Cryptanalysis of a Code-Based Signature scheme based on the Schnorr-Lyubashevsky framework

Marco Baldi[1], Jean-Christophe Deneuville[2], Edoardo Persichetti[3], and Paolo Santini[1]

[1] {m.baldi,p.santini}@univpm.it, Marche Polytechnic University, Italy
[2] jean-christophe.deneuville@enac.fr, ENAC, University of Toulouse, France
[3] epersichetti@fau.edu, Florida Atlantic University, USA

Abstract. We propose an attack on the recent attempt by Li, Xing and Yeo to produce a code-based signature scheme using the Schnorr-Lyubashevsky approach in the Hamming metric, and verify its effectiveness through numerical simulations. Differently from other (unsuccessful) proposals, this new scheme exploits rejection sampling along with dense noise vectors to hide the secret key structure in produced signatures. We show that these measures, besides yielding very slow signing times and rather long signatures, do not succeed in protecting the secret key. We are indeed able to prove the existence of a strong correlation between produced signatures, which ultimately leaks information about the secret key. To support this claim, we use both theoretical arguments and numerical evidences. Finally, we employ such a weakness to mount a full key recovery attack, which is able to recover the secret key after the observation of a bunch of signatures. Our results show that the considered scheme may be secure only for one-time usage.

Keywords: Code-based cryptography, cryptanalysis, digital signature, zero-knowledge identification scheme.

1 Introduction

Public-key cryptography is a fundamental piece of modern day's communications. Indeed, public-key cryptosystems are invaluable tools to achieve confidentiality, authentication, non-repudiation, and several other tasks which are essential for secure communications. The vast majority of public-key primitives is currently based on "classical" problems from number theory, such as factoring (RSA) and computing discrete logarithms (Diffie-Hellman, El Gamal). However, due to Shor's seminal work [21], such systems will not be considered secure anymore, once a sufficiently large and stable quantum computer is built. In response, over the last few years the cryptographic community has been focusing on a coordinated effort to produce viable alternatives, based on hard problems which are not vulnerable to quantum cryptanalytic attacks. These include cryptosystems based on lattices, linear codes, multivariate equations, isogenies on elliptic curves, and others. Such an effort is led by NIST's call for Post-Quantum Standardization [17], which is currently nearing its end.

Code-based cryptographic schemes, which mostly rely on the hardness of decoding a random linear code [7, 6], are one of the major players in this scenario. However, while code-based encryption schemes, stemming from McEliece's seminal work [16], offer some very credible and strong candidates, the situation is not quite the same for code-based signature schemes. It is easy to evince this even just looking at the history of NIST's competition. In fact, code-based schemes, together with lattice-schemes, form the lion's share of all the candidates; yet, while code-based encryption schemes are varied, well-represented and made it to the third (and likely final) round with three promising candidates [2, 4, 1], only three candidates for code-based signature schemes were presented to the first round to begin with, and they have all since been either broken, or withdrawn.

The difficulty of producing secure and efficient code-based signatures is, in a sense, inherent to the setting. In fact, "hash-and-sign" schemes à la CFS [10] have to somehow deal with the fact that it is not straightforward to find a preimage (decode) with the required characteristics (low Hamming weight), given a random input; this results in very slow signing times, besides the large public keys which are typical of code-based cryptography in general. A new approach, that uses rejection sampling and ternary vectors of very high weight [11], still presents very large keys, despite the improvement in signing time and the very short signature size. Another approach to the design of code-based digital signature schemes is that of deriving them from zero-knowledge code-based identification schemes [24, 25, 9, 8] through the well-known Fiat-Shamir transformation. This generally results in a quite large signature size, which is a consequence of the multiple repetitions necessary to reach the desired (negligible) soundness error. In this scenario, it would be a boon to be able to devise a scheme following the Schnorr-Lyubashevsky approach [15], which has been extremely successful for the lattice case [13]; unfortunately, despite the similarities between the two settings, this has been an insurmountable task so far, at least for the Hamming metric. A first attempt was given in [18], where a negative result is presented, concluding that a straightforward adaptation (including a rank-metric version) was unlikely to succeed. A subsequent work [19], that used quasi-cyclic codes, was cryptanalyzed in [12, 20], despite offering only a one-time solution. New adaptations were proposed again [22] and again [23], always with the same negative outcome [5, 3].

In this work, we cryptanalyze the latest installment in this long series of unsuccessful attempts [14]. This new scheme, which we shorten to LXY (using the authors initials), incorporates a rejection sampling component, as in the original work of Lyubashevsky [15]. Furthermore, as another difference with its ancestors, the authors propose to use a denser noise vector to hide the secret key structure into a produced signature. In order to accommodate such choices, the parameters grow considerably, and the rejection rate is quite high, resulting in a scheme with very slow signing times and very long signatures. Unfortunately, it appears not even these extreme measures are enough to guarantee the security of the scheme. Indeed, we show that the considered rejection sampling does not guarantee indistinguishability of the signatures, since it only takes into account their weight and not their supports. Starting from this observation, we first describe why the signatures are correlated with the secret key, and then how the secret key can be fully recovered after a very limited number of observations. As a result, the scheme can be considered, at best, as one-time secure, and therefore, considering the performance aspects mentioned above, not interesting in practice.

The paper is organized as follows. In Section 2 we introduce the notation we use throughout the paper, and remind the LXY scheme. In Section 3 we explain why produced signatures leak information on the secret key, and in Section 4 we exploit this leakage to mount a successful key recovery attack.

2 Notation and background

In the rest of the paper, we denote with $\mathbb{F}_2$ the finite field with two elements. Vectors will be denoted with bold small letters, while capital bold letters will be used for matrices. For a vector $\mathbf{a}$, we indicate the Hamming weight as $\mathrm{wt}(\mathbf{a})$, and the support (*i.e.* the set of indices of non-null entries) as $\mathrm{Supp}(\mathbf{a})$.

Given two integers a, b, we denote with $[a; b]$ the range of integers from a to b. If a and b are, in general, reals, we use $[\![a; b]\!]$ to denote the set of integers from $\lfloor a \rfloor$ to $\lceil b \rceil$. For $n \in \mathbb{N}$, we denote as $\mathcal{R} = \mathbb{F}_2[x]/(x^n + 1)$ the ring of binary polynomials modulo $x^n + 1$, and represent each element of $\mathcal{R}$ as the corresponding vector of coefficients with entries over $\mathbb{F}_2$ and length n. We use Rot_j to denote the operator that applies a cyclic-shift of j positions; in other words, for $\mathbf{a} = (a_0, a_1, \cdots, a_{n-1})$, we have

$$\mathrm{Rot}_j(\mathbf{a}) = (a_j, a_{j+1}, \cdots, a_{n-1}, a_0, a_1, \cdots, a_{j-1}).$$

Finally, we use $\mathfrak{L}$ to denote the lifting of a vector from $\mathbb{F}_2$ to $\{0,1\} \subseteq \mathbb{Z}$.

We will use $\mathcal{B}_\tau^m$ to denote the distribution of length-m vectors where each entry follows a Bernoulli distribution with parameter τ (*i.e.* is set with probability τ and is null with probability $1-\tau$). The *Probability Distribution Function (PDF)* of the weight of such vectors is as follows

$$f_\tau^m(x) = \begin{cases} 0 & \text{if } x \notin [0; m], \\ \binom{m}{x}\tau^x(1-\tau)^{m-x} & \text{otherwise.} \end{cases}$$

Furthermore, we denote with $\widetilde{\mathcal{B}}_{\xi,\tau}^m$ the distribution of vectors sampled from $\mathcal{B}_\tau^m$ and whose weight is in $[\![m\tau - \xi; m\tau + \xi]\!]$. The PDF of the weight of such vectors follows the so-called truncated binomial distribution, given by

$$\widetilde{f}_{\xi,\tau}^m(x) = \begin{cases} 0 & \text{if } x \notin [\![m\tau - \xi, m\tau + \xi]\!], \\ \frac{f_\tau^m(x)}{\sum_{i=\lfloor m\tau-\xi \rfloor}^{\lceil m\tau+\xi \rceil} f_\tau^m(i)} & \text{otherwise.} \end{cases}$$

2.1 The LXY scheme

In [14, Section 4], the authors describe a general setting for their signature scheme, which already incorporates the rejection sampling in the signature generation. Such a construction is then specialized in [14, Section 6], using quasi-cyclicity and codes with rate $1/d$, where $d \geq 2$. Note that parameters are provided only for the case of $d = 2$ (*i.e.* code with rate $1/2$). Hence, we mainly focus on such a variant, and will refer to it as the LXY scheme; in Section 4.1 we will briefly comment on the security of the more general setting.

Algorithms 1, 2 and 3 recall the functioning of the LXY scheme. We have used H to denote a so-called *Weight Restricted Hash Function*, *i.e.* a hash function that outputs digests of length n and weight w. The rejection sampling in the signing algorithm is performed through the instructions in lines 6–9. The function $\varphi(\omega)$ is such that the PDF of the signature weight ω is tuned to be indistinguishable from the truncated binomial distribution $\widetilde{f}_{\xi,\tau}^{2n}$. For details about how $\varphi(\omega)$ is computed, we refer the interested reader to [14, Section 3].

Algorithm 1 KeyGen

```
Input: Public parameters $n, u \in \mathbb{N}$.
Output: $\mathbf{h} \in \mathcal{R}$
1: Choose $\mathbf{s}_1, \mathbf{s}_2 \in \mathcal{R}$ with weight $u$
2: $\mathbf{h} \leftarrow \mathbf{s}_2\mathbf{s}_1^{-1}$
3: $\mathsf{sk} \leftarrow (\mathbf{s}_1, \mathbf{s}_2)$,   $\mathsf{pk} \leftarrow \mathbf{h}$
4: return $\mathsf{pk}, \mathsf{sk}$
```

Algorithm 2 Sign

```
Input: pk = $\mathbf{h}$, sk = $(\mathbf{s}_1, \mathbf{s}_2)$, message $m$, WRHF $\mathsf{H}$, rejection function $\varphi$ and parameters $n, \xi \in \mathbb{N}$,
    $\tau \in \mathbb{R}$.
Output: Signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c}) \in \mathcal{R}^3$ on message $m$.
1: Sample $\mathbf{e}_1$, $\mathbf{e}_2$ from $\mathcal{B}_\tau^n$
2: $\mathbf{y} \leftarrow \mathbf{h}\mathbf{e}_1 + \mathbf{e}_2$
3: $\mathbf{c} \leftarrow \mathsf{H}(\mathbf{y}, m)$
4: $\mathbf{z}_1 \leftarrow \mathbf{c}\mathbf{s}_1 + \mathbf{e}_1$,  $\mathbf{z}_2 \leftarrow \mathbf{c}\mathbf{s}_2 + \mathbf{e}_2$
5: $\omega \leftarrow \mathrm{wt}(\mathbf{z}_1) + \mathrm{wt}(\mathbf{z}_2)$
6: if $\omega \in [\![2n\tau - \xi, 2n\tau + \xi]\!]$ then
7:     Output $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$ with probability $\varphi(\omega)$, else restart
8: else
9:     Restart
```

Algorithm 3 Verify

```
Input: pk = $\mathbf{h}$, message $m$, signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$, WRHF $\mathsf{H}$ and parameters $n, \xi \in \mathbb{N}$, $\tau \in \mathbb{R}$.
Output: Accept if $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$ is valid, Reject otherwise
1: $w \leftarrow \mathrm{wt}(\mathbf{z}_1) + \mathrm{wt}(\mathbf{z}_2)$
2: if $w \in [\![2n\tau - \xi, 2n\tau + \xi]\!]$ and $\mathsf{H}(\mathbf{h}\mathbf{z}_1 + \mathbf{z}_2, m) == \mathbf{c}$ then return Accept
3: else return Reject
```

It is easy to see that an honest signature always gets accepted. Indeed, rejection sampling guarantees that the weight of a signature is in the range $[\![2n\tau - \xi; 2n\tau + \xi]\!]$ and, for a valid signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$, the following holds

$$\mathbf{h}\mathbf{z}_1 + \mathbf{z}_2 = \mathbf{c}(\mathbf{h}\mathbf{s}_1 + \mathbf{s}_2) + \mathbf{h}\mathbf{e}_1 + \mathbf{e}_2 = \mathbf{h}\mathbf{e}_1 + \mathbf{e}_2 = \mathbf{y},$$

since $\mathbf{h}\mathbf{s}_1 = \mathbf{s}_2\mathbf{s}_1^{-1}\mathbf{s}_1 = \mathbf{s}_2$. So, the digest of $\mathbf{h}\mathbf{z}_1 + \mathbf{z}_2$ is identical to $\mathbf{c}$.

The parameter sets recommended in [14] are reported in Table 1, where λ denotes the claimed security level in bits.

Table 1. LXY instances proposed in [14]

λ	n	u	w	τ	ξ
80	66,467	49	6	0.23925	70
128	248,579	75	8	0.24305	135

The public key size corresponds to n bits, while the signature size is given by $2n + w\lceil\log_2(n)\rceil$ bits ($2n$ bits for $\mathbf{z}$, and additional $w\lceil\log_2(n)\rceil$ bits to represent the support of $\mathbf{c}$). Hence, the resulting public key and signature sizes are 8.31 kB and 16.63 kB, respectively, for the instance with 80-bit security, while they are 31.07 kB and 62.16 kB, respectively, for the instance with 128-bit security.

3 Information leakage

Signatures produced by the LXY scheme have essentially the same structure as those of Persichetti's one-time scheme [19]. Indeed, for both schemes, the signature is $\big(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c}\big)$, where $\mathbf{z}_i = \mathbf{c}\mathbf{s}_i + \mathbf{e}_i$ and $\mathbf{c}$, $\mathbf{s}_i$ and $\mathbf{e}_i$ are somehow sparse. The vector $\mathbf{e}_i$ acts as a noise term, and its role is that of hiding $\mathbf{c}\mathbf{s}_i$ into $\mathbf{z}_i$. Note that the support of each $\mathbf{z}_i$ is contained in

$$\mathrm{Supp}(\mathbf{e}_i) \cup \{j + \ell \mod n \mid j \in \mathrm{Supp}(\mathbf{c}),\ \ell \in \mathrm{Supp}(\mathbf{s}_i)\}.$$

In Persichetti's scheme, $\mathbf{s}_i$ and $\mathbf{e}_i$ are very sparse and their weights are close, so that most of the set entries in a signature come from the product $\mathbf{c}\mathbf{s}_i$. This gives the possibility of mounting attacks based on the correlation due to quasi-cyclicity. More precisely, an adversary can compute all vectors $\mathrm{Rot}_j(\mathbf{z}_i)$, for $j \in \mathrm{Supp}(\mathbf{c})$, and consider the positions of the entries that are set in a large number of such vectors: these positions belong to the support of $\mathbf{s}_i$ with high probability. As shown in [20], this procedure allows retrieving a significant amount of information about the secret key (the remaining portion can be recovered via Information Set Decoding).

The LXY scheme differs from Persichetti's scheme in two major aspects:

i. the weight of each $\mathbf{e}_i$ is much larger than that of both $\mathbf{s}_i$ and $\mathbf{c}\mathbf{s}_i$. Then, an overwhelming majority of the set entries in each $\mathbf{z}_i$ corresponds to those of $\mathbf{e}_i$;
ii. with the rejection sampling in the signing algorithm, the PDF of the weight of the signatures is tuned to the truncated binomial distribution $\widetilde{f}^{2n}_{\xi,\tau}$ (*i.e.* that of vectors sampled from $\widetilde{\mathcal{B}}^{2n}_{\xi,\tau}$).

Because of these two features, in [14, Section 7.2], the authors claim immunity against known attacks such as [12, 20]; technically, they affirm that, as an effect of rejection sampling, the produced signatures are indistinguishable from vectors that are randomly sampled according to $\widetilde{\mathcal{B}}^{2n}_{\xi,\tau}$. The security proof follows from this statement.

We are able to refute this claim and prove that the signatures in the LXY scheme leak information about the secret key. We show that $\mathbf{e}_i$ cancels a non trivial, but still not large enough, portion of $\mathbf{c}\mathbf{s}_i$: this opens up for the possibility of statistical attacks as in [20]. The reason lies in the fact that rejection sampling in the LXY scheme only takes into account the weights of $\mathbf{z}_1$ and $\mathbf{z}_2$, but not their supports. As a result, the support of each $\mathbf{z}_i$ contains a moderately large portion of the support of $\mathbf{c}\mathbf{s}_i$, in which the positions of set entries are correlated due to quasi-cyclicity. As in Persichetti's scheme, this bias allows to gather information about the secret key.

3.1 Measuring the information leakage

To keep the computation as simple as possible, we introduce some elementary and plausible simplifications:

A) we assume that the weight of each $\mathbf{z}_i$, which we denote with ω_i, follows a truncated binomial distribution $\widetilde{\mathcal{B}}^{n}_{\xi/2,\tau}$;
B) we assume that the product $\mathbf{c}\mathbf{s}_i$ always has maximum weight uw.

Note that, in principles, both $\mathbf{z}_1$ and $\mathbf{z}_2$ can have any weight in $[\![0; 2n\tau + \xi]\!]$. However, we expect their weight distribution to be rather concentrated around the average value (which is $n\tau$), so that very low or high weights appear with negligible probability. Hence, it seems natural to use the distribution $\widetilde{\mathcal{B}}^{n}_{\xi/2,\tau}$ for both ω_1 and ω_2. This guarantees that $\omega = \omega_1 + \omega_2$ is not outside the range $[\![2n\tau - \xi; 2n\tau + \xi]\!]$ and follows a binomial distribution with average value $2n\tau$. Assumption B comes from the fact that both $\mathbf{c}$ and $\mathbf{s}_i$ are extremely sparse: one expects that cancellations are very rare, and that their product has maximum weight with very high probability.

Let us now consider a signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$, and define

- ω_i' as the number of elements that are contained in both $\text{Supp}(\mathbf{cs}_i)$ and $\text{Supp}(\mathbf{e}_i)$;
- ω_i'' as the number of elements that are in $\text{Supp}(\mathbf{e}_i)$ but not in $\text{Supp}(\mathbf{cs}_i)$.

Note that ω_i' corresponds to the number of set entries of $\mathbf{cs}_i$ that are cancelled by $\mathbf{e}_i$, while ω_i'' corresponds to the amount of set entries in $\mathbf{z}_i$ which are due to $\mathbf{e}_i$. Furthermore, we have $\text{wt}(\mathbf{z}_i) = \text{wt}(\mathbf{cs}_i) - \omega_i' + \omega_i''$. Because of assumption B, we have $\text{wt}(\mathbf{cs}_i) = uw$, so that

$$\omega_i = uw - \omega_i' + \omega_i''.$$

When $\omega_i = a \in [\![n\tau - \xi/2; n\tau + \xi/2]\!]$, the probability that ω_i' has value $x \in [0; uw]$ is given by

$$\Pr[\omega_i' = x \mid \omega_i = a] = \frac{f_\tau^{uw}(x) f_\tau^{n-uw}(a + x - uw)}{\sum_{b=0}^{uw} f_\tau^{uw}(b) f_\tau^{n-uw}(a + b - uw)}.$$

Hence, summing over all possible ω_i, we obtain the expression of the PDF of ω_i', that is

$$\begin{aligned}\Pr[\omega_i' = x] &= \sum_{a=\lfloor n\tau-\xi/2\rfloor}^{\lceil n\tau+\xi/2\rceil} \Pr[\omega_i = a]\Pr[\omega_i' = x \mid \omega_i = a] \\ &= \sum_{a=\lfloor n\tau-\xi/2\rfloor}^{\lceil n\tau+\xi/2\rceil} \tilde{f}_{\xi/2,\tau}^{n}(a)\Pr[\omega_i' = x \mid \omega_i = a]. \qquad (1)\end{aligned}$$

A validation of the above formula is shown in Figure 1, where we compare the theoretical PDF of ω_i' with the one we have obtained empirically, by measuring the number of cancellations on a large number of produced signatures. As we see, the experimental PDF closely matches the theoretical one; this also provides evidence that assumptions A and B introduced earlier have no practical impact on our analysis.

Let ϵ be the ratio between the average value of ω_i' and the weight of $\mathbf{cs}_i$, that is

$$\epsilon = \frac{\sum_{j=0}^{uw} j \cdot \Pr[\omega' = j]}{uw}.$$

Note that ϵ corresponds to the fraction of entries of $\mathbf{cs}_i$ that are canceled by $\mathbf{e}_i$ and, by definition, $\epsilon \in [\![0; 1]\!]$: thus, we can use it as the probability that a set entry in $\mathbf{cs}_i$ will not be set in the final $\mathbf{z}_i$. For $j \in \text{Supp}(\mathbf{c})$, let $\mathbf{z}_i^{(j)} = \text{Rot}_j(\mathbf{z}_i)$; we have the following two situations:

- if $\ell \in \text{Supp}(\mathbf{s}_i)$, then the ℓ-th entry of $\mathbf{z}_i^{(j)}$ is set with probability

$$\rho' = 1 - \epsilon.$$

  In fact, it will be set unless $\mathbf{e}_i$ cancels it (*i.e.* $\mathbf{e}_i$ has a one in the same position);
- if $\ell \notin \text{Supp}(\mathbf{s}_i)$, then the ℓ-th entry of $\mathbf{z}_i^{(j)}$ is set with probability

$$\rho'' = \tau\frac{n - uw}{n} + (1 - \epsilon)\frac{uw}{n} = \tau(1 - uw/n) + (1 - \epsilon)uw/n.$$

  Since $uw \ll n$, we have $\rho'' \approx \tau$.

For the proposed parameters for the LXY scheme, the values of ρ' and ρ'' are significantly different:

- for the 80-bits parameters set, we have $\epsilon = 0.23831$, so that $\rho' = 0.7617$ and $\rho'' = 0.24156$;
- for the 128-bits parameters set, we have $\epsilon = 0.24252$, so that $\rho' = 0.75748$ and $\rho'' = 0.24769$.

The gap between ρ' and ρ'' allows to distinguish whether a position is in $\text{Supp}(\mathbf{cs}_i)$ or not. Starting from this observation, in the next section we present a full key-recovery attack (in a fashion similar to that proposed in [20]).

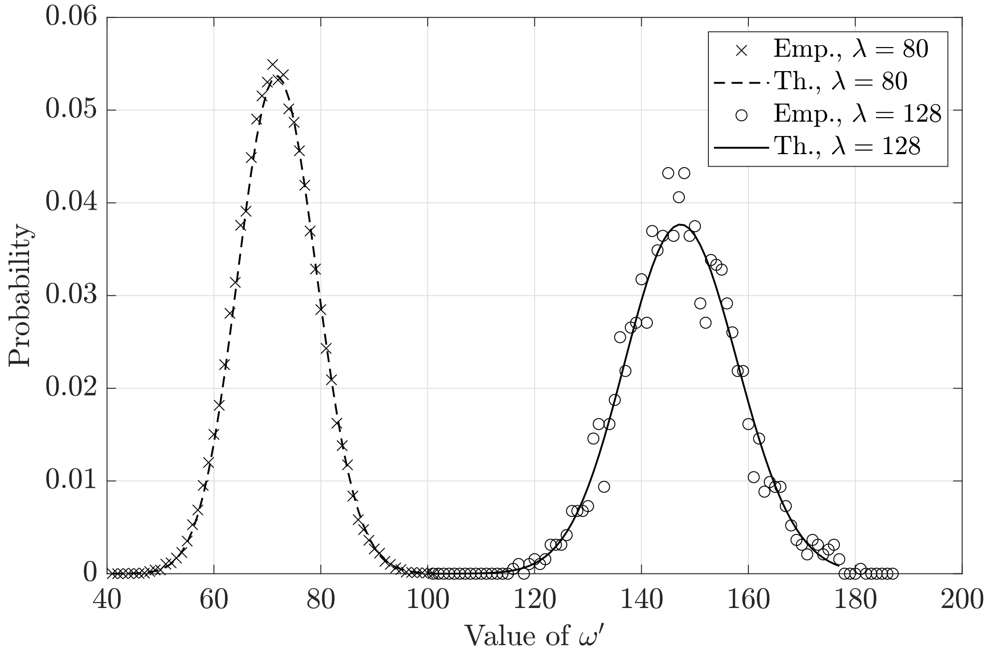

Fig. 1. Validation of (1) through numerical simulations. To empirically estimate the PDF, we have used approximately 10,000 signatures for the 80-bit parameters set, and 2,000 signatures for the 128-bit one.

4 Cryptanalysis results

Our key-recovery attack on the LXY scheme is reported in Algorithm 4. Basically, the procedure consists in collecting signatures and exploiting the aforementioned bias in the signature support to retrieve the secret $\mathbf{s}_1$ and $\mathbf{s}_2$. For each new collected signature, we produce a candidate for the secret key as $(\tilde{\mathbf{s}}_1, \tilde{\mathbf{s}}_2)$. To test it, we compute $\tilde{\mathbf{h}} = \tilde{\mathbf{s}}_1\tilde{\mathbf{s}}_2^{-1}$ and check if it is equal to the public key $\mathsf{pk} = \mathbf{h}$.

Algorithm 4 Key-recovery attack

```
Input: $u \in \mathbb{N}$, public key $\mathsf{pk} = \mathbf{h}$
Output: $\tilde{\mathbf{s}}_1, \tilde{\mathbf{s}}_2 \in \mathcal{R}$ with weight $u$, such that $\tilde{\mathbf{s}}_2\tilde{\mathbf{s}}_1^{-1} = \mathbf{h}$
 1: Set $\tilde{\mathbf{h}} \in \mathcal{R}$ as the null element
 2: Set $\mathbf{a}_1, \mathbf{a}_2 \in \mathbb{Z}^n$ as null vectors
 3: while $\tilde{\mathbf{h}} \neq \mathbf{h}$ do
 4:     Collect a new signature $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$
 5:     for $i \in \{1, 2\}$ do
 6:         for $j \in \mathrm{Supp}(\mathbf{c})$ do
 7:             $\mathbf{a}_i \leftarrow \mathbf{a}_i + \mathfrak{L}\big(\mathrm{Rot}_j(\mathbf{z}_i)\big)$
 8:     $S_i \leftarrow u$ positions with largest values in $\mathbf{a}_i$
 9:     $\tilde{\mathbf{s}}_i \leftarrow$ vector with support $S_i$
10:     $\tilde{\mathbf{h}} = \tilde{\mathbf{s}}_1\tilde{\mathbf{s}}_2^{-1}$
11: return $(\tilde{\mathbf{s}}_1, \tilde{\mathbf{s}}_2)$
```

It is clear that Algorithm 4 runs in polynomial time, and a rough estimate of its complexity is as follows

$$N\left(2wn + 2u + n^2 + n^3\right),$$

where N is the number of required signatures. Indeed, for each new collected signature, we update $\mathbf{a}_1$ and $\mathbf{a}_2$ with a cost of $2wn$ operations and create $\tilde{\mathbf{s}}_1$ and $\tilde{\mathbf{s}}_2$ with approximately $2u$ operations (we assume the cost is equal to the number of entries we set). Finally, to compute $\widetilde{\mathbf{h}}$, we use n^3 operations to invert $\tilde{\mathbf{s}}_2^{-1}$ and n^2 operations to multiply it by $\tilde{\mathbf{s}}_1$.

We note that, in principle, the algorithm may return a pair $(\tilde{\mathbf{s}}_1, \tilde{\mathbf{s}}_2)$ which is different from the actual secret key sk. This may happen only when *equivalent keys* exist, i.e., pairs $(\tilde{\mathbf{s}}_1, \tilde{\mathbf{s}}_1) \neq \mathsf{sk}$, with $\tilde{\mathbf{s}}_1$ and $\tilde{\mathbf{s}}_2$ having weight u and such that $\mathbf{h} = \tilde{\mathbf{s}}_1\tilde{\mathbf{s}}_2^{-1}$. Yet, any of such pairs can be used to construct signatures that will get accepted by the verification algorithm. Indeed, it is enough to run Algorithm 2, using $\tilde{\mathbf{s}}_1$ and $\tilde{\mathbf{s}}_1$ instead of $\mathbf{s}_1$ and $\mathbf{s}_2$ to obtain a valid signature.

We have implemented our algorithm[4], building upon the authors' implementation[5] of their signature scheme. Experimentally, the number N of signatures necessary to successfully perform a key recovery ranges from 4 to 9, for both parameter sets given in Table 1. Other results are reported in Table 2.

Table 2. Cryptanalysis results, sampling, keygen, signature and cryptanalysis timings in seconds, along with the average number N_{mean} of necessary signatures

λ	t_{samp}	t_{keygen}	t_{sign}	$t_{\text{cryptanalysis}}$	N_{mean}
80	82.47	1.34	108.48	17.07	6.76
128	626.38	5.77	1425.56	63.75	5.96

The results were obtain on Intel® Xeon® Gold 6230 @ 2.10GHz running SageMath version 9.0. It is worth noting that once the signature scheme is set and enough signatures have been collected, the cryptanalysis in itself is quite efficient.

4.1 Comments on possible variants

In [14], the authors also consider some possible variants for the scheme, although they do not recommend concrete parameters or provide any implementation. Yet, we are able to briefly comment about these possible generalizations.

Changing the code rate The LXY scheme may be instantiated with QC codes of rate $1/d$, with d being an integer ≥ 2 (see [14, Algorithms 5,6,7]); in such a case, the signature $\mathbf{z}$ is made of d polynomials. For $d = 2$ (the only case for which parameters are recommended), the scheme corresponds to the one we have analyzed in this paper. Notice that, for $d > 2$, signatures are constructed in exactly the same way (*i.e.* they are of the form $\mathbf{c}\mathbf{s}_i + \mathbf{e}_i$ and are the output of a rejection sampling involving analogous distributions): hence, we believe that the same security issues will arise.

Using unstructured codes In [14, Algorithms 1,2,3] the authors describe a general version of the scheme, based on unstructured codes. Again, they neither propose concrete parameters nor provide an implementation for this variant. Yet, we believe that also this version can be attacked

[4] Available at `https://github.com/deneuville/cryptanalysis_LXY`
[5] Available at `https://github.com/zhli271828/rand_code_sign`

with techniques similar to that in [3]. Indeed, in this case the signature is in the form $\mathbf{cS} + \mathbf{e}$, where $\mathbf{S}$ is a very sparse matrix, $\mathbf{c}$ is the very sparse public digest and $\mathbf{e}$ is a moderately sparse random vector. Basically, if we consider two signatures with digests sharing a common set entry (say, the one in position j), then the corresponding signatures will be both obtained by summing the j-th row of $\mathbf{S}$ to other sparse terms. The rejection sampling will not guarantee a sufficient number of cancellations in the term $\mathbf{cS}$, so that a large portion of its support will still appear in the output signature. Hence, there will still be correlation in the produced signatures, and collecting a sufficiently large number of signatures will still allow mounting a statistical attack to each row of $\mathbf{S}$. Thus, we argue that also this variant should be considered secure, at best, only for the one-time use.

5 Conclusion

In this paper we have cryptanalyzed a code-based signature scheme constructed upon the Schnorr-Lyubashevsky framework. This scheme, which has a construction that is very similar to that of Persichetti's one-time scheme, comes with a different parameter choice and an ad-hoc rejection sampling step in the signing algorithm. However, it suffers from analogous weaknesses of other broken schemes, and its secret key can be successfully recovered upon collection of a relatively small number of output signatures. Hence, according to our results, the scheme can be deemed secure only for the one-time usage case.

References

1. Aguilar Melchor, C., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Bos, J., Deneuville, J.C., Gaborit, P., Persichetti, E., Robert, J.M., Vron, P., Zmor, G.: HQC (Hamming Quasi-Cyclic). (2017)
2. Albrecht, M.R., Bernstein, D.J., Chou, T., Cid, C., Gilcher, J., Lange, T., Maram, V., von Maurich, I., Misoczki, R., Niederhagen, R., Paterson, K.G., Persichetti, E., Peters, C., Schwabe, P., Sendrier, N., Szefer, J., Tjhai, C.J., Tomlinson, M., Wang, W.: Classic McEliece: conservative code-based cryptography. (2017)
3. Aragon, N., Baldi, M., Deneuville, J.C., Khathuria, K., Persichetti, E., Santini, P.: Cryptanalysis of a code-based full-time signature (2020) submitted to DCC.
4. Aragon, N., Barreto, P.S.L.M., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Gueron, S., Güneysu, T., Melchor, C.A., Misoczki, R., Persichetti, E., Sendrier, N., Tillich, J.P., Vasseur, V., Zmor, G.: BIKE: Bit Flipping Key Encapsulation. (2017)
5. Aragon, N., Blazy, O., Deneuville, J.C., Gaborit, P., Lau, T.S.C., Tan, C.H., Xagawa, K.: Cryptanalysis of a rank-based signature with short public keys. Designs, Codes and Cryptography (2019) 1–11
6. Barg, S.: Some new NP-complete coding problems. Problemy Peredachi Informatsii **30**(3) (1994) 23–28
7. Berlekamp, E.R., McEliece, R.J., van Tilborg, H.C.A.: On the inherent intractability of certain coding problems (corresp.). IEEE Trans. Information Theory **24**(3) (1978) 384–386
8. Biasse, J.F., Micheli, G., Persichetti, E., Santini, P.: LESS is more: Code-based signatures without syndromes. In Nitaj, A., Youssef, A., eds.: Progress in Cryptology - AFRICACRYPT 2020, Cham, Springer International Publishing (2020) 45–65
9. Cayrel, P.L., Véron, P., El Yousfi Alaoui, S.M.: A zero-knowledge identification scheme based on the q-ary syndrome decoding problem. In: Selected Areas in Cryptography, Springer Berlin Heidelberg (2011) 171–186
10. Courtois, N., Finiasz, M., Sendrier, N.: How to achieve a McEliece-based digital signature scheme. In: ASIACRYPT. (2001) 157–174

11. Debris-Alazard, T., Sendrier, N., Tillich, J.P.: Wave: A new family of trapdoor one-way preimage sampleable functions based on codes. In: International Conference on the Theory and Application of Cryptology and Information Security, Springer (2019) 21–51
12. Deneuville, J.C., Gaborit, P.: Cryptanalysis of a code-based one-time signature. Designs, Codes and Cryptography **88**(9) (2020) 1857–1866
13. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Dilithium. (2017)
14. Li, Z., Xing, C., Yeo, S.L.: A new code based signature scheme without trapdoors. Cryptology ePrint Archive, Report 2020/1250 (2020)
15. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer (2012) 738–755
16. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. DSN Progress **42-44** (1978) 114–116
17. National Institute of Standards and Technology: NIST Post-Quantum Standardization process (2017) `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography`.
18. Persichetti, E.: Improving the Efficiency of Code-Based Cryptography. PhD thesis, Department of Mathematics, University of Auckland (2012)
19. Persichetti, E.: Efficient one-time signatures from quasi-cyclic codes: A full treatment. Cryptography **2** (10 2018) 30
20. Santini, P., Baldi, M., Chiaraluce, F.: Cryptanalysis of a one-time code-based digital signature scheme. In: 2019 IEEE International Symposium on Information Theory (ISIT). (2019) 2594–2598
21. Shor, P.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Journal on Computing **26**(5) (1997) 1484–1509
22. Song, Y., Huang, X., Mu, Y., Wu, W.: A new code-based signature scheme with shorter public key. IACR Cryptol. ePrint Arch. **2019** (2019) 53
23. Song, Y., Huang, X., Mu, Y., Wu, W., Wang, H.: A code-based signature scheme from the Lyubashevsky framework. Theoretical Computer Science **835** (2020) 15–30
24. Stern, J.: A new identification scheme based on syndrome decoding. In Stinson, D.R., ed.: Advances in Cryptology — CRYPTO' 93, Springer Berlin Heidelberg (1994) 13–21
25. Véron, P.: Improved identification schemes based on error-correcting codes. Applicable Algebra in Engineering, Communication and Computing **8**(1) (1997) 57–69