# The security of the code-based signature scheme based on the Stern identification protocol

Victoria Vysotskaya[1,2] and Ivan Chizhov[1,2,3]

[1] JSC "NPK Kryptonite", Russia
[2] Lomonosov Moscow State University, Russia
[3] Federal Research Center "Informatics and Control" of Russian Academy of Science, Russia
{v.vysotskaya, i.chizhov}@kryptonite.ru

**Abstract.** The paper provides a complete description of the digital signature scheme based on the Stern identification protocol. We also present the proof of the existential unforgeability of the scheme under the chosen message attack (EUF-CMA) in the random oracle model (ROM) under assumptions of hardness of syndrome decoding and hash function collision finding problems. Finally, we discuss the choice of the signature parameters and introduce a parameter set providing 80-bit security.

**Keywords:** post-quantum cryptography · code-based cryptography · digital signature · Stern's scheme · Fiat-Shamir transform · provable security · EUF-CMA security.

## 1 Introduction

The security of all standardized cryptographic algorithms used all around the world is based on the complexity of several number-theoretical problems. The latter include the discrete logarithm and factorization problems. However, in 1994 P. Shor showed [1] that quantum computers could break all schemes constructed this way. And in 2001 the Shor's algorithm was implemented on a 7-qubit quantum computer. Since then various companies have been actively developing more powerful quantum computers. Potential progress in this area poses a real threat to modern public-key cryptography.

This led to the emergence of so-called post-quantum cryptographic schemes. Most of them can be categorized into the following classes: code-based, lattice-based, multivariate, hash-based and isogeny-based. No successful quantum-computer attacks on "hard" problems from these areas are known.

The interest in code-based schemes as post-quantum ones can be noticed in the works submitted to the contest for prospective public-key post-quantum algorithms which was announced in 2016 by the US National Institute of Standards and Technology (NIST) [2]. The algorithms that win this contest will be accepted as US national standards. 21 of 69 applications filed (that is, almost a third of all works) were based on coding theory. However, it is worth noting that only three of them presented digital signature schemes. These were pqsigRM [3], RaCoSS [4] and RankSign [5] schemes. However, attacks on each of them were built during the peer review. The attack on the RankSign scheme was presented on Asiacrypt conference [6]. Out of the competition pqsigRM and RaCoSS schemes were fixed and presented as Modified pqsigRM [7] and RaCoSS-R [8], respectively. However, the RaCoSS-R scheme was also proven to be insecure [9]. As a result, none of the signatures based on the error-correcting codes made it to the final of the NIST competition.

In general, the development of code-based signature schemes was advancing less successfully than of the encryption ones. The first signature scheme of this type was KKS, presented by G. Kabatianskii, E. Krouk and B. Smeets in the paper [10] in 1997. However, in 2007 it was shown [11] that re-signing on one key pair leads to the disclosure of some information about the secret key. Thus it is necessary either to use the signature as a one-time one or use additional resources for building and maintaining auxiliary structure.

After that for a rather long time attacks on all proposed signature schemes were built so quickly that there was a fear that such schemes could not be created at all [12].

In 2001 N. Courtois, M. Finiasz and N. Sendrier presented a digital signature CFS [13] based on encryption schemes by R. McEliece [14] and H. Niederreiter [15] (provably secure version of this signature, called mCFS, was later proposed by L. Dallot in [16]). The authors used a decryption algorithm as the signature generation one. Unfortunately because of the inner decoding procedure with extremely small probability of success on a random input, the signature generation algorithm had to be repeated many times. Also, a significant disadvantage of CFS-type schemes is that their security depends on the assumption that the base code is indistinguishable from a random one. This leads to the emergence of attacks on signatures, previously considered provably secure. One of the latest schemes of this type is Wave [17], based on generalized $(U, U+V)$ codes.

Another approach to constructing a signature scheme is to apply the Fiat–Shamir transformation [18] to an identification protocol. For example, one may use identification schemes by J. Stern [19], A. Jain et al. [20] or CVE [21]. This method does not take into account features of codes. But it allows to prove the security without assumptions that depend on their structure. However, due to the fact that the basic scheme has a certain cheating probability, the signature algorithm has to be repeated several times, that leads to an increase in its operation time and in the resulting signature length.

This drawback is overcome in Lyubashevsky-type signatures, the original version of which is lattice-based [22]. Despite of the fact the original version remains secure, all known attempts to replace lattices with codes in Hamming metric resulted in the loss of security. However, a code-based signature in the rank metric called Durandal was proposed in [23] and is still secure. Yet it is not proven that the signature distribution is independent from the secret key and reveals no information on it. Moreover, the security proof is based on the hardness of a new problem PSSI+. Therefore, Stern-type schemes remain the only ones whose security proofs are based only on NP-hard problems.

Despite the fact that the signature based on the Stern identification scheme has been repeatedly mentioned in the literature, it has never been fully presented. For example, the review [24] by R. Overbeck and N. Sendrier only mentions the possibility of constructing such a signature, without giving the algorithm itself. In the paper [25] the scheme is formulated with an error, which leads to the significant decrease of the security level compared to the expected value. A correct, but short description of the scheme can be found in [26].

Moreover, the security proof of the scheme is considered to be proved by D. Pointcheval and J. Stern in [27]. This paper presents so-called Forking lemma by which the security of the signature scheme to existential forgery under an adaptively chosen-message attack in the random oracle model is proved. The authors mention there the applicability of the Forking lemma to the Stern signature scheme. However this fact was not proven neither in this article nor elsewhere later.

In our work we provide a complete description of the signature based on the Stern identification scheme along with the proof of the existential unforgeability under the chosen message attack (EUF-CMA) under assumptions of hardness of syndrome decoding and hash function collision finding problems.

The rest of this paper is structured as follows. In Section 2 we give basic definitions, describe some hard problems and show the original Stern identification protocol. We present the signature scheme together with the security model in Section 3. Section 4 is devoted to the security proof of our signature in the EUF-CMA model. We give some restrictions on the scheme parameters and introduce an example parameter set in Section 5. Finally, conclusions are presented in Section 6.

## 2 Definitions and Preliminary Results

Our signature is based on linear block error-correcting codes. We will call them *codes* for brevity.

We denote by $S_n$ the symmetric group of order $n$, i.e. the group of all permutations of elements of the set $\{1, \dots, n\}$. If $\sigma \in S_n$, $u \in \{0,1\}^n$, then $\sigma(u) \in \{0,1\}^n$, $\sigma(u)_i = u_{\sigma(i)}$. The weight of the vector $u$ is the number of its nonzero elements. It is denoted by $\mathrm{wt}(u)$.

The security of the signature scheme is based on the hardness of the following problems.

Problem $\mathbf{SD}(\mathbf{H}, \mathbf{y}, \omega)$. **Syndrome Decoding**
**Input**: $(n-k) \times n$ parity-check matrix $H$ of some binary code, nonzero vector $y \in \{0,1\}^{n-k}$, called *syndrome*, and number $\omega > 0$.
**Output**: vector $e \in \{0,1\}^n$ such that $\mathrm{wt}(e) = \omega$ and $He^T = y^T$.

Problem $\mathbf{Coll}(\mathbf{h})$. **Collision Finding**
**Input**: hash function $h : \{0,1\}^* \to \{0,1\}^\ell$.
**Output**: vectors $x', x'' \in \{0,1\}^*, x' \neq x''$ such that $h(x') = h(x'')$.

The former problem is known to be NP-hard [28]. The best known algorithm solves it in $\mathcal{O}\left(2^{0.0885n}\right)$ bit operations [29]. The complexity of latter problem depends on the structure of the function $h$. In the general case, the complexity of solving such a problem using the birthday paradox can be estimated as $\mathcal{O}\left(2^{\frac{\ell}{2}}\right)$.

Let us recall the Stern identification protocol presented in [19]. The protocol parameters depend on the parameters of the underlying code: its length $n$, dimension $k$ and minimum distance $\omega$. The parity-check matrix of this code is a random matrix $H \in \{0,1\}^{(n-k)\times n}$. Also, the protocol is based on a hash function $h(\cdot) : \{0,1\}^* \to \{0,1\}^\ell$.

To generate a secret key one randomly uniformly chooses $s = \{0,1\}^n$ such that $\mathrm{wt}(s) = \omega$. Now public key can be derived as $y = Hs^T$. The description of the identification protocol is shown on Fig. 1. Here the notation $s \overset{\mathcal{U}}{\leftarrow} S$ means that $s$ is chosen from the set $S$ uniformly at random. We also denote the assignment of value $v$ to $x$ by $x \leftarrow v$.

In his paper Stern proposes a strategy for an adversary to pass identification without knowing the secret key with probability of success equal to 2/3. So to reduce this value and to reach the required level of security one should repeat the algorithm several times.

Recall the general definition of a digital signature scheme.

**Definition 1.** *A digital signature scheme is a triple $\Sigma = (KeyGen, Sig, Ver)$ of (possibly probabilistic) polynomial time algorithms, where*

1. *$KeyGen()$ outputs a key pair $(pk, sk)$.*
2. *$Sig(sk, m)$ receives as input the secret key $sk$ and a message $m \in \{0,1\}^*$ and outputs a signature $\zeta$.*
3. *$Ver(pk, m, \zeta)$ receives as input the public key $pk$, a message $m$ and a signature $\zeta$. It outputs 0 or 1, where 1 means that $\zeta$ is accepted as a signature for message $m$ and public key $pk$. 0 means that the signature is not accepted.*

## 3 Signature scheme

In this section we show the digital signature scheme that is the result of the Fiat–Shamir transformation applied to the Stern identification scheme. The transformation consists of replacing the random value $b$, generated by the verifier, by some function $f$ of the message and values received from the prover. It is important for $f$ to depend on all of these values at once.

Parameters of the signature are the same as in the original identification protocol described in Section 2. Additionally the scheme uses a hash function $f(\cdot) : \{0,1\}^* \to \{0,1,2\}^\delta$. The length of the signature depends on the parameter $\delta$ that is determined by the security parameter $\lambda$.

| **Prover**$(s)$ | | **Verifier**$(y)$ |
|---|---|---|
| $u \overset{\mathcal{U}}{\leftarrow} \{0,1\}^n, \sigma \overset{\mathcal{U}}{\leftarrow} S_n$ | | |
| $c_0 \leftarrow h(\sigma \| Hu^T)$ | | |
| $c_1 \leftarrow h(\sigma(u))$ | | |
| $c_2 \leftarrow h(\sigma(u \oplus s))$ | | |
| | $\xrightarrow{c_0, c_1, c_2}$ | |
| | | $b \overset{\mathcal{U}}{\leftarrow} \{0,1,2\}$ |
| | $\xleftarrow{b}$ | |
| **if** $b = 0 : r_0 \leftarrow \sigma, r_1 \leftarrow u$ | | |
| **if** $b = 1 : r_0 \leftarrow \sigma, r_1 \leftarrow u \oplus s$ | | |
| **if** $b = 2 : r_0 \leftarrow \sigma(u), r_1 \leftarrow \sigma(s)$ | | |
| | $\xrightarrow{r_0, r_1}$ | |
| | | **if** $b = 0 :$ |
| | | **Check** $c_0 \overset{?}{=} h(r_0 \| Hr_1^T)$, |
| | | $c_1 \overset{?}{=} h(r_0(r_1))$ |
| | | **if** $b = 1 :$ |
| | | **Check** $c_0 \overset{?}{=} h(r_0 \| (Hr_1^T \oplus y))$, |
| | | $c_2 \overset{?}{=} h(r_0(r_1))$ |
| | | **if** $b = 2 :$ |
| | | **Check** $c_1 \overset{?}{=} h(r_0)$, |
| | | $c_2 \overset{?}{=} h(r_0 \oplus r_1)$, |
| | | $\mathrm{wt}(r_1) \overset{?}{=} \omega$ |

**Fig. 1.** Stern identification scheme

```
$Stern.KeyGen()$
$s \overset{\mathcal{U}}{\leftarrow} \{\{0,1\}^n : \mathrm{wt}(s) = \omega\}$
$y \leftarrow Hs^T$
**return** $(y, s)$
```

```
$Stern.Sig(s, m)$
**foreach** $0 \leqslant i < \delta$ :
    $u_i \overset{\mathcal{U}}{\leftarrow} \{0,1\}^n, \sigma_i \overset{\mathcal{U}}{\leftarrow} S_n$
    $c_{i,0} \leftarrow h(\sigma_i \| Hu_i^T)$
    $c_{i,1} \leftarrow h(\sigma_i(u_i))$
    $c_{i,2} \leftarrow h(\sigma_i(u_i \oplus s))$
    $c_i \leftarrow c_{i,0} \| c_{i,1} \| c_{i,2}$
$c \leftarrow c_0 \| \ldots \| c_{\delta-1}$
$b \leftarrow f(m \| c)$
**foreach** $0 \leqslant i < \delta$ :
    **if** $b_i = 0 : r_i \leftarrow \sigma_i \| u_i$
    **if** $b_i = 1 : r_i \leftarrow \sigma_i \| (u_i \oplus s)$
    **if** $b_i = 2 : r_i \leftarrow \sigma_i(u_i) \| \sigma_i(s)$
$r \leftarrow r_0 \| \ldots \| r_{\delta-1}$
**return** $c \| r$
```

```
$Stern.Ver(y, m, (c\|r))$
1: $b \leftarrow f(m\|c)$
2: foreach $0 \leqslant i < \delta$ :
3:     if $[b_i = 0] \wedge \Big[[c_{i,0} \neq h(r_{i,0}\|Hr_{i,1}^T)] \vee [c_{i,1} \neq h(r_{i,0}(r_{i,1}))]\Big]$ :
4:         return 0
5:     if $[b_i = 1] \wedge \Big[[c_{i,0} \neq h(r_{i,0}\|(Hr_{i,1}^T \oplus y))] \vee [c_{i,2} \neq h(r_{i,0}(r_{i,1}))]\Big]$ :
6:         return 0
7:     if $[b_i = 2] \wedge \Big[[c_{i,1} \neq h(r_{i,0})] \vee [c_{i,2} \neq h(r_{i,0} \oplus r_{i,1})] \vee [\mathrm{wt}(r_{i,1}) \neq \omega]\Big]$ :
8:         return 0
9: return 1
```

To estimate the scheme security under different assumptions about the adversary we construct experiments, where the adversary is represented by a probabilistic polynomial-time Turing machine. The notation **Exp** $\Rightarrow b$ means that $b$ is the output of the experiment **Exp**. We write **abort** in the oracle pseudocode to denote that experiment should stop and return 0. We denote the set of all mappings from set $A$ to set $B$ by $Func(A, B)$. To emphasize the fact that $x$ is the result of a probabilistic algorithm $A$ we write $x \leftarrow\$ A(...)$.

To model a random oracle $F : \{0,1\}^* \to \{0,1,2\}^\delta$ we use lazy sampling. We introduce the set $\Pi^F$, which contains pairs of form $(\alpha, F(\alpha))$. Further we write $(\alpha, \cdot) \in \Pi^F$ for $\alpha \in \{0,1\}^*$ to show that there exists $\beta \in \{0,1,2\}^\delta$ such that $(\alpha, \beta) \in \Pi^F$. As far as $\Pi^F$ contains not more than one pair $(\alpha, \beta)$ for each $\alpha$, then $\Pi^F(\alpha)$ denotes either $\beta$ if $(\alpha, \beta) \in \Pi^F$ or special value $\perp$ if there is no such a pair.

**Definition 2.** *For the signature scheme Stern.$\Sigma$ we denote the advantage of the adversary $\mathcal{A}$ in $q_f$-EUF-NMA model with random oracle access by*

$$\mathsf{Adv}_{Stern}^{q_f\text{-EUF-NMA}}(\mathcal{A}) = \Pr[\mathbf{Exp}_{Stern}^{q_f\text{-EUF-NMA}}(\mathcal{A}) \Rightarrow 1],$$

*where the experiment $\mathbf{Exp}_{Stern}^{q_f\text{-EUF-NMA}}(\mathcal{A})$ is defined as follows:*

```
$\mathbf{Exp}_{Stern}^{q_f\text{-EUF-NMA}}(\mathcal{A})$
1: $(pk, sk) \leftarrow\$ Stern.KeyGen()$
2: $\Pi^F \leftarrow \varnothing$
3: $(m, \zeta) \leftarrow\$ \mathcal{A}^F(pk)$
4: return $Stern.Ver(pk, m, \zeta)$
```

```
Oracle $F(\alpha)$
1: if $\alpha \in \Pi^F$ : $\beta \leftarrow \Pi^F(\alpha)$
2: else
3:     $\beta \xleftarrow{\mathcal{U}} \{0,1,2\}^\delta$
4:     $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
5: return $\beta$
```

*Parameter $q_f$ corresponds to the number of queries made by the adversary $\mathcal{A}$ to the oracle $F$. If this does not cause ambiguity, we will omit this parameter.*

**Definition 3.** *For the signature scheme Stern.$\Sigma$ we denote the advantage of the adversary $\mathcal{A}$ in $(q_f, q_s)$-EUF-CMA model with random oracle access by*

$$\mathsf{Adv}^{(q_f,q_s)\text{-EUF-CMA}}_{Stern}(\mathcal{A}) = \Pr[\mathbf{Exp}^{(q_f,q_s)\text{-EUF-CMA}}_{Stern}(\mathcal{A}) \Rightarrow 1],$$

*where the experiment* $\mathbf{Exp}^{(q_f,q_s)\text{-EUF-CMA}}_{Stern}$ *is defined as follows:*

```
$\mathbf{Exp}^{(q_f,q_s)\text{-EUF-CMA}}_{Stern}(\mathcal{A})$
1 : $(pk, sk) \leftarrow\$ Stern.KeyGen()$
2 : $\mathcal{L} \leftarrow \varnothing$
3 : $\Pi^F \leftarrow \varnothing$
4 : $(m, \zeta) \leftarrow\$ \mathcal{A}^{Sign,F}(pk)$
5 : if $m \in \mathcal{L}$ : return 0
6 : return $Stern.Ver(pk, m, \zeta)$
```

```
Oracle $F(\alpha)$
1 : if $\alpha \in \Pi^F : \beta \leftarrow \Pi^F(\alpha)$
2 : else
3 :     $\beta \overset{\mathcal{U}}{\leftarrow} \{0,1,2\}^\delta$
4 :     $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
5 : return $\beta$
```

```
Oracle $Sign(m)$
1 : $\zeta \leftarrow\$ Stern.Sig(sk, m)$
2 : $\mathcal{L} \leftarrow \mathcal{L} \cup \{m\}$
3 : return $\zeta$
```

*Parameters $q_f$ and $q_s$ correspond to the number of queries made by the adversary $\mathcal{A}$ to the oracles $F$ and Sign, respectively. They can also be omitted.*

## 4 Security bounds

Let us give several definitions that will be needed below.

**Definition 4.** *If $T$ is a ternary tree of depth $\delta$ with $N$ leaves, then the density of $T$ is defined as $N/3^\delta$.*

**Definition 5.** *Let us call a tree a $\rho$-dense one if its density is not less than $\rho$.*

**Definition 6.** *We call a tree a uniformly $\rho$-dense one if each of its subtrees, excluding leaves, is a $\rho$-dense tree.*

**Proposition 1.** *A $\rho$-dense tree $T$ with all leaves having depth $\delta$, considered as a graph, contains a subgraph that is a uniformly $\frac{\rho}{\delta}$-dense tree with the same root. Moreover, each of its leaves has depth $\delta$.*

*Proof.* Let us describe the algorithm to choose such a subgraph. We start from the $(\delta-1)$-th level of the tree and move to the root (level 0) disposing of vertices that are roots of subtrees of density less than $\theta = \frac{\rho}{\delta}$. Note that until the algorithm stops (i.e. reaches the root) some leaves may have depth less than $\delta$. However, after the algorithm completion each of the survived leaves will have depth $\delta$.

Let us show that at each step of this algorithm the root density decreases by no more than $\theta$. Suppose that there are $\kappa$ vertices at the $i$-th level of the original tree $T$. The densities of subtrees formed by them are $\rho_{i,1}, \dots, \rho_{i,\kappa}$. If we denote the number of leaves of $T$ by $t$, then

$$\rho_{i,1} + \dots + \rho_{i,\kappa} = \frac{t}{3^{\delta-i}} = 3^i \rho.$$

After the step of the algorithm at the $i$-th level, some of these vertices may be disposed of, namely those that have density less than $\theta$. Thus the new density $\rho'_{i,j}$ may either be equal to $\rho_{i,j}$ or become 0 if $\rho_{i,j} < \theta$. So

$$\rho'_{i,1} + \dots + \rho'_{i,\kappa} \geqslant \rho_{i,1} + \dots + \rho_{i,\kappa} - \kappa\theta \geqslant 3^i \rho - \kappa\theta.$$

Thus, for the new density of the root $\rho'$ holds $3^i \rho' \geqslant 3^i \rho - \kappa\theta$ and

$$\rho' \geqslant \rho - \frac{\kappa}{3^i}\theta \geqslant \rho - \theta.$$

As a result of all deletions, $\rho$ has decreased by at most $(\delta-1)\theta$. Since $\theta = \frac{\rho}{\delta}$, then

$$\rho - (\delta-1)\frac{\rho}{\delta} = \frac{\rho\delta - (\delta-1)\rho}{\delta} = \frac{\rho}{\delta} = \theta.$$

So the resulting tree is uniformly $\theta$-dense.

**Theorem 1.** *Let $\mathcal{A}$ be an adversary with time complexity at most $T$ in the EUF-NMA model for Stern signature scheme, making at most one query to the hashing oracle $F$, then*

$$\mathsf{Adv}_{Stern}^{1\text{-}EUF\text{-}NMA}(\mathcal{A}) \leqslant 6 \cdot \sqrt[3]{\frac{\delta^2 T}{\min\{T_{SD}, T_{Coll}\}}} + \left(\frac{2}{3}\right)^{\delta},$$

*where $T_{SD}$ and $T_{Coll}$ are complexities of optimal algorithms solving $SD(H, y, \omega)$ and $Coll(h)$ problems with probabilities of success at least $1 - \frac{1}{e}$.*

*Proof.* Denote

$$\varepsilon = \mathsf{Adv}_{Stern}^{1\text{-}EUF\text{-}NMA}(\mathcal{A}) - \left(\frac{2}{3}\right)^{\delta}. \tag{1}$$

In case if $\varepsilon \leqslant 0$ the proof is complete. Therefore, further we will consider the case

$$\mathsf{Adv}_{Stern}^{1\text{-}EUF\text{-}NMA}(\mathcal{A}) = \left(\frac{2}{3}\right)^{\delta} + \varepsilon,\ \varepsilon > 0.$$

We can represent the execution of the adversary $\mathcal{A}$ at all outputs of the random oracle $F$ as an incomplete ternary tree $T(x)$, each leaf of which has depth $\delta$. It is determined by $\mathcal{A}$'s random tape $x$. Each output $b_i$ of the random oracle corresponds to a certain path in the tree. If the corresponding $b_i$ equals 0 the vertex has the left child, if $b_i = 1$ then the vertex has the middle one and if $b_i = 2$ then it has the right one. If the adversary was not able to build a signature for some output of the random oracle correctly, then the corresponding branch is removed from the tree. Note that fixing the adversary's random tape we guarantee that at each level of the tree the same part of the signature, corresponding to $c_{i,0}\|c_{i,1}\|c_{i,2}$, is checked.

Let us show that if there exists a level $i$ with a vertex with a left child, a vertex with a middle child and a vertex with a right child (denoted respectively $v_{i,0}$, $v_{i,1}$ and $v_{i,2}$), then one of the $SD(H, y, \omega)$ and $Coll(h)$ problems can be solved. Note that some of vertices $v_{i,0}, v_{i,1}$ and $v_{i,2}$ may coincide. Later we will present the algorithm that let the adversary $\mathcal{A}$ find such vertices in the tree $T(x)$ with probability $1 - 1/e$.

Let the tree have such vertices. In this case the adversary has successfully generated three signatures on outputs of random oracle that all differ in the $i$-th trit. Let $r_{i,0} = \sigma_0$ and $r_{i,1} = u_0$ for $b_i = 0$. For $b_i = 1$ let $r_{i,0} = \sigma_1$ and $r_{i,1} = w_1$, where $w_1$ corresponds to $u_i \oplus s$. Finally, for $b_i = 2$ let $r_{i,0} = z_2$ and $r_{i,1} = t_2$, where $z_2$ corresponds to $\sigma_i(u_i)$ and $t_2$ corresponds to $\sigma_i(s)$. Since $c_0$ can be obtained in two cases ($b_i = 0$ and $b_i = 1$), then

$$c_{i,0} = h(\sigma_0 \| H u_0^T) = h(\sigma_1 \| H w_1^T \oplus y).$$

Hence, either collision of hash function $h$ can be found or $\sigma_0 = \sigma_1$ and $Hu_0^T = Hw_1^T \oplus y$. Similarly it can be shown that if no collisions were found, then $z_2 = \sigma_0(u_0)$ and $z_2 \oplus t_2 = \sigma_1(w_1)$. Note that as the third answer was accepted, $t_2$ satisfies the weight constraint. Denoting $\sigma = \sigma_0 = \sigma_1$ we have

$$t_2 = z_2 \oplus (t_2 \oplus z_2) = \sigma(u_0 \oplus w_1).$$

Therefore, $u_0 \oplus w_1$ also has the acceptable weight. Then

$$H(u_0 \oplus w_1)^T = Hu_0^T \oplus Hw_1^T = y$$

and $u_0 \oplus w_1$ is an acceptable secret key.

Now let us describe an algorithm that finds a tree with vertices $v_{i,0}, v_{i,1}$ and $v_{i,2}$ with some probability.

**Algorithm 1**

1. Randomly choose a value $x$ of the adversary's random tape (i.e. fix the tree $T(x)$).
2. Fix $\frac{6T}{\theta^2}$ outputs of the random oracle (defining the number of branches of the tree).
3. Traverse the tree level by level to find vertices $v_{i,0}, v_{i,1}$ and $v_{i,2}$. If they are found, then compute an equivalent secret key $s'$ as described above, create a signature for a random message on this key using the original signature algorithm and return this message and the signature. Otherwise return to Step 1.

**Lemma 1.** *Under the assumptions of Theorem 1, the probability of success of each run of Algorithm 1 is not less than* $\frac{\varepsilon}{4}$, *where* $\varepsilon$ *is defined as in* (1).

*Proof.* Define the set $X$ as

$$X = \{x \mid \text{there are at least } 2^\delta + \frac{\varepsilon}{2} \cdot 3^\delta \text{ branches in } T(x)\}.$$

Then $\Pr[x \in X] \geqslant \varepsilon/2$.

Let, on the contrary, $\Pr[x \in X] < \varepsilon/2$. Let us denote the number of leaves of $T(x)$ by $t$. Then $\Pr[\mathcal{A} \Rightarrow 1 \wedge x \notin X] = t/3^\delta < (2/3)^\delta + \varepsilon/2$. Therefore, the success probability of $\mathcal{A}$ is

$$\Pr[\mathcal{A} \Rightarrow 1] = \Pr[\mathcal{A} \Rightarrow 1 \wedge x \in X] + \Pr[\mathcal{A} \Rightarrow 1 \wedge x \notin X] \leqslant \Pr[x \in X] + \\ + \Pr[\mathcal{A} \Rightarrow 1 \wedge x \notin X] < \varepsilon/2 + \left((2/3)^\delta + \varepsilon/2\right) = (2/3)^\delta + \varepsilon.$$

And we came to a contradiction.

Let us consider separately the case $x \in X$. Note that $X$ defines a set of $\varepsilon/2$-dense trees. Therefore, by Proposition 1 one can select a uniformly $\theta$-dense subtree with leaves of depth $\delta$ from each such tree. Here $\theta = \frac{\varepsilon}{2\delta}$. Let us call this tree $T_1(x)$.

For any index $j$, $0 \leqslant j \leqslant \delta$ we denote by $n_j$ the number of vertices at the $j$-th level of $T_1(x)$. Also for $0 \leqslant j < \delta$ we define the value $\alpha_j = n_{j+1}/n_j$. Let us denote $\theta' = \theta - \left(\frac{2}{3}\right)^\delta$. Then, since $n_0 = 1$ ($j = 0$ corresponds the root of the tree),

$$\prod_{j=0}^{\delta-1} \alpha_j \geqslant 2^\delta + \theta' \cdot 3^\delta$$

or, equivalently,

$$\sum_{j=0}^{\delta-1} \log_2(\alpha_j) \geqslant \log_2\left(2^\delta + \theta' \cdot 3^\delta\right).$$

Using the convexity of the logarithm, we can extend this inequality as

$$\sum_{j=0}^{\delta-1} \log_2(\alpha_j) \geqslant \delta + \theta'\delta \log_2 3.$$

Let $i = \arg \max\limits_{j=0,\ldots,\delta-1} \{\alpha_j\}$. Then $\log_2(\alpha_i) \geqslant 1 + \theta' \log_2 3$. From this and the fact that the inequality $e^x \geqslant 1 + x$ holds we obtain

$$\alpha_i \geqslant 2 \cdot 2^{\theta' \log_2 3} \geqslant 2 + 2\theta' \ln 2 \log_2 3.$$

Denote by $n_{i,\leqslant 2}$ and $n_{i,3}$ the number of vertices of $T(x)$ having no more than two children and exactly three children at $i$-th level, respectively. Then

$$\alpha_i \leqslant \frac{2n_{i,\leqslant 2} + 3n_{i,3}}{n_{i,\leqslant 2} + n_{i,3}} = 2 + \frac{n_{i,3}}{n_i}.$$

This shows that for the ratio of vertices with three children $\Delta$ at $i$-th level it holds

$$\Delta \geqslant 2\theta' \ln 2 \log_2 3 > 2\theta. \tag{2}$$

Let there be $\kappa$ vertices at this level: $v_1, \dots, v_\kappa$. Then

$$\Delta = \frac{\kappa}{3^i}. \tag{3}$$

We denote by $L_i(\pi)$ the predicate that the path $\pi$ lies in $T_1(x)$ and goes through the left child of some vertex of level $i$. Similarly, we define predicates $C_i(\pi)$ and $R_i(\pi)$. By $L_i$ we denote the predicate $(\exists \pi : L_i(\pi))$.

We can write for the probability of the event $L_i$:

$$\begin{aligned}\Pr[L_i] = \Pr[\exists \pi : (v_1 \in \pi \vee v_2 \in \pi \vee \dots \vee v_\kappa \in \pi) \wedge L_i(\pi)] =\\ = \sum_{i=1}^{\kappa} \Pr[\exists \pi : (v_i \in \pi) \wedge L_i(\pi)]\end{aligned}$$

The probability $\Pr[\exists \pi : (v_i \in \pi) \wedge L_i(\pi)]$ is equal to the number $S$ of paths in $T_1(x)$ passing through the left child of $v_i$ divided by $3^\delta$. Since $T_1(x)$ is a uniformly $\theta$-dense tree, it holds that

$$\theta \leqslant \frac{S}{3^{\delta-i-1}} \Rightarrow S \geqslant 3^{\delta-i-1}\theta \Rightarrow \Pr[\exists \pi : (v_i \in \pi) \wedge L_i(\pi)] \geqslant \frac{3^{\delta-i-1}\theta}{3^\delta}. \tag{4}$$

From this, from (2) and (3) we can finally conclude that

$$\Pr[L_i] \geqslant \kappa \cdot \frac{3^{\delta-i-1}\theta}{3^\delta} \geqslant \frac{2\theta^2}{3}.$$

Now let us find the probability $P$ that choosing $3/\theta^2$ branches $\pi_j$ we find vertices $v_{i,0}, v_{i,1}$ and $v_{i,2}$ at the $i$-th level of $T_1(x)$.

$$\begin{aligned}P &= \Pr[\exists j_0, j_1, j_2 : L_i(\pi_{j_0}) \wedge C_i(\pi_{j_1}) \wedge R_i(\pi_{j_2})] =\\ &= 1 - \Pr[\nexists j_0 : L_i(\pi_{j_0}) \vee \nexists j_1 : C_i(\pi_{j_1}) \vee \nexists j_2 : R_i(\pi_{j_2})] \geqslant\\ &\geqslant 1 - \Pr[\nexists j_0 : L_i(\pi_{j_0})] - \Pr[\nexists j_1 : C_i(\pi_{j_1})] - \Pr[\nexists j_2 : R_i(\pi_{j_2})] =\\ &= 1 - 3\Pr[\nexists j_0 : L_i(\pi_{j_L})] = 1 - 3\Pr[\overline{L}_i]^{\frac{3}{\theta^2}} = 1 - 3(1 - \Pr[L_i])^{\frac{3}{\theta^2}} \geqslant\\ &\geqslant 1 - 3\left(1 - \frac{2\theta^2}{3}\right)^{\frac{3}{\theta^2}} \geqslant 1 - \frac{3}{e^2}.\end{aligned}$$

Thus, the success probability of Algorithm 1 searching vertices $v_{i,0}, v_{i,1}$ and $v_{i,2}$ is obtained from the probability of choosing a dense tree $T(x)$ and the probability $P$. It equals $p := \varepsilon/2 \cdot (1 - 3/e^2) > \varepsilon/4$.

The complexity of $\mathcal{A}$ builds up from the complexity of $3/\theta^2$ runs of Algorithm 1 and is equal to $T' = 3T/\theta^2 = 12\delta^2 T/\varepsilon^2$. In order to achieve a constant probability of success, $\mathcal{A}$ must repeat the above algorithm $1/p$ times. The probability that after $1/p$ runs there is still no success is $(1-p)^{1/p}$. Then $\mathcal{A}$'s probability of success is $1 - (1-p)^{1/p}$. Let us show that

$$1 - (1-p)^{\frac{1}{p}} > 1 - \frac{1}{e}.$$

Indeed, the Maclaurin series for $1/(1-p)$ and $e^p$ are:

$$\frac{1}{1-p} = 1 + p + p^2 + \dots, \quad e^p = 1 + \frac{p}{1!} + \frac{p^2}{2!} + \dots,$$

thus for all $p \in (0,1)$ holds

$$1/(1-p) > e^p \Rightarrow (1-p) < \frac{1}{e^p} \Rightarrow (1-p)^{\frac{1}{p}} < \frac{1}{e}.$$

The resulting complexity of the adversary $\mathcal{A}$ is $T'' := T'/p = 48\delta^2 T/\varepsilon^3$. Let $\mathcal{A}$ solve $SD(H, y, \omega)$ and $Coll(h)$ with probabilities $p_1$ and $p_2$, respectively. Then

$$p_1 + p_2 \geqslant 1 - \frac{1}{e}.$$

We denote by $T_{SD,(1-1/e)}$ and $T_{Coll,(1-1/e)}$ complexities of optimal algorithms, solving $SD(H, y, \omega)$ and $Coll(h)$ with success probability $1 - 1/e$. Then

$$T_{SD,(1-1/e)} \leqslant \frac{1}{p_1} T_{SD,p_1} \leqslant \frac{1}{p_1} T'',$$

$$T_{Coll,(1-1/e)} \leqslant \frac{1}{p_2} T_{Coll,p_2} \leqslant \frac{1}{p_2} T''.$$

The first inequalities follow from the fact that repeating an algorithm with a success probability $p_1$ for $1/p_1$ times gives an algorithm with a success probability of $1 - 1/e$, but possibly suboptimal. The second inequality follows from the fact that $\mathcal{A}$ solves one of two problems. Accordingly, its complexity cannot be less than the complexity of the algorithm that solves one of them.

Hence,

$$T'' \geqslant p_1 T_{SD,(1-1/e)} \quad \text{and} \quad T'' \geqslant p_2 T_{Coll,(1-1/e)}.$$

Therefore, denoting $\widetilde{T} = \min\{T_{SD,(1-1/e)}, T_{Coll,(1-1/e)}\}$, we can write

$$T'' \geqslant \frac{1}{2}\bigl(p_1 T_{SD,(1-1/e)} + p_2 T_{Coll,(1-1/e)}\bigr) \geqslant \frac{1}{2}(p_1 + p_2)\widetilde{T} \geqslant \frac{1 - \frac{1}{e}}{2} \widetilde{T}.$$

Equivalently,

$$\frac{48\delta^2 T}{\varepsilon^3} \geqslant \frac{1 - \frac{1}{e}}{2} \widetilde{T}.$$

Finding $\varepsilon$ from the last inequality and noting that $\sqrt[3]{\frac{96}{1-\frac{1}{e}}} \leqslant 6$, we obtain:

$$\varepsilon \leqslant 6 \cdot \sqrt[3]{\frac{\delta^2 T}{\widetilde{T}}}.$$

Finally,

$$\mathsf{Adv}_{Stern}^{\text{1-EUF-NMA}}(\mathcal{A}) \leqslant 6 \cdot \sqrt[3]{\frac{\delta^2 T}{\min\{T_{SD}, T_{Coll}\}}} + \left(\frac{2}{3}\right)^{\delta}.$$

**Theorem 2.** *Let $\mathcal{A}$ be an adversary in the EUF-NMA model for Stern signature scheme, making at most $q_f$ queries to the hashing oracle $F$. Then there exists an adversary $\mathcal{B}$ in the 1-EUF-NMA model for Stern signature scheme making at most one query to the hashing oracle and satisfying*

$$q_f \cdot \mathsf{Adv}^{\text{1-EUF-NMA}}_{Stern}(\mathcal{B}) \geqslant \mathsf{Adv}^{\text{EUF-NMA}}_{Stern}(\mathcal{A}) - \left(\frac{1}{3}\right)^{\delta}.$$

*Furthermore, if the complexity of $\mathcal{A}$ is $T$, then the complexity of $\mathcal{B}$ is $T + c'q_f$, where $c'$ is a constant depending on the model of computation.*

*Proof.* Let $\mathbf{Exp}^0$ denote the original experiment in the EUF-NMA security model with $q_f$ queries to the hashing oracle $F$. In this experiment $\mathcal{A}$ is the adversary that makes an existential forgery for the Stern signature scheme using the random oracle $F$. Therefore,

$$\mathsf{Adv}^{\text{EUF-NMA}}_{Stern}(\mathcal{A}) := \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1].$$

```
$\mathbf{Exp}^1(\mathcal{A})$
1: $s \overset{\mathcal{U}}{\leftarrow} \{\{0,1\}^n : \mathrm{wt}(s) = \omega\}$
2: $y \leftarrow Hs^T$
3: $\Pi^F \leftarrow \varnothing$
4: $(m, c\|r) \leftarrow\$ \mathcal{A}^F(y)$
5: return $Stern.Ver(y, m, c\|r)$
```

```
Oracle $F(\alpha)$
1: if $\alpha \in \Pi^F : \beta \leftarrow \Pi^F(\alpha)$
2: else
3:     $\beta \overset{\mathcal{U}}{\leftarrow} \{0,1,2\}^{\delta}$
4:     $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
5: return $\beta$
```

Now basing on the adversary $\mathcal{A}$ we construct an adversary $\mathcal{B}$ that makes an existential forgery in the model with one query to the random oracle. $\mathcal{B}$ simulates the oracle $F$ that can give $q_f$ answers to $\mathcal{A}$'s queries using algorithm $SimF_t$. Here the notation $\mathcal{B}^{SimF_t}$ means that the only $\mathcal{B}$'s query to its own random oracle $F^*$ matches the $\mathcal{A}$'s $t$-th query to the oracle $F$. Note that the output of the oracle $F^*$ has a uniform distribution, i.e. values $\beta$ obtained on lines 3 and 4 of $SimF_t$ cannot be distinguished.

```
$\mathcal{B}^{F^*}(y)$
1: $\Pi^F \leftarrow \varnothing$
2: $j \leftarrow 0$
3: $t \overset{\mathcal{U}}{\leftarrow} \{1, \ldots, q_f'\}$
4: $(m, c\|r) \leftarrow\$ \mathcal{A}^{SimF_t}(y)$
5: return $(m, c\|r)$
```

```
$SimF_t(\alpha)$
1: $j \leftarrow j + 1$
2: if $(\alpha, \cdot) \in \Pi^F : \beta \leftarrow \Pi^F(\alpha)$
3: elseif $j = t : \beta \leftarrow F^*(\alpha)$
4: else $: \beta \overset{\mathcal{U}}{\leftarrow} \{0,1,2\}^{\delta}$
5: $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
6: return $\beta$
```

The adversary $\mathcal{A}$ can make a signature including answer for one of the queries made to the oracle $F$ or none of them. Let $I$ be a random variable that corresponds to the number of $\mathcal{A}$'s query to the oracle $F$ that it uses to create a forgery.

In case $\mathcal{A}$ does not use any, let $I = 0$. Hence,

$$\Pr[\mathbf{Exp}_{Stern}^{\text{1-EUF-NMA}}(\mathcal{B}) \Rightarrow 1] \geqslant \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1 \wedge t = I] \geqslant \\ \geqslant \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1 \wedge t = I \wedge I \geqslant 1].$$

Note that $\mathcal{A}$'s probability of success in case it does not use any query to random oracle $F$ is no more than $\left(\frac{1}{3}\right)^\delta$ as it has to guess full output $b = F(\alpha)$. From this and the definition of conditional probability holds

$$\Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1] \leqslant \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1 \wedge I \geqslant 1] + \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1 \wedge I = 0] \leqslant \\ \leqslant \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1 \wedge I \geqslant 1] + \left(\frac{1}{3}\right)^\delta.$$

Then

$$\Pr[\mathbf{Exp}_{Stern}^{\text{1-EUF-NMA}}(\mathcal{B}) \Rightarrow 1] \geqslant \Pr[t = I] \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1 \wedge I \geqslant 1] \geqslant \\ \geqslant \frac{1}{q_f} \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1 \wedge I \geqslant 1].$$

Consequently,

$$\Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1] - \left(\frac{1}{3}\right)^\delta \leqslant q_f \cdot \Pr[\mathbf{Exp}_{Stern}^{\text{1-EUF-NMA}}(\mathcal{B}) \Rightarrow 1].$$

From the above holds

$$q_f \cdot \mathsf{Adv}_{Stern}^{\text{1-EUF-NMA}}(\mathcal{B}) = q_f \cdot \Pr[\mathbf{Exp}(\mathcal{B}) \Rightarrow 1] \geqslant \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1] - \left(\frac{1}{3}\right)^\delta = \\ = \mathsf{Adv}_{Stern}^{\text{EUF-NMA}}(\mathcal{A}) - \left(\frac{1}{3}\right)^\delta.$$

$\mathcal{B}$ runs $\mathcal{A}$ and simulate $q_f$ queries to oracle $F$. I.e. if the complexity of $\mathcal{A}$ is $T$, then the complexity of $\mathcal{B}$ does not exceed $T + c'q_f$ for some constant $c'$.

**Theorem 3.** *Let $\mathcal{A}$ be an adversary in the EUF-CMA model for Stern signature scheme, making at most $q_f$ queries to the hashing oracle $F$ and at most $q_s$ queries to the signing oracle Sign. Then there exists an adversary $\mathcal{B}$ in the EUF-NMA model for Stern signature scheme making at most $q_f$ queries to the hashing oracle and*

$$\mathsf{Adv}_{Stern}^{\text{EUF-NMA}}(\mathcal{B}) \geqslant \mathsf{Adv}_{Stern}^{\text{EUF-CMA}}(\mathcal{A}) - q_f^2 q_s \left(\max\left\{\frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n} n!}\right\}\right)^\delta,$$

*where $\tilde{c}$ is a constant depending on the model of computation.*

*Furthermore, if the complexity of $\mathcal{A}$ is $T$, then the complexity of $\mathcal{B}$ is $T + c''(q_f + q_s)$, where $c''$ is another constant depending on the model of computation.*

*Proof.* Let $\mathbf{Exp}^0$ denote the original experiment in the EUF-CMA security model. In this experiment $\mathcal{A}$ is the adversary that makes an existential forgery for the Stern signature scheme using the random oracle $F$ and signing oracle $Sign$. $\mathcal{A}$ can make at most $q_f$ queries to $F$ and at most $q_s$ queries to $Sign$.

```
$\mathbf{Exp}^0(\mathcal{A}) = \mathbf{Exp}^{\text{EUF-CMA}}_{Stern}(\mathcal{A})$
1 : $s \xleftarrow{\mathcal{U}} \{\{0,1\}^n : \mathrm{wt}(s) = \omega\}$
2 : $y \leftarrow Hs^T$
3 : $\mathcal{L} \leftarrow \varnothing$
4 : $\Pi^F \leftarrow \varnothing$
5 : $(m, c\|r) \leftarrow\$ \mathcal{A}^{Sign,F}(y)$
6 : if $m \in \mathcal{L}$ : return $0$
7 : return $Stern.Ver(y, m, c\|r)$
```

```
Oracle $F(\alpha)$
1 : if $\alpha \in \Pi^F : \beta \leftarrow \Pi^F(\alpha)$
2 : else
3 :     $\beta \xleftarrow{\mathcal{U}} \{0,1,2\}^\delta$
4 :     $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
5 : return $\beta$
```

```
Oracle $Sign(s, m)$
1 : foreach $0 \leqslant i < \delta$ :
2 :     $u_i \xleftarrow{\mathcal{U}} \{0,1\}^n, \sigma_i \xleftarrow{\mathcal{U}} S_n$
3 :     $c_{i,0} \leftarrow h(\sigma_i \| Hu_i^T)$
4 :     $c_{i,1} \leftarrow h(\sigma_i(u_i))$
5 :     $c_{i,2} \leftarrow h(\sigma_i(u_i \oplus s))$
6 :     $c_i \leftarrow c_{i,0}\|c_{i,1}\|c_{i,2}$
7 : $c \leftarrow c_0\|\ldots\|c_{\delta-1}$
8 : $b \leftarrow F(m\|c)$
9 : foreach $0 \leqslant i < \delta$ :
10 :     if $b_i = 0 : r_i \leftarrow \sigma_i\|u_i$
11 :     if $b_i = 1 : r_i \leftarrow \sigma_i\|(u_i \oplus s)$
12 :     if $b_i = 2 : r_i \leftarrow \sigma_i(u_i)\|\sigma_i(s)$
13 : $r \leftarrow r_0\|\ldots\|r_{\delta-1}$
14 : $\mathcal{L} \leftarrow \mathcal{L} \cup \{m\}$
15 : return $c\|r$
```

$$\mathsf{Adv}^{\text{EUF-CMA}}_{Stern}(\mathcal{A}) := \Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1].$$

The experiment $\mathbf{Exp}^1$ is a modification of $\mathbf{Exp}^0$ obtained by introducing sets $\Pi^S, \Pi \subset \{0,1\}^* \times \{0,1,2\}^\delta$. $\Pi^S$ is filled while communicating with the oracle $Sign$ and $\Pi = \Pi^F \cup \Pi^S$.

Modifications of algorithms $F$ and $Sign$ do not affect the distributions of their outputs, therefore,

$$\Pr[\mathbf{Exp}^0(\mathcal{A}) \Rightarrow 1] = \Pr[\mathbf{Exp}^1(\mathcal{A}) \Rightarrow 1].$$

The experiment $\mathbf{Exp}^2$ differs from $\mathbf{Exp}^1$ only in algorithm $Sign$. Now it does not use the secret key, and the result is formed by a random vector $b$.

We show that the distributions of outputs $c\|r$ of the algorithm $Sign$ in experiments $\mathbf{Exp}^1$ and $\mathbf{Exp}^2$ are indistinguishable if the condition on the line 23 was not satisfied. If we show that the distribution of each such part $c_i\|r_i = c_{i,0}\|c_{i,1}\|c_{i,2}\|r_{i,0}\|r_{i,1}$ for $i = 0, \ldots, \delta - 1$ in $\mathbf{Exp}^2$ coincides with the distribution of the corresponding part in $\mathbf{Exp}^1$, then the distributions of the signatures also coincide.

Further we will consider arguments of the hash function $h$ corresponding to $c_{i,j}$ instead of the values themselves. The reason for it is the fact that if distributions of variables $\xi$ and $\eta$ coincide, then distributions of variables $h(\xi)$ and $h(\eta)$ also coincide. Indeed,

$$\Pr[h(\xi) = a] = \Pr[\xi \in h^{-1}(a)] = \Pr[\eta \in h^{-1}(a)] = \Pr[h(\eta) = a].$$

$\mathbf{Exp}^1(\mathcal{A})$

```
1 :  $s \xleftarrow{\mathcal{U}} \{\{0,1\}^n : \mathrm{wt}(s) = \omega\}$
2 :  $y \leftarrow Hs^T$
3 :  $\mathcal{L} \leftarrow \varnothing$
4 :  $(\Pi^F, \Pi^S) \leftarrow (\varnothing, \varnothing)$
5 :  $\Pi \leftarrow \Pi^F \cup \Pi^S$
6 :  $(m, c\|r) \leftarrow\$ \mathcal{A}^{Sign,F}(y)$
7 :  if $m \in \mathcal{L}$ : return $0$
8 :  return $Stern.Ver(y, m, c\|r)$
```

Oracle $F(\alpha)$

```
1 :  if $(\alpha, \cdot) \in \Pi$ : return $\Pi(\alpha)$
2 :  $\beta \xleftarrow{\mathcal{U}} \{0,1,2\}^\delta$
3 :  $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
4 :  $\Pi \leftarrow \Pi^F \cup \Pi^S$
5 :  return $\beta$
```

Oracle $Sign(s, m)$ ($\mathbf{Exp}^1$)

```
1 :  foreach $0 \leqslant i < \delta$ :
2 :      $u_i \xleftarrow{\mathcal{U}} \{0,1\}^n, \sigma_i \xleftarrow{\mathcal{U}} S_n$
3 :      $c_{i,0} \leftarrow h(\sigma_i \| Hu_i^T)$
4 :      $c_{i,1} \leftarrow h(\sigma_i(u_i))$
5 :      $c_{i,2} \leftarrow h(\sigma_i(u_i \oplus s))$
6 :      $c_i \leftarrow c_{i,0} \| c_{i,1} \| c_{i,2}$
7 :  $c \leftarrow c_0 \| \dots \| c_{\delta-1}$
8 :  if $(m\|c, \cdot) \in \Pi$ : $b \leftarrow \Pi(m\|c)$
9 :  else
10 :     $b \xleftarrow{\mathcal{U}} \{0,1,2\}^\delta$
11 :     $\Pi^S \leftarrow \Pi^S \cup \{(m\|c, b)\}$
12 :     $\Pi \leftarrow \Pi^F \cup \Pi^S$
13 : foreach $0 \leqslant i < \delta$ :
14 :     if $b_i = 0$ : $r_i \leftarrow \sigma_i \| u_i$
15 :     if $b_i = 1$ : $r_i \leftarrow \sigma_i \| (u_i \oplus s)$
16 :     if $b_i = 2$ : $r_i \leftarrow \sigma_i(u_i) \| \sigma_i(s)$
17 : $r \leftarrow r_0 \| \dots \| r_{\delta-1}$
18 : $\mathcal{L} \leftarrow \mathcal{L} \cup \{m\}$
19 : return $c\|r$
```

Oracle $Sign(m)$ ($\mathbf{Exp}^2$)

```
1 :  $s' \xleftarrow{\mathcal{U}} \{\{0,1\}^n : \mathrm{wt}(s') = \omega\}$
2 :  foreach $0 \leqslant i < \delta$ :
3 :      $b_i \xleftarrow{\mathcal{U}} \{0,1,2\}$
4 :      $u'_i \xleftarrow{\mathcal{U}} \{0,1\}^n, \sigma'_i \xleftarrow{\mathcal{U}} S_n,$
5 :      if $b_i = 0$ :
6 :          $c_{i,0} \leftarrow h(\sigma'_i \| Hu_i'^T)$
7 :          $c_{i,1} \leftarrow h(\sigma'_i(u'_i))$
8 :          $c_{i,2} \leftarrow h(\sigma'_i(u'_i \oplus s'))$
9 :          $r_i \leftarrow \sigma'_i \| u'_i$
10 :     if $b_i = 1$ :
11 :         $c_{i,0} \leftarrow h(\sigma'_i \| (Hu_i'^T \oplus y))$
12 :         $c_{i,1} \leftarrow h(\sigma'_i(s'))$
13 :         $c_{i,2} \leftarrow h(\sigma'_i(u'_i))$
14 :         $r_i \leftarrow \sigma'_i \| u'_i$
15 :     if $b_i = 2$ :
16 :         $c_{i,0} \leftarrow h(\sigma'_i \| H(u'_i \oplus s')^T)$
17 :         $c_{i,1} \leftarrow h(\sigma'_i(u'_i \oplus s'))$
18 :         $c_{i,2} \leftarrow h(\sigma'_i(u'_i))$
19 :         $r_i \leftarrow \sigma'_i(u'_i \oplus s') \| \sigma'_i(s')$
20 :     $c_i \leftarrow c_{i,0} \| c_{i,1} \| c_{i,2}$
21 : $c \leftarrow c_0 \| \dots \| c_{\delta-1}$
22 : $r \leftarrow r_0 \| \dots \| r_{\delta-1}$
23 : if $(m\|c, \cdot) \in \Pi^F$ : abort
24 : $\Pi^S \leftarrow \Pi^S \cup \{(m\|c, b)\}$
25 : $\Pi \leftarrow \Pi^F \cup \Pi^S$
26 : $\mathcal{L} \leftarrow \mathcal{L} \cup \{m\}$
27 : return $c\|r$
```

In case $b_i = 0$ for an external observer the secret key $s$ is a random variable. And other values are randomly selected in the same way as in the original protocol. So distributions, obviously, coincide.

If $b_i = 1$, then the probability that in $\mathbf{Exp}^1$ the string $c_i \| r_i$ equals $a_1 \| a_2 \| a_3 \| a_4 \| a_5 \| a_6$ is

$$\begin{aligned} &P_{a_1,a_2,a_3,a_4,a_5,a_6} = \\ &= \Pr\left[\sigma_i = a_1, Hu_i^T = a_2, \sigma_i(u_i) = a_3, \sigma_i(u_i \oplus s) = a_4, \sigma_i = a_5, u_i \oplus s = a_6\right] = \\ &\qquad = \mathbb{I}\left[a_1 = a_5,\ H(a_6 \oplus s)^T = a_2, a_1(a_6) = a_4\right] \times \\ &\qquad\qquad \times \Pr\left[\sigma_i = a_1, s = a_1^{-1}(a_3) \oplus a_6, u_i = a_1^{-1}(a_3)\right], \end{aligned}$$

where $\mathbb{I}\left[\theta\right]$ is the indicator of expression $\theta$. In $\mathbf{Exp}^2$ this probability is

$$\begin{aligned} &\widehat{P}_{a_1,a_2,a_3,a_4,a_5,a_6} = \\ &= \Pr\left[\sigma_i' = a_1, Hu_i'^T \oplus y = a_2, \sigma_i'(s') = a_3, \sigma_i'(u_i') = a_4, \sigma_i' = a_5, u_i' = a_6\right] = \\ &= \mathbb{I}\left[a_1 = a_5, Ha_6^T \oplus y = a_2, a_1(a_6) = a_4\right] \Pr\left[\sigma_i' = a_1, s' = a_1^{-1}(a_3), u_i' = a_6\right]. \end{aligned}$$

As far as $H(a_6 \oplus s)^T = Ha_6^T \oplus y$, indicators of these two expressions coincide.

Let us evaluate the probabilities. Note that since all random variables are selected independently, the probability of the conjunction of events equals to the product of their probabilities. So we can find them separately.

$$\begin{aligned} &\Pr[\sigma_i = a_1] = \Pr[\sigma_i' = a_1] = \frac{1}{n!}, \\ &\Pr[s = a_1^{-1}(a_3) \oplus a_6 = a'] = \frac{1}{2^n}, \\ &\Pr[u_i = a_1^{-1}(a_3) = a''] = \frac{1}{2^n}, \\ &\Pr[s' = a_1^{-1}(a_3) = a'''] = \frac{1}{2^n}, \\ &\Pr[u_i' = a_5] = \frac{1}{2^n} \end{aligned}$$

for any constants $a', a''$ and $a'''$. Then

$$\begin{aligned} &\Pr\left[\sigma_i = a_1, s = a_1^{-1}(a_3) \oplus a_6, u_i = a_1^{-1}(a_3)\right] = \\ &\qquad = \Pr\left[\sigma_i' = a_1, s' = a_3, u_i' = a_5\right] = \frac{1}{n! 2^{2n}} \end{aligned}$$

and distributions are indistinguishable.

Finally, if $b_i = 2$, then the similar probability in $\mathbf{Exp}^1$ is

$$\begin{aligned} &P_{a_1,a_2,a_3,a_4,a_5,a_6} = \\ &= \Pr\left[\sigma_i = a_1, Hu_i^T = a_2, \sigma_i(u_i) = a_3, \sigma_i(u_i \oplus s) = a_4, \sigma_i(u_i) = a_5, \sigma_i(s) = a_6\right] = \\ &\qquad = \mathbb{I}\left[a_3 = a_5, a_3 \oplus a_6 = a_4, H(a_1^{-1}(a_3))^T = a_2\right] \times \\ &\qquad\qquad \times \Pr\left[\sigma_i = a_1, u_i = a_1^{-1}(a_3), s = a_1^{-1}(a_6)\right] \end{aligned}$$

and in $\mathbf{Exp}^2$ it is

$$\widehat{P}_{a_1,a_2,a_3,a_4,a_5,a_6} = \Pr\left[\sigma'_i = a_1, H(u'_i \oplus s')^T = a_2, \sigma'_i(u'_i \oplus s') = a_3, \sigma'_i(u'_i) = a_4,\right.$$
$$\left.\sigma'_i(u'_i \oplus s') = a_5, \sigma'_i(s') = a_6\right) = \mathbb{I}\left[a_3 = a_5, a_4 \oplus a_6 = a_3, H(a_1^{-1}(a_3))^T = a_2\right] \times$$
$$\times \Pr\left[\sigma'_i = a_1, u'_i = a_1^{-1}(a_4), s' = a_1^{-1}(a_6)\right].$$

Reasoning similar to above

$$\Pr\left[\sigma_i = a_1, u_i = a_1^{-1}(a_3), s = a_1^{-1}(a_6)\right] =$$
$$= \Pr\left[\sigma'_i = a_1, u'_i = a_1^{-1}(a_4), s' = a_1^{-1}(a_6)\right] = \frac{1}{n! 2^{2n}}$$

and distributions coincide.

The check on the line 23 corresponds to the case when the value $c$, created while generating a signature for the message $m$, already exists in the set $\Pi^F$. Then let $E_\kappa$ denote the event that strings $c$ and $\hat{c}$ coincide, where $c$ is the $\kappa$-th query to the signing oracle and $\hat{c}$ is one of queries made to the oracle $F$. Also let us introduce values $d_{i,j}$ and $\hat{d}_{i,j}$ that are inputs of the hash function $h$ resulting in values $c_{i,j}$ and $\hat{c}_{i,j}$:

$$c_{i,j} = h(d_{i,j}),\ \hat{c}_{i,j} = h(\hat{d}_{i,j}).$$

Note that $c_{i,j}$ is generated according to the $\mathbf{Exp}^2$ *Sign* algorithm and $\hat{c}_{i,j}$ — to the $\mathbf{Exp}^1$ *Sign* algorithm.

Let us find the probability of the event that substrings $c_i$ and $\hat{c}_i$ coincide. Denote $p = \Pr[\exists j \in \{0,1,2\} : c_{i,j} = \hat{c}_{i,j} \wedge d_{i,j} \neq \hat{d}_{i,j}]$. To evaluate $p$, note that there exists a suboptimal algorithm that solves $Coll(h)$ problem choosing at random three pairs of strings corresponding to $\mathbf{Exp}^2$ and $\mathbf{Exp}^1$ *Sign* algorithms and comparing their hash values. The complexity of this algorithm is $\tilde{c}p^{-1}$, where $\tilde{c}$ is the complexity of one iteration. Then $p \leqslant \tilde{c}T_{Coll}^{-1}$, where $T_{Coll}$ is the complexity of an optimal algorithm solving $Coll(h)$ problem with the probability of success at least $1 - 1/e$.

Otherwise, for all $j \in \{0,1,2\}$ holds $d_{i,j} = \hat{d}_{i,j}$. Let $b_i = 0$. Then

$$\Pr[\forall j \in \{0,1,2\} : d_{0,j} = \hat{d}_{0,j}] = \Pr[\sigma = \sigma'] \Pr[u = u'] \Pr[s = s'] = \frac{1}{2^{2n} n!}.$$

The same may be also shown for $b_i = 1$ and $b_i = 2$.

After the interaction with the oracle $F$ at most $q_f$ strings can be obtained. Hence, a pair of strings can be selected from them in $q_f^2$ ways. Then

$$\Pr[E_\kappa] \leqslant q_f^2 \left(\max\left\{\frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n} n!}\right\}\right)^\delta.$$

Now let us find the probability of the event $G$, that corresponds to the case when the condition on the line 23 was never satisfied.

$$\Pr[G] = 1 - \Pr[E_1 \vee E_2 \vee \cdots \vee E_{q_s}] \geqslant 1 - \sum_{\kappa=1}^{q_s} \Pr[E_\kappa] \geqslant$$
$$\geqslant 1 - \sum_{\kappa=1}^{q_s} q_f^2 \left( \max\left\{ \frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n} n!} \right\} \right)^{\delta} = 1 - q_f^2 q_s \left( \max\left\{ \frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n} n!} \right\} \right)^{\delta}.$$

Since

$$\Pr[\mathbf{Exp}^1(\mathcal{A}) \Rightarrow 1] = \Pr[\mathbf{Exp}^1(\mathcal{A}) \Rightarrow 1 \wedge G] + \Pr[\mathbf{Exp}^1(\mathcal{A}) \Rightarrow 1 \wedge \overline{G}] \leqslant$$
$$\leqslant \Pr[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1] + \Pr[\overline{G}],$$

then

$$\Pr[\mathbf{Exp}^1(\mathcal{A}) \Rightarrow 1] - \Pr[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1] \leqslant \Pr[\overline{G}] = q_f^2 q_s \left( \max\left\{ \frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n} n!} \right\} \right)^{\delta}.$$

Now basing on the adversary $\mathcal{A}$ we construct an adversary $\mathcal{B}$ that makes an existential forgery in the NMA model. It simulates oracles $F$ and $Sign$ using algorithms $SimF$ and $SimSign$. The algorithm $SimSign$ repeats the algorithm $Sign$ in the experiment $\mathbf{Exp}^2$. The oracle $F^*$ is the random oracle of $\mathcal{B}$.

```
$\mathcal{B}^{F^*}(y)$
1 :  $\mathcal{L} \leftarrow \varnothing$
2 :  $(m, c\|r) \leftarrow\$ \mathcal{A}^{SimSign, SimF}(y)$
3 :  **if** $m \in \mathcal{L}$ : **return** 0
4 :  **return** $(m, c\|r)$
```

```
Oracle $SimF(\alpha)$
1 :  **return** $F^*(\alpha)$
```

Line 3 ensures that the forgery was built for a new message that had never been the input of the oracle $SimSign$. Hence $\mathcal{A}$ used the random oracle $F^*$ and the forgery is valid for $\mathcal{B}$. Note that the case $m \in \mathcal{L}$ is similarly processed in $\mathbf{Exp}^2(\mathcal{A})$. Thus

$$\Pr[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1] = \Pr[\mathbf{Exp}^{\text{EUF-NMA}}_{Stern}(\mathcal{B}) \Rightarrow 1].$$

Consequently,

$$\mathsf{Adv}^{\text{EUF-NMA}}_{Stern}(\mathcal{B}) \geqslant \mathsf{Adv}^{\text{EUF-CMA}}_{Stern}(\mathcal{A}) - q_f^2 q_s \left( \max\left\{ \frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n} n!} \right\} \right)^{\delta}.$$

The adversary $\mathcal{B}$ runs $\mathcal{A}$ and simulates $q_f$ queries to the oracle $F$ and $q_s$ queries to the oracle $Sign$. Hence, its complexity is $T + c''(q_f + q_s)$.

**Corollary 1.** *Let $\mathcal{A}$ be an adversary in the EUF-CMA model for Stern signature scheme, making at most $q_f$ queries to the hashing oracle $F$ and at most $q_s$ queries* *to the signing oracle Sign. Then*

$$\mathsf{Adv}^{\text{EUF-CMA}}_{Stern}(\mathcal{A}) \leqslant$$
$$\leqslant 6q_f \sqrt[3]{\frac{\delta^2(T+\tilde{\tilde{c}}(2q_f+q_s))}{\min\{T_{SD},T_{Coll}\}}} + \left(\frac{2}{3}\right)^{\delta} \cdot (q_f+1) + q_f^2 q_s \left(\max\left\{\frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n}n!}\right\}\right)^{\delta},$$

*where $T$ is the maximum possible time complexity of $\mathcal{A}$, $T_{SD}$ and $T_{Coll}$ are complexities of optimal algorithms solving $SD(H,y,\omega)$ and $Coll(h)$ problems with probabilities of success at least $1-\frac{1}{e}$, $\tilde{c}$ and $\tilde{\tilde{c}}$ are constants depending on the model of computation.*

*Proof.* The complexity $T'$ of an adversary in the 1-EUF-NMA model from Theorems 1–3 is $T+\tilde{\tilde{c}}(2q_f+q_s)$, where $T$ is the complexity of an adversary in the EUF-CMA model and $\tilde{\tilde{c}} = \max\{c', c''\}$. Also

$$\mathsf{Adv}^{\text{EUF-NMA}}_{Stern}(\mathcal{B}) \leqslant 6q_f \cdot \sqrt[3]{\frac{\delta^2 T'}{\min\{T_{SD},T_{Coll}\}}} + \left(\frac{2}{3}\right)^{\delta} \cdot (q_f+1),$$

$$\mathsf{Adv}^{\text{EUF-CMA}}_{Stern}(\mathcal{A}) \leqslant \mathsf{Adv}^{\text{EUF-NMA}}_{Stern}(\mathcal{B}) + q_f^2 q_s \left(\max\left\{\frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n}n!}\right\}\right)^{\delta} \leqslant$$
$$\leqslant 6q_f \cdot \sqrt[3]{\frac{\delta^2 T'}{\min\{T_{SD},T_{Coll}\}}} + \left(\frac{2}{3}\right)^{\delta} \cdot (q_f+1) + q_f^2 q_s \left(\max\left\{\frac{\tilde{c}}{T_{Coll}}, \frac{1}{2^{2n}n!}\right\}\right)^{\delta}.$$

## 5 Parameters

In this section we mention different constraints on the signature parameters and introduce some parameter sets.

### 5.1 Choosing general parameters

Stern showed [19] that his identification scheme can be forged with probability equal to 2/3. Similar reasoning allows to assert that an adversary can build a forgery of the signature without knowing the secret key with the probability $(2/3)^{\delta}$. Thus parameter $\delta$ should be chosen to satisfy the condition:

$$\left(\frac{2}{3}\right)^{\delta} < 2^{-\lambda},$$

where $\lambda$ is the security parameter.

The best known method to find collisions for hash functions used in practice is based on the birthday paradox and has complexity about $2^{\ell/2}$, where $\ell$ is the length of the hash value. Since we want to maximize this complexity, it is worth using hash functions with the maximal $\ell$. This can be, for example, such well-known functions as the American standard SHA3-512 or the Russian standard

Streebog-512, in which $\ell = 512$ bits. For them $T_{Coll} \approx 2^{256}$ bit operations. Further we suppose that hash function Streebog is used.

We assume that the maximal complexity $T$ of the adversary does not exceed $2^{80}$ bit operations. Each of $q_f$ queries to the hashing oracle consists in evaluation of the hash function of messages, which in practice can be several megabytes in size. According to [30] the complexity of Streebog hash function is about $2^{25}$ CPU cycles or more than $2^{30}$ bit operations. Signing oracle has to evaluate Streebog hash function at least three times, thus $q_s \leqslant q_f$. As a result, an adversary with complexity $T$ is able to do no more that $2^{50}$ queries to each oracle.

Then in order to maximize the value of $\min\{T_{SD}, T_{Coll}\}$ it is necessary to choose the length of the code so that $T_{SD} \geqslant T_{Coll}$. From the fact [29] that $T_{SD} \approx 2^{0.0885n}$ a lower estimate for the length of the code $n$ can be found. We choose $n$ so as the advantage of the adversary, estimated above, is not greater than 1/2. We choose the code dimension as $k = n/2$ and require the code to lie on the Varshamov–Gilbert boundary:

$$\frac{k}{n} = 1 - H\left(\frac{\omega}{n}\right),$$

whence it follows that $\omega \approx 0.11n$.

Note that the last term is negligible and may not be taken into account in calculations.

## 5.2 Public data and signature sizes

The public key is a vector $y$ of $n$ bits in size. The public parameter $H$ is an $(n-k) \times n$-matrix that can be stored as $k(n-k)$ bits in systematic form.

The size of $c$ is $3\delta\ell$ bits. The maximal size of $r_i$ is $n + n\log_2 n$ bits and, accordingly, size of $r$ can be upper estimated as $\delta(n + n\log_2 n)$ bits. So the total size of the signature can be estimated from above as $\delta \cdot (3\ell + n + n\log_2 n)$ bits.

## 5.3 Example parameter sets

This part contains an example of the signature parameter set selected in accordance with the conditions above. The adversary's advantage in this case equals approximately 0.39.

| $\lambda$ | $n$ | $k$ | $\omega$ | $\delta$ | $\ell$ | $H$, MB | $y$, KB | $\zeta$, MB |
|---|---|---|---|---|---|---|---|---|
| 80 | 2896 | 1448 | 318 | 137 | 512 | 0.25 | 0.35 | 0.62 |

The fact that the first term depends linearly on the parameter $q_f$ does not allow one to get parameter sets for a much higher level of security. Nevertheless, we presume that our estimate is rather rough and in fact even the introduced parameter set provides more security than 80 bit.

## 6 Conclusion

The paper presents the security bounds for a digital signature based on the Stern identification protocol. We connect the security of the scheme with the hardness of syndrome decoding and hash function collision finding problems. Basing on the security notions we introduce a parameter set providing 80-bit security of the signature. As a direction for further research, we consider the extension of the security proof to a model with quantum access to the random oracle.

## References

1. Shor, P. V.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Journal on Computing **26**(5), 1484–1509 (1997)
2. NIST PQC Call for Proposals, https://csrc.nist.gov/Projects/post-quantum-cryptography/Post-Quantum-Cryptography-Standardization/Call-for-Proposals.
3. Lee, W., Kim, Y.-S., Lee, Y.-W., No, J.-S.: Post quantum signature scheme based on modified Reed-Muller code pqsigRM. First round submission to the NIST post-quantum cryptography call (2017)
4. Fukushima, K., Roy, P. S., Xu, R., Kiyomoto, S., Morozov, K., Takagi, T.: Supporting Documentation of RaCoSS (Random Code-based Signature Scheme). First round submission to the NIST post-quantum cryptography call (2017)
5. Aragon, N., Gaborit, P., Hauteville, A., Ruatta, O., Zemor, G.: RankSign — a signature proposal for the NIST's call. First round submission to the NIST post-quantum cryptography call (2017)
6. Debris-Alazard, T., Tillich, J.-P.: Two Attacks on Rank Metric Code-Based Schemes: RankSign and an IBE Scheme. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, LNCS, vol. 11272, pp. 62–92. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03326-2_3
7. Lee, Y., Lee, W., Kim, Y. S., No, J.-S.: Modified pqsigRM: RM Code-Based Signature Scheme. IEEE Access **8**, 177506–177518 (2020)
8. Roy, P. S., Morozov, K., Fukushima, K., Kiyomoto, S., Takagi, T.: Code-Based Signature Scheme without Trapdoors. IEICE Tech. Rep. **118**(151), 17–22 (2018)
9. Xagawa, K.: Practical Attack on RaCoSS-R. Cryptology ePrint Archive, Report 2018/831 (2018), http://eprint.iacr.org/
10. Kabatianskii, G., Krouk, E., Smeets, B.: A digital signature scheme based on random error-correcting codes. In: Darnell, M. (eds.) CRYPTOGRAPHY AND CODING 1997, LNCS, vol. 1355, pp. 161–167. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0024461
11. Cayrel, P.-L., Otmani, A., Vergnaud, D.: On Kabatianskii-Krouk-Smeets Signatures. In: Carlet, C., Sunar, B. (eds) WAIFI 2007, LNCS, vol. 4547, pp. 237–252. Springer, Heidelberg (2007). doi10.1007/978-3-540-73074-3_18
12. Stern, J.: Can one design a signature scheme based on error-correcting codes? In: Pieprzyk, J., Safavi-Naini, R. (eds.) ASIACRYPT 1994, LNCS, vol. 917, pp. 424–426. Springer, Heidelberg (1995). doi10.1007/BFb0000454
13. Courtois, N., Finiasz, M., Sendrier, N.: How to achieve a McEliece-based digital signature scheme. In: Boyd, C. (eds.) ASIACRYPT 2001, LNCS, vol. 2248, pp. 157–174. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_10

14. McEliece, R. J.: A public-key cryptosystem based on algebraic coding theory. DSN Progress Report **4244**, 114–116 (1978)
15. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Problems of Control and Information Theory **15**(2), 159–166 (1986)
16. Dallot, L.: Towards a concrete security proof of Courtois, Finiasz and Sendrier signature scheme. In: Lucks, S., Sadeghi, A.-R., Wolf C. (eds.) WEWoRC 2007, LNCS, vol. 4945, pp. 65–77. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88353-1_6
17. Debris-Alazard, T., Sendrier, N., Tillich, J.-P.Wave: A New Family of Trapdoor One-Way Preimage Sampleable Functions Based on Codes. In: Galbraith, S., Moriai, S. (eds.) ASIACRYPT 2019, LNCS, vol. 11921, pp.21–51. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34578-5_2
18. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A. M. (eds.) CRYPTO 1986, LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12
19. Stern, J.: A new identification scheme based on syndrome decoding. In: Stinson, D. R. (eds.) CRYPTO 1993, LNCS, vol. 773, pp. 13–21. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_2
20. Jain, A., Krenn, S., Pietrzak, K., Tentes, A.: Commitments and Efficient Zero-Knowledge Proofs from Learning Parity with Noise. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012, LNCS, vol. 7658, pp. 663–680. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_40
21. 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 (2010). https://doi.org/10.1007/978-3-642-19574-7_12
22. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012, LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_43
23. Aragon, N., Blazy, O., Gaborit, P., Hauteville, A., Zémor, G. Durandal: A Rank Metric Based Signature Scheme. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, LNCS, vol. 11478, pp. 728–758. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_25
24. Overbeck, R., Sendrier, N.: Code-based cryptography. In: Bernstein, D. J., Buchmann, J., Dahmen, E. (eds.) Post-Quantum Cryptography, pp. 95–146. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-88702-7_4
25. Roy, P. S., Morozov, K., Fukushima, K., Kiyomoto, S.: Evaluation of Code-based Signature Schemes. Cryptology ePrint Archive, Report 2019/544 (2019), https://eprint.iacr.org/
26. El Yousfi Alaoui, S. M., Cayrel, P.-L., El Bansarkhani, R., Hoffmann, G.: Code-Based Identification and Signature Schemes in Software. In: Cuzzocrea, A., Kittl, C., Simos, D. E., Weippl, E., Xu, L. (eds.) CD-ARES 2013, LNCS, vol. 8128, pp. 122–136. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40588-4_9
27. Pointcheval, D., Stern J. Security proofs for signature schemes. EUROCRYPT, LNCS, vol. 1070, pp. 387–398. Springer, Berlin, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_33
28. Berlekamp, E., McEliece, R., van Tilborg, H.: On the inherent intractability of certain coding problems (Corresp.). IEEE Transactions on Information Theory **24**(3), 384–386 (1978)

29. Both, L., May, A.: Decoding Linear Codes with High Error Rate and Its Impact for LPN Security. In: Lange, T., Steinwandt, R. (eds.) PQCrypto 2018, LNCS, vol. 10786, pp. 25–46. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-79063-3_2
30. Lebedev, P. A.: Comparison of old and new cryptographic hash function national standards of Russian Federation on CPUs and NVIDIA GPUs. Mat. Vopr. Kriptogr. **4**(2), pp. 73–80 (2013). https://doi.org/10.4213/mvk84