quite threatening for all these schemes. For the first parameter set, namely $(k, n, K, N) = (60, 1023, 192, 3000)$, this suggests to take $p = 3$. Actually taking $p = 1$ is already enough to break the scheme. The problem with these low values of p comes from the dependency of the complexity in p as detailed in the following section. For instance as long as p is smaller than 3 the complexity of one iteration is dominated by the Gaussian elimination Step 2.

Finally, let us observe that when this attack gives a message/signature pair, it can be used as a bootstrap for an attack that recovers the whole private key as will be explained in the following subsection.

Table 1. KKS Parameters with the corresponding value of $d_{\mathrm{GV}}(n', k)$.

Article	ρ	n	l	$n' \stackrel{\mathrm{def}}{=} \mathbb{E}\{I'\}$	R	N	$d_{\mathrm{GV}}(n', k)$
[KKS97]	$\frac{60}{1023} \approx 0.059$	1,023	8	89	$\frac{192}{3000} \approx 0.064$	3,000	6
[KKS05]	$\frac{48}{255} \approx 0.188$	255	8	65	$\frac{273}{1200} \approx 0.228$	1,200	5
[KKS97]	$\frac{48}{180} \approx 0.267$	180	8	64	$\frac{335}{1100} \approx 0.305$	1,100	4
[BMJ11]	1/2	320	12	165	1/2	11,626	1
[BMJ11]	1/2	448	13	230	1/2	16,294	1
[BMJ11]	1/2	512	13	264	1/2	18,586	1
[BMJ11]	1/2	768	13	395	1/2	27,994	2
[BMJ11]	1/2	1,024	14	527	1/2	37,274	2

3.4 Exploiting a signature for extracting the private key

If a signature σ of a message $\boldsymbol{x}$ is known, then $\boldsymbol{y} \stackrel{\mathrm{def}}{=} (\sigma, \boldsymbol{x})$ is a codeword of $\mathscr{C}_{\mathrm{sec}}$ which has weight about $n/2$ when restricted to its N first positions. This yields almost half of the positions of J. This can be exploited as follows. We perform the same attack as in the previous subsection, but we avoid choosing positions i for which $\sigma_i = 1$. More precisely, if we let $J_\sigma \stackrel{\mathrm{def}}{=} \mathsf{supp}(\sigma) = \{i : \sigma_i = 1\}$, then we choose $K + k + l$ positions among $1N \setminus J_\sigma$ to form I. The point of this choice is that we have more chances to have a smaller size for $I' = I \cap J$. Let $n' \stackrel{\mathrm{def}}{=} I'$, we have now:

$$\mathbb{E}\{n' \,|J_\sigma\} = \frac{n - |J_\sigma|}{N - |J_\sigma|}(k + K + l) \tag{8}$$

$$\mathbb{E}\{I'\} = \mathbb{E}\{\mathbb{E}\{n' \,|J_\sigma\}\} \approx \frac{n/2}{(N-n/2)}(k+K+l). \tag{9}$$

The last approximation follows from the fact that the weight $\mathsf{wt}(\sigma)$ is quite concentrated around $n/2$. The same reasoning can be made as before, but the odds that the algorithm finds other valid signatures are much higher. This comes from the fact that the expectation I' is half the expected size of I' in the previous case as given in Equation (3). Previously we had $\mathbb{E}\left\{\frac{I'}{k}\right\} \approx \frac{R}{\rho}$, whereas now we have:

$$\mathbb{E}\left\{\frac{I'}{k}\right\} \approx \frac{R}{2\rho}.$$

In other words, in order to avoid the previous attack we had to take ρ significantly smaller than R and now, we have to take ρ significantly smaller than $R/2$. For all the parameters proposed in the past, it turns out that $d_{\mathrm{GV}}(I',k)$ is almost always equal to 1, which makes the attack generally successful in just one iteration by choosing $p=1$.

Moreover, if another valid signature σ' is obtained and by taking the union $J_\sigma \cup J_{\sigma'}$ of the supports, then about 3/4 of the positions of J will be revealed. We can start again the process of finding other message/signature pairs by choosing $K+k+l$ positions among $\{1,2,\ldots,N\} \setminus (J_\sigma \cup J_{\sigma'})$ to form the sets I. This approach can be iterated as explained in Algorithm **??**. This process will quickly reveal the whole set J and from this, the private key is easily extracted as detailed in [COV07].

Algorithm 2 Recovering the private key from $t \geqslant 1$ signatures.

PARAMETERS:

r : number of iterations
l : small integer ($l \leqslant 40$)
p : very small integer ($1 \leqslant p \leqslant 4$).

INPUT:

$\hat{\boldsymbol{H}}$: public matrix as defined in Figure **??**
$\{\sigma_1,\ldots,\sigma_t\}$: list of $t \geqslant 1$ valid signatures

OUTPUT: $J \subset 1N$ of cardinality n

```
J ← ∪_{i=1}^t supp(σ_i)
repeat
    S ← 1N \ J
    L ← KKSforge(r,l,p,S,Ĥ)
    for all σ ∈ L do
        J ← J ∪ supp(σ)
    end for
until J = n
return J
```

Finally, let us focus on the variant proposed in [BMJ11]. In this case, we have slightly less information than in the original KKS scheme. This can be explained by the following reasoning. In this case too, we choose S again as $[1\cdots N]\setminus J_\sigma$, where as before J_σ is defined as $J_\sigma \stackrel{\text{def}}{=} \{i : \sigma_i = 1\}$. However this time, by defining n' again as $n' \stackrel{\text{def}}{=} I'$, we have

$$\mathbb{E}\{n' \,|J_\sigma\} = \frac{|J'_\sigma|}{N-|J_\sigma|}(k+K+l)$$

where

$$J'_\sigma = J \setminus J_\sigma.$$

However, this time due to the noise which is added, $|J_\sigma|$ is expected to be larger than before (namely of order $\frac{n}{2} + \frac{(N-n)n}{N}$).

4 Analysis of the Attack

The purpose of this section is to provide a very crude upper-bound on the complexity of the attack. We assume here that the code $\mathscr{C}_{\text{rand}}$ of length n which is equal to the restriction on J of $\mathscr{C}_{\text{sec}}$:

$$\mathscr{C}_{\text{rand}} \stackrel{\text{def}}{=} \left\{ (x_j)_{j \in J} \; : \; \boldsymbol{x} = (x_1, \dots, x_{N+k}) \in \mathscr{C}_{\text{sec}} \right\}$$

behaves like a random code. More precisely we assume that it has been chosen by picking a random parity-check matrix $\boldsymbol{H}_{\text{rand}}$ of size $(n-k) \times n$ (by choosing its entries uniformly at random among $\mathbb{F}_2$). This specifies a code $\mathscr{C}_{\text{rand}}$ of length n as $\mathscr{C}_{\text{rand}} = \{\boldsymbol{x} \in \mathbb{F}_2^n : \boldsymbol{H}_{\text{rand}} \boldsymbol{x}^T = 0\}$. We first give in the following section some quite helpful lemmas about codes of this kind.

4.1 Preliminaries about random codes

We are interested in this section in obtaining a lower bound on the probability that a certain subset X of $\mathbb{F}_2^n$ has a non empty intersection with $\mathscr{C}_{\text{rand}}$. For this purpose, we first calculate the two following probabilities. The probabilities are taken here over the random choices of $\boldsymbol{H}_{\text{rand}}$.

Lemma 1. *Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two different and nonzero elements of $\mathbb{F}_2^n$. Then*

$$\mathbf{prob}(\boldsymbol{x} \in \mathscr{C}_{rand}) = 2^{k-n} \tag{10}$$

$$\mathbf{prob}(\boldsymbol{x} \in \mathscr{C}_{rand}, \boldsymbol{y} \in \mathscr{C}_{rand}) = 2^{2(k-n)} \tag{11}$$

To prove this lemma, we will introduce the following notation and lemma. For $\boldsymbol{x} = (x_i)_{1 \leqslant i \leqslant s}$ and $\boldsymbol{y} = (y_i)_{1 \leqslant i \leqslant s}$ being two elements of $\mathbb{F}_2^s$ for some arbitrary s, we define $\boldsymbol{x} \cdot \boldsymbol{y}$ as

$$\boldsymbol{x} \cdot \boldsymbol{y} = \sum_{1 \leqslant i \leqslant s} x_i y_i,$$

the addition being performed over $\mathbb{F}_2$.

Lemma 2. *Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two different and nonzero elements of $\mathbb{F}_2^n$ and choose $\boldsymbol{h}$ uniformly at random in $\mathbb{F}_2^n$, then*

$$\mathbf{prob}(\boldsymbol{x} \cdot \boldsymbol{h} = 0) = \frac{1}{2} \tag{12}$$

$$\mathbf{prob}(\boldsymbol{x} \cdot \boldsymbol{h} = 0, \boldsymbol{y} \cdot \boldsymbol{h} = 0) = \frac{1}{4} \tag{13}$$

Proof. To prove Equation (12) we just notice that the subspace $\{\boldsymbol{h} \in \mathbb{F}_2^n : \boldsymbol{x} \cdot \boldsymbol{h} = 0\}$ is of dimension $n-1$. There are therefore 2^{n-1} solutions to this equation and

$$\mathbf{prob}(\boldsymbol{x} \cdot \boldsymbol{h} = 0) = \frac{2^{n-1}}{2^n} = \frac{1}{2}.$$

On the other hand, the hypothesis made on $\boldsymbol{x}$ and $\boldsymbol{y}$ implies that $\boldsymbol{x}$ and $\boldsymbol{y}$ generate a subspace of dimension 2 in $\mathbb{F}_2^n$ and that the dual space, that is $\{\boldsymbol{h} \in \mathbb{F}_2^n : \boldsymbol{x} \cdot \boldsymbol{h} = 0, \boldsymbol{y} \cdot \boldsymbol{h} = 0\}$ is of dimension $n-2$. Therefore

$$\mathbf{prob}(\boldsymbol{x} \cdot \boldsymbol{h} = 0, \boldsymbol{y} \cdot \boldsymbol{h} = 0) = \frac{2^{n-2}}{2^n} = \frac{1}{4}$$

$\square$

Proof (of Lemma 1). Let $\boldsymbol{h}_1, \dots, \boldsymbol{h}_{n-k}$ be the $n-k$ rows of $\boldsymbol{H}_{\text{rand}}$. Then

$$\begin{aligned}
\mathbf{prob}(\boldsymbol{x} \in \mathscr{C}_{\text{rand}}) &= \mathbf{prob}(\boldsymbol{H}_{\text{rand}} \boldsymbol{x}^T = 0) \\
&= \mathbf{prob}(\boldsymbol{h}_1 \cdot \boldsymbol{x} = 0, \dots, \boldsymbol{h}_{n-k} \cdot \boldsymbol{x} = 0) \\
&= \mathbf{prob}(\boldsymbol{h}_1 \cdot \boldsymbol{x} = 0) \dots \mathbf{prob}(\boldsymbol{h}_{n-k} \cdot \boldsymbol{x} = 0) && (14) \\
&= 2^{k-n} && (15)
\end{aligned}$$

where Equation (14) follows by the independence of the events and Equation (15) uses Lemma 2. Equation (11) is obtained in a similar fashion. □

Lemma 3. *Let X be some subset of $\mathbb{F}_2^n$ of size m and let f be the function defined by $f(x) \stackrel{def}{=} \max\left(x(1-x/2), 1-\frac{1}{x}\right)$. We denote by x the quantity $\frac{m}{2^{n-k}}$, then*

$$\mathbf{prob}(X \cap \mathscr{C}_{rand} \neq \emptyset) \geq f(x).$$

Proof. For $\boldsymbol{x}$ in X we define $E_{\boldsymbol{x}}$ as the event "$\boldsymbol{x}$ belongs to $\mathscr{C}_{\text{rand}}$" and we let $q \stackrel{\text{def}}{=} 2^{k-n}$. We first notice that

$$\mathbf{prob}(X \cap \mathscr{C}_{\text{rand}} \neq \emptyset) = \mathbf{prob}\left(\bigcup_{\boldsymbol{x} \in X} E_{\boldsymbol{x}}\right).$$

By using the Bonferroni inequality [Com74, p. 193] on the probability of the union of events we obtain

$$\begin{aligned}
\mathbf{prob}\left(\bigcup_{\boldsymbol{x} \in X} E_{\boldsymbol{x}}\right) &\geq \sum_{\boldsymbol{x} \in X} \mathbf{prob}(E_{\boldsymbol{x}}) - \sum_{\{\boldsymbol{x},\boldsymbol{y}\} \subset X} \mathbf{prob}(E_{\boldsymbol{x}} \cap E_{\boldsymbol{y}}) && (16) \\
&\geq mq - \frac{m(m-1)}{2} q^2 && (17) \\
&\geq mq - \frac{m^2 q^2}{2} \\
&\geq mq(1 - mq/2),
\end{aligned}$$

where (17) follows from Lemma 1. This bound is rather sharp for small values of mq. On the other hand for larger values of mq, another lower bound on $\mathbf{prob}(X \cap \mathscr{C}_{\text{rand}} \neq \emptyset)$ is more suitable [dC97]. It gives

$$\begin{aligned}
\mathbf{prob}\left(\bigcup_{\boldsymbol{x} \in X} E_{\boldsymbol{x}}\right) &\geq \sum_{\boldsymbol{x} \in X} \frac{\mathbf{prob}(E_{\boldsymbol{x}})^2}{\sum_{\boldsymbol{y} \in X} \mathbf{prob}(E_{\boldsymbol{x}} \cap E_{\boldsymbol{y}})} && (18) \\
&\geq \frac{mq^2}{q + (m-1)q^2} && (19) \\
&\geq \frac{mq^2}{q + mq^2} && (20) \\
&\geq \frac{1}{1 + \frac{1}{mq}} \\
&\geq 1 - \frac{1}{mq},
\end{aligned}$$

By taking the maximum of both lower bounds, we obtain our lemma. □

4.2 Estimating the complexity of Algorithm 1

Here we estimate how many iterations have to be performed in order to break the scheme when no signature is known and when $S = [1 \cdots N]$. For this purpose, we start by lower-bounding the probability that an iteration is successful. Let us bring the following random variables for $i \in \{1, 2\}$:

$$I_i' \stackrel{\text{def}}{=} I_i \cap J \quad \text{and} \quad W_i \stackrel{\text{def}}{=} |I_i'|\,.$$

By using Lemma 1, we know that an iteration finds a valid signature when there is an $\boldsymbol{x}$ in $\mathscr{C}_{\text{sec}}$ such that

$$|\boldsymbol{x}_{I_1'}| = |\boldsymbol{x}_{I_2'}| = p.$$

Therefore the probability of success P_{succ} is lower bounded by

$$P_{\text{succ}} \geq \sum_{w_1, w_2 : w_1 + w_2 \leqslant n} \mathbf{prob}(W_1 = w_1, W_2 = w_2)\mathbf{prob}\left\{\exists \boldsymbol{x} \in \mathscr{C}_{\text{sec}} : |\boldsymbol{x}_{I_1'}| = \left|\boldsymbol{x}_{I_2'}\right| = p | W_1 = w_1, W_2 = w_2\right\} \tag{21}$$

On the other hand, by using Lemma 3 with the set

$$X \stackrel{\text{def}}{=} \left\{\boldsymbol{x} = (x_j)_{j \in J} : |\boldsymbol{x}_{I_1'}| = \left|\boldsymbol{x}_{I_2'}\right| = p\right\}$$

which is of size $\binom{w_1}{p}\binom{w_2}{p}2^{n - w_1 - w_2}$, we obtain

$$\mathbf{prob}\left\{\exists \boldsymbol{x} \in \mathscr{C}_{\text{sec}} : |\boldsymbol{x}_{I_1'}| = \left|\boldsymbol{x}_{I_2'}\right| = p | W_1 = w_1, W_2 = w_2\right\} \geq f(x). \tag{22}$$

with

$$x \stackrel{\text{def}}{=} \frac{\binom{w_1}{p}\binom{w_2}{p}2^{n - w_1 - w_2}}{2^{n-k}} = \binom{w_1}{p}\binom{w_2}{p}2^{k - w_1 - w_2}$$

The first quantity is clearly equal to

$$\mathbf{prob}(W_1 = w_1, W_2 = w_2) = \frac{\binom{n}{w_1}\binom{n - w_1}{w_2}\binom{N-n}{(K+k+l)/2 - w_1}\binom{N - n - (K+k+l)/2 + w_1}{(K+k+l)/2 - w_2}}{\binom{N}{(K+k+l)/2}\binom{N - (K+k+l)/2}{(K+k+l)/2}}. \tag{23}$$

Plugging in the expressions obtained in (**??**) and (**??**) in (**??**) we have an explicit expression of a lower bound on P_{succ}. The number of iterations for our attack to be successful is estimated to be of order $\frac{1}{P_{\text{succ}}}$. We obtain therefore an upper-bound on the expected number of iterations, what we denote by `UpperBound`. Table **??** shows for various KKS parameters, p and l the expected number of iterations.

Table 2. KKS Parameters with the corresponding value of $\frac{1}{P_{\text{succ}}}$.

Article	ρ	n	l	p	$n' \stackrel{\text{def}}{=} \mathbb{E}\{\lvert I'\rvert\}$	R	N	`UpperBound`
[KKS97]	$\frac{60}{1023} \approx 0.059$	1,023	8	1	91	$\frac{192}{3000} \approx 0.064$	3,000	111.26
	$\frac{60}{1023} \approx 0.059$	1,023	14	2	91	$\frac{192}{3000} \approx 0.064$	3,000	14.17
[KKS05]	$\frac{48}{255} \approx 0.188$	255	8	1	66	$\frac{273}{1200} \approx 0.228$	1,200	26.41
	$\frac{48}{255} \approx 0.188$	255	14	2	66	$\frac{273}{1200} \approx 0.228$	1,200	4.37
[KKS97]	$\frac{48}{180} \approx 0.267$	180	8	1	65	$\frac{335}{1100} \approx 0.305$	1,100	6.07
	$\frac{48}{180} \approx 0.267$	180	15	2	65	$\frac{335}{1100} \approx 0.305$	1,100	1.82
[BMJ11]	1/2	320	12	1	165	1/2	11,626	1.24
[BMJ11]	1/2	448	13	1	230	1/2	16,294	1.34
[BMJ11]	1/2	512	13	1	264	1/2	18,586	1.39
[BMJ11]	1/2	768	13	1	395	1/2	27,994	1.61
[BMJ11]	1/2	1,024	14	1	527	1/2	37,274	1.85

4.3 Number of operations of one iteration

The complexity of one iteration of Algorithm 1 is $C(p,l) = C_{\text{Gauss}} + C_{\text{hash}} + C_{\text{check}}$ where C_{Gauss} is the complexity of a Gaussian elimination, C_{hash} is the complexity of hashing all the $\boldsymbol{x}_1$'s and C_{check} is the complexity of checking all the $\boldsymbol{x}_2$'s with the following expressions:

$$C_{\text{Gauss}} = O\Big((N+k)(N-k)(N-k-l)\Big) = O(N^3) \tag{24}$$

$$C_{\text{hash}} = O\left(\binom{(K+k+l)/2}{p}\right) \tag{25}$$

$$C_{\text{check}} = O\left(\frac{1}{2^l}(N-K-l)^2\binom{(K+k+l)/2}{p}^2\right) \tag{26}$$

The last expression giving C_{check} comes from the fact that the algorithm considers $\binom{(K+k+l)/2}{p}$ elements $\boldsymbol{x}_2$, and for each of these candidates, we check about $O\left(\frac{1}{2^l}\binom{(K+k+l)/2}{p}\right)$ elements $\boldsymbol{x}_1$'s, which involves a matrix multiplication in Step 9. Let us note that l will be chosen such that C_{hash} and C_{check} are roughly of the same order, say $2^l \approx \binom{(K+k+l)/2}{p}$.

Table 3. KKS Parameters.

Scheme	Version	k	n	t_1	t_2	$N-K$	N	Security
KKS-2	[KKS97]	60	1,023	352	672	2,808	3,000	36
	[KKS05]	48	255	48	208	927	1,200	
KKS-3	[KKS97]	60	280	50	230	990	1,250	17
	[COV07]	160	1,000	90	110	1,100	2,000	80
KKS-4	[KKS97]	48	180	96	96	765	1,100	53

Table 4. Parameters proposed in [BMJ11]

k	n	t_1	t_2	$N-K$	N	Security
160	320	100	177	5,813	11,626	80
224	448	133	243	8,147	16,294	112
256	512	149	275	9,293	18,586	128
384	768	214	405	13,997	27,994	192
512	1,024	278	535	18,637	37,274	256

Table 5. Parameters proposed in [BMJ11] with $t_1 = n/2 - 3\sqrt{n}/2$ and $t_2 = n/2 + 3\sqrt{n}/2$.

k	n	t_1	t_2	$N-K$	N	Security
160	320	133	187	5,813	11,626	80
224	448	192	256	8,147	16,294	112
256	512	222	290	9,293	18,586	128
384	768	342	426	13,997	27,994	192
512	1,024	464	560	18,637	37,274	256

References

[BMJ11] P. S.L.M. Barreto, R. Misoczki, and M. A. Simplicio Jr. One-time signature scheme from syndrome decoding over generic error-correcting codes. *Journal of Systems and Software*, 84(2):198 – 204, 2011.

[BMvT78] 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.

[Com74] L. Comtet. *Advanced Combinatorics.* Reidel, Dordrecht, 1974.

[COV07] P.L. Cayrel, A. Otmani, and D. Vergnaud. On Kabatianskii-Krouk-Smeets Signatures. In *Proceedings of the first International Workshop on the Arithmetic of Finite Fields (WAIFI 2007)*, Springer Verlag Lecture Notes, pages 237–251, Madrid, Spain, June 21–22 2007.

[dC97] D. de Caen. A lower bound on the probability of a union. *Discrete Mathematics*, 169:217–220, 1997.

[KKS97] G. Kabatianskii, E. Krouk, and B. Smeets. A digital signature scheme based on random error-correcting codes. In *Proceedings of the 6th IMA International Conference on Cryptography and Coding*, pages 161–167, London, UK, 1997. Springer-Verlag.

[KKS05] G. Kabatiansky, E. Krouk, and S. Semenov. *Error Correcting Coding and Security for Data Networks: Analysis of the Superchannel Concept.* John Wiley & Sons, 2005.