УДК 519.719.2 DOI 10.17223/20710410/57/5

THE SECURITY OF THE CODE-BASED SIGNATURE SCHEME BASED ON THE STERN IDENTIFICATION PROTOCOL

V. V. Vysotskaya*,**, I. V. Chizhov*,**,***

* *JSC "NPK Kryptonite", Moscow, Russia*
** *Lomonosov Moscow State University, Moscow, Russia*
*** *Federal Research Center "Informatics and Control" of Russian Academy of Science, Moscow, Russia*

E-mail: {v.vysotskaya, i.chizhov}@kryptonite.ru

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). Finally, we discuss the choice of the signature parameters, in particular providing 70-bit security.

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

О СТОЙКОСТИ КОДОВОЙ ЭЛЕКТРОННОЙ ПОДПИСИ НА ОСНОВЕ ПРОТОКОЛА ИДЕНТИФИКАЦИИ ШТЕРНА

В. В. Высоцкая*,**, И. В. Чижов*,**,***

* *АО НПК «Криптонит», г. Москва, Россия*
** *МГУ им М. В. Ломоносова, г. Москва, Россия*
*** *ФИЦ ИУ РАН, г. Москва, Россия*

Представлено полное описание схемы электронной подписи на основе схемы идентификации Штерна. Доказана стойкость схемы относительно построения экзистенциальной подделки при атаке с выбором сообщений (EUF-CMA) в модели со случайным оракулом. Обсуждается выбор параметров подписи, в частности обеспечивающий стойкость, равную 70 битам.

Ключевые слова: *постквантовая криптография, кодовая криптография, электронная подпись, схема Штерна, преобразование Фиата — Шамира, доказуемая стойкость, EUF-CMA-стойкость.*

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 includes 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 [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, due to the inner decoding procedure with extremely small probability of success on a random input, the signature generation algorithm has 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 that 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 considered 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+. In turn, the security of Stern-type schemes is based only on NP-hard problems and the hardness of finding a collision of the underlying hash function.

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 provided 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 may be proved. The authors mention there the applicability of the Forking lemma to the Stern signature scheme proof. However, this fact was not proven neither in this paper nor elsewhere later.

In the paper we provide a complete description of the signature scheme 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. The set of all binary strings of length n we denote by $\{0,1\}^n$ and the set of strings of arbitrary length by $\{0,1\}^*$. We denote the symmetric group of order n by S_n, i.e., the group of all permutations of elements of the set $\{1,\ldots,n\}$. 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 SD(H, y, ω). **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 $He^{\mathrm{T}} = y^{\mathrm{T}}$.

Problem Coll(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 $\mathcal{O}\big(2^{0.0885n}\big)$ bit operations [29]. The complexity of the 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 $\mathcal{O}\big(2^{\ell/2}\big)$.

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

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

Fig. 1. Stern identification scheme

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 $\Sigma = (\mathrm{KeyGen}, \mathrm{Sig}, \mathrm{Ver})$ of (possibly probabilistic) polynomial time algorithms, where

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

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 δ that is determined by the security parameter λ.

Stern.KeyGen()

$s \xleftarrow{\mathcal{U}} \{x \in \{0,1\}^n : \mathrm{wt}(x) = \omega\}$
$y \leftarrow Hs^T$
return (y, s)

Stern.Sig(s, m)

foreach $0 \leqslant i < \delta$:
  $u_i \xleftarrow{\mathcal{U}} \{0,1\}^n, \sigma_i \xleftarrow{\mathcal{U}} S_n$
  $c_{i,0} \leftarrow h(\sigma_i \| Hu_i^T)$
  $c_{i,1} \leftarrow h(\sigma_i(u_i))$
  $c_{i,2} \leftarrow h(\sigma_i(u_i \oplus s))$
  $c_i \leftarrow c_{i,0} \| c_{i,1} \| c_{i,2}$
$c \leftarrow c_0 \| \ldots \| c_{\delta-1}$
$b \leftarrow f(m \| c)$
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$(y, m, (c\|r))$

$b \leftarrow f(m\|c)$
foreach $0 \leqslant i < \delta$:
  if $\big[b_i = 0\big] \wedge \Big[\big[c_{i,0} \neq h(r_{i,0}\|Hr_{i,1}^T)\big] \vee \big[c_{i,1} \neq h(r_{i,0}(r_{i,1}))\big]\Big]$:
    return 0
  if $\big[b_i = 1\big] \wedge \Big[\big[c_{i,0} \neq h(r_{i,0}\|(Hr_{i,1}^T \oplus y))\big] \vee \big[c_{i,2} \neq h(r_{i,0}(r_{i,1}))\big]\Big]$:
    return 0
  if $\big[b_i = 2\big] \wedge \Big[\big[c_{i,1} \neq h(r_{i,0})\big] \vee \big[c_{i,2} \neq h(r_{i,0} \oplus r_{i,1})\big] \vee \big[\mathrm{wt}(r_{i,1}) \neq \omega\big]\Big]$:
    return 0
return 1

To estimate the scheme security, we construct experiments where the adversary is represented by a probabilistic polynomial-time Turing machine. The notation $\mathbf{Exp} \Rightarrow b$ 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 $x \leftarrow\!\$\ A(\ldots)$.

To model a random oracle $F : \{0,1\}^* \to \{0,1,2\}^\delta$, we use lazy sampling. We introduce the set Π^F containing pairs of the 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 Π^F contains not more than one pair (α, β) for each α, then $\Pi^F(\alpha)$ denotes either β if $(\alpha, \beta) \in \Pi^F$ or special value $\perp$ if there is no such a pair.

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

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

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

$\mathbf{Exp}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{A})$
$(pk, sk) \leftarrow\!\$\ \text{Stern.KeyGen}()$
$\Pi^F \leftarrow \varnothing$
$(m, \zeta) \leftarrow\!\$\ \mathcal{A}^F(pk)$
return Stern.Ver(pk, m, ζ)

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

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

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

where the experiment $\mathbf{Exp}^{\text{EUF-CMA}}_{\text{Stern}}$ is defined as follows:

$\mathbf{Exp}^{\text{EUF-CMA}}_{\text{Stern}}(\mathcal{A})$

1 : $(pk, sk) \leftarrow\$ \text{Stern.KeyGen}()$
2 : $\mathcal{L} \leftarrow \varnothing$
3 : $\Pi^F \leftarrow \varnothing$
4 : $(m, \zeta) \leftarrow\$ \mathcal{A}^{\text{Sign},F}(pk)$
5 : **if** $m \in \mathcal{L}$: **return** 0
6 : **return** $\text{Stern.Ver}(pk, m, \zeta)$

Oracle $F(\alpha)$

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

Oracle $\text{Sign}(m)$

1 : $\zeta \leftarrow\$ \text{Stern.Sig}(sk, m)$
2 : $\mathcal{L} \leftarrow \mathcal{L} \cup \{m\}$
3 : **return** ζ

4. Security bounds

Let us give several definitions that will be needed below.

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

Definition 5. Let us call a tree ρ-dense if its density is not less than ρ.

Definition 6. We call a tree uniformly ρ-dense if each of its subtrees, excluding leaves, is a ρ-dense tree.

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

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 $\theta = \rho/\delta$. Note that until the algorithm stops (i.e. reaches the root), some leaves may have depth less than δ. However, after the algorithm completion each of the survived leaves will have depth δ.

Let us show that at each step of this algorithm the root density decreases by no more than θ. Suppose that there are $n_{i,3}$ vertices at the i-th level of the original tree T. The densities of subtrees formed by them are $\rho_{i,1}, \ldots, \rho_{i,n_{i,3}}$. If we denote the number of leaves of T by t, then

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

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 θ. Thus, the new density $\rho'_{i,j}$ may either be equal to $\rho_{i,j}$ or become 0 if $\rho_{i,j} < \theta$. So

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

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

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

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

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

So the resulting tree is uniformly θ-dense. ■

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

$$\mathsf{Adv}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{A})\leqslant \max\left\{15\sqrt[3]{\frac{\delta^2 T}{\min\{T_{SD},T_{Coll}\}}}+\left(\frac{2}{3}\right)^{\delta},\ \left(\frac{2}{3}\right)^{\delta}\left(1+2\delta\cdot 1.1^{\delta}\right)\right\},$$

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

Proof.

Denote

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

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

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

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 δ. It is determined by $\mathcal{A}$'s random tape x. Each output b of the random oracle corresponds to a certain path in the tree. If the corresponding b_i equals 0, then 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 $\mathrm{SD}(H,y,\omega)$ and $\mathrm{Coll}(h)$ 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 some adversary $\mathcal{B}$ find such vertices in the tree $T(x)$ with probability $1-1/e$.

Let the tree has 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 $r_{i,0}=\tilde{\sigma}_0$ and $r_{i,1}=\tilde{u}_0$ for $b_i=0$. For $b_i=1$, let $r_{i,0}=\tilde{\sigma}_1$ and $r_{i,1}=\tilde{w}_1$, where $\tilde{w}_1$ corresponds to $u_i\oplus s$. Finally, for $b_i=2$, let $r_{i,0}=\tilde{z}_2$ and $r_{i,1}=\tilde{t}_2$, where $\tilde{z}_2$ corresponds to $\sigma_i(u_i)$ and $\tilde{t}_2$ corresponds to $\sigma_i(s)$. Since $c_{i,0}$ can be obtained in two cases ($b_i=0$ and $b_i=1$), then

$$c_{i,0}=h(\tilde{\sigma}_0\|H\tilde{u}_0^{\mathrm{T}})=h(\tilde{\sigma}_1\|H\tilde{w}_1^{\mathrm{T}}\oplus y).$$

Hence, either collision of hash function h can be found, or $\tilde{\sigma}_0=\tilde{\sigma}_1$ and $H\tilde{u}_0^{\mathrm{T}}=H\tilde{w}_1^{\mathrm{T}}\oplus y$. Similarly, it can be shown that if no collisions were found, then $\tilde{z}_2=\tilde{\sigma}_0(\tilde{u}_0)$ and

$\tilde{z}_2 \oplus \tilde{t}_2 = \tilde{\sigma}_1(\tilde{w}_1)$. Note that as the third answer was accepted, $\tilde{t}_2$ satisfies the weight constraint. Denoting $\sigma = \tilde{\sigma}_0 = \tilde{\sigma}_1$, we have

$$\tilde{t}_2 = \tilde{z}_2 \oplus (\tilde{t}_2 \oplus \tilde{z}_2) = \sigma(\tilde{u}_0 \oplus \tilde{w}_1).$$

Therefore, $\tilde{u}_0 \oplus \tilde{w}_1$ also has the acceptable weight. Then

$$H(\tilde{u}_0 \oplus \tilde{w}_1)^{\mathrm{T}} = H\tilde{u}_0^{\mathrm{T}} \oplus H\tilde{w}_1^{\mathrm{T}} = y$$

and $\tilde{u}_0 \oplus \tilde{w}_1$ is an acceptable secret key.

Let us denote $\theta = \dfrac{\varepsilon}{2\delta}$ and 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) Randomly choose $60/\theta^2$ inputs of the random oracle and evaluate its outputs (defining the 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 solve either $\mathrm{SD}(H, y, \omega)$ or $\mathrm{Coll}(h)$ problem. Otherwise, return to Step 1.

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

Proof. Define the set X as

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

Then $\mathsf{P}[x \in X] \geqslant \varepsilon/2$.

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

$$\begin{aligned}\mathsf{P}[\mathcal{A} \Rightarrow 1] = \mathsf{P}[\mathcal{A} \Rightarrow 1 \wedge x \in X] + \mathsf{P}[\mathcal{A} \Rightarrow 1 \wedge x \notin X] \leqslant \mathsf{P}[x \in X] + \\ + \mathsf{P}[\mathcal{A} \Rightarrow 1 \wedge x \notin X] < \varepsilon/2 + \big((2/3)^\delta + \varepsilon/2\big) = (2/3)^\delta + \varepsilon.\end{aligned}$$

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 the Proposition 1 one can select a uniformly θ-dense subtree with leaves of depth δ from each such tree. Let us call this tree $T_1(x)$.

For any index i, $0 \leqslant i \leqslant \delta$, we denote the number of vertices of $T_1(x)$ at i-th level having one, two, and three children by $n_{i,1}, n_{i,2}$, and $n_{i,3}$. We denote the total number of vertices at the i-th level by n_i. Then for $0 \leqslant i < \delta$ it holds that

$$n_{i+1} = n_{i,1} + 2n_{i,2} + 3n_{i,3} = n_i + n_{i,2} + 2n_{i,3}. \tag{2}$$

Let $q = \max\limits_i n_{i,3}/n_i$. Then $n_{i,3} \leqslant qn_i$. From (2) it holds that

$$n_{i+1} \leqslant n_i + n_{i,2} + 2qn_i \leqslant 2n_i + 2qn_i = 2n_i(1+q).$$

From the definition of uniformly θ-dense tree for each i, $0 \leqslant i \leqslant \delta$, holds the inequality

$$n_i \geqslant 3^i\theta.$$

Then

$$3^{\delta}\theta \leqslant n_{\delta} \leqslant 2^{\delta} n_0 (1+q)^{\delta}.$$

Since $n_0 = 1$ ($i = 0$ corresponds to the root of the tree), we have

$$\delta \ln(1+q) + \delta \ln 2 \geqslant \ln\theta + \delta \ln 3.$$

Dividing by δ we finally obtain

$$q \geqslant \frac{3}{2} \cdot \theta^{1/\delta} - 1.$$

Now let us fix $\alpha = \log_{3/2}\left(1.1\,(2\delta)^{1/\delta}\right)$ and consider separately two cases.

If $\varepsilon \leqslant (2/3)^{\delta(1-\alpha)}$, then

$$\mathsf{Adv}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{A}) \leqslant \left(\frac{2}{3}\right)^{\delta}\left(1 + \left(\frac{2}{3}\right)^{-\alpha\delta}\right) = \left(\frac{2}{3}\right)^{\delta}\left(1 + 2\delta \cdot 1.1^{\delta}\right).$$

Otherwise, if $\varepsilon > (2/3)^{\delta(1-\alpha)}$ it holds that

$$\theta^{1/\delta} = \left(\frac{\varepsilon}{2\delta}\right)^{1/\delta} = \left(\frac{(2/3)^{\delta(1-\alpha)}}{2\delta}\right)^{1/\delta} = \frac{(2/3)^{(1-\alpha)}}{\sqrt[\delta]{2\delta}}.$$

It can be checked that for α defined as above it holds that $q \geqslant 0.1$. Note that q does not actually depend on θ.

Let j be the number of the level at which the maximum value of q is achieved. Then $n_{j,3} = qn_j$. We denote by $L_j(\pi)$ the predicate that the path π lies in $T_1(x)$ and goes through the left child of some vertex of level j. Similarly, we define predicates $C_j(\pi)$ and $R_j(\pi)$. By L_j we denote the predicate $\exists\pi\,(L_j(\pi))$.

We can write for the probability of the event L_j:

$$\mathsf{P}[L_j] \geqslant \mathsf{P}[\exists\pi((v_1 \in \pi \vee v_2 \in \pi \vee \ldots \vee v_{n_{j,3}} \in \pi) \wedge L_j(\pi))] = \sum_{i=1}^{n_{j,3}} \mathsf{P}[\exists\pi(v_i \in \pi \wedge L_j(\pi))].$$

Here v_i is a vertex from j-th level which has three children. For such a vertex the existence of the left child is guaranteed.

The probability $\mathsf{P}[\exists\pi(v_i \in \pi \wedge L_j(\pi)]$ for $1 \leqslant i \leqslant n_{j,3}$ is equal to the number S of paths in $T_1(x)$ passing through the left child of v_i divided by 3^{δ}. There are at most $3^{\delta-j-1}$ leaves in the subtree with root v_i. But since $T_1(x)$ is a uniformly θ-dense tree, it holds that

$$S \geqslant 3^{\delta-j-1}\theta \Rightarrow \mathsf{P}[\exists\pi(v_i \in \pi \wedge L_j(\pi))] \geqslant \frac{3^{\delta-j-1}\theta}{3^{\delta}}.$$

From this we can conclude that

$$\mathsf{P}[L_j] \geqslant n_{j,3} \cdot \frac{3^{\delta-j-1}\theta}{3^{\delta}} = \frac{n_{j,3}}{3^j} \cdot \frac{\theta}{3} = \frac{n_{j,3}}{n_j} \cdot \frac{n_j}{3^j} \cdot \frac{\theta}{3} \geqslant q \cdot \theta \cdot \frac{\theta}{3} \geqslant \frac{\theta^2}{30}.$$

Now let us find the probability P that by choosing $60/\theta^2$ branches π_j we find vertices $v_{j,0}, v_{j,1}$, and $v_{j,2}$ at the j-th level of $T_1(x)$.

$$\begin{gathered}
P = \mathsf{P}[\exists j_0, j_1, j_2\,(L_j(\pi_{j_0}) \wedge C_j(\pi_{j_1}) \wedge R_j(\pi_{j_2}))] = \\
= 1 - \mathsf{P}[\nexists j_0\, L_j(\pi_{j_0}) \vee \nexists j_1\, C_j(\pi_{j_1}) \vee \nexists j_2\, R_i(\pi_{j_2})] \geqslant \\
\geqslant 1 - \mathsf{P}[\nexists j_0\, L_j(\pi_{j_0})] - \mathsf{P}[\nexists j_1\, C_j(\pi_{j_1})] - \mathsf{P}[\nexists j_2\, R_j(\pi_{j_2})] = \\
= 1 - 3\,\mathsf{P}[\nexists j_0\, L_j(\pi_{j_0})] = 1 - 3\,\mathsf{P}[\overline{L}_j]^{60/\theta^2} = 1 - 3(1 - \mathsf{P}[L_j])^{60/\theta^2} \geqslant \\
\geqslant 1 - 3\left(1 - \frac{\theta^2}{30}\right)^{60/\theta^2} \geqslant 1 - \frac{3}{e^2}.
\end{gathered}$$

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 $p = \varepsilon/2\,(1 - 3/e^2) > \varepsilon/4$. ■

$\mathcal{B}$ runs the algorithm $1/p$ times. The complexity of one run is $T' = 60\,T/\theta^2$. The probability of failure is $(1-p)^{1/p}$ and, accordingly, the probability of success is $1-(1-p)^{1/p}$. Let us show that

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

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

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

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

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

The resulting complexity of the adversary $\mathcal{B}$ is $T'' = T'/p < 240\,T/(\theta^2\varepsilon)$. Let $\mathcal{B}$ solve $\mathrm{SD}(H, y, \omega)$ and $\mathrm{Coll}(h)$ with probabilities p_1 and p_2 respectively. Then

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

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

$$T_{\mathrm{SD},(1-1/e)} \leqslant \frac{1}{p_1} T_{\mathrm{SD},p_1} \leqslant \frac{1}{p_1} T'',$$
$$T_{\mathrm{Coll},(1-1/e)} \leqslant \frac{1}{p_2} T_{\mathrm{Coll},p_2} \leqslant \frac{1}{p_2} T''.$$

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

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

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

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

Equivalently,

$$\frac{960\delta^2 T}{\varepsilon^3} \geqslant \frac{1 - 1/e}{2}\,\widetilde{T}.$$

Finding ε from the last inequality and noting that $\sqrt[3]{\dfrac{1920}{1-1/e}} \leqslant 15$, we obtain

$$\varepsilon \leqslant 15\,\sqrt[3]{\delta^2 T/\widetilde{T}}.$$

Finally, for $\varepsilon > (2/3)^{\delta(1-\alpha)}$ we obtain

$$\mathsf{Adv}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{A}) \leqslant 15 \sqrt[3]{\frac{\delta^2 T}{\min\{T_{SD}, T_{Coll}\}}} + \left(\frac{2}{3}\right)^{\delta}.$$

For an arbitrary ε it holds that

$$\mathsf{Adv}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{A}) \leqslant \max\left\{15 \sqrt[3]{\frac{\delta^2 T}{\min\{T_{\text{SD}}, T_{\text{Coll}}\}}} + \left(\frac{2}{3}\right)^{\delta}, \left(\frac{2}{3}\right)^{\delta}\left(1 + 2\delta \cdot 1.1^{\delta}\right)\right\}.$$

Theorem 1 is proven. ■

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

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

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,

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

$\mathbf{Exp}^0(\mathcal{A})$

1 : $s \xleftarrow{\mathcal{U}} \{x \in \{0,1\}^n : \text{wt}(x) = \omega\}$
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 : $\quad\beta \xleftarrow{\mathcal{U}} \{0,1,2\}^{\delta}$
4 : $\quad\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
5 : **return** β

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 $\mathcal{A}^{\text{Sim}F_t}$ 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 β 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 : $t \xleftarrow{\mathcal{U}} \{1, \ldots, q_f\}$
4 : $(m, c\|r) \leftarrow\$\ \mathcal{A}^{SimF_t}(y)$
5 : **return** $(m, c\|r)$

Sim$F_t(\alpha)$

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** $: \beta \xleftarrow{\mathcal{U}} \{0,1,2\}^{\delta}$
5 : $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
6 : **return** β

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,

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

The equality here follows from the independent random choice of t.

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

$$\mathsf{P}[\mathbf{Exp}^{0}(\mathcal{A}) \Rightarrow 1] \leqslant \mathsf{P}[\mathbf{Exp}^{0}(\mathcal{A}) \Rightarrow 1 \wedge I \geqslant 1] + \mathsf{P}[\mathbf{Exp}^{0}(\mathcal{A}) \Rightarrow 1 \wedge I = 0] \leqslant$$
$$\leqslant \mathsf{P}[\mathbf{Exp}^{0}(\mathcal{A}) \Rightarrow 1 \wedge I \geqslant 1] + 3^{-\delta}.$$

Consequently,

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

From the above holds

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

$\mathcal{B}$ 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 the 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 the Stern signature scheme making at most q_f queries to the hashing oracle and

$$\mathsf{Adv}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) \geqslant \mathsf{Adv}^{\text{EUF-CMA}}_{\text{Stern}}(\mathcal{A}) - q_s \left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^{\delta},$$

where T_{Coll} is the complexity of optimal algorithm solving $\text{Coll}(h)$ problem with probability of success at least $1 - 1/e$ and $\tilde{c}$ 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 upper bounded by $T + c''(q_f + q_s T^{\text{Sig}}_{\text{Stern}})$, where $T^{\text{Sig}}_{\text{Stern}}$ is the complexity of the signature generation algorithm and c'' is a 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.

$\mathbf{Exp}^0(\mathcal{A}) = \mathbf{Exp}^{\text{EUF-CMA}}_{\text{Stern}}(\mathcal{A})$

1 : $s \xleftarrow{\mathcal{U}} \{x \in \{0,1\}^n : \text{wt}(x) = \omega\}$
2 : $y \leftarrow Hs^T$
3 : $\mathcal{L} \leftarrow \varnothing$
4 : $\Pi^F \leftarrow \varnothing$
5 : $(m, c\|r) \leftarrow\$\ \mathcal{A}^{\text{Sign},F}(y)$
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 : $\quad\beta \xleftarrow{\mathcal{U}} \{0,1,2\}^\delta$
4 : $\quad\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
5 : **return** β

Oracle Sign(s, m)

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

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

The experiment $\mathbf{Exp}^1$ is a modification of $\mathbf{Exp}^0$ obtained by introducing sets $\Pi^S, \Pi \subset \{0,1\}^* \times \{0,1,2\}^\delta$, Π^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,

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

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 $0 \leqslant i \leqslant \delta - 1$ 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 ξ and η coincide, then distributions of variables $h(\xi)$ and $h(\eta)$ also coincide. Indeed,

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

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

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

Oracle $F(\alpha)$

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

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

foreach $0 \leqslant i < \delta$:
  $u_i \xleftarrow{\mathcal{U}} \{0,1\}^n, \sigma_i \xleftarrow{\mathcal{U}} S_n$
  $c_{i,0} \leftarrow h(\sigma_i\|Hu_i^T)$
  $c_{i,1} \leftarrow h(\sigma_i(u_i))$
  $c_{i,2} \leftarrow h(\sigma_i(u_i \oplus s))$
  $c_i \leftarrow c_{i,0}\|c_{i,1}\|c_{i,2}$
$c \leftarrow c_0\| \ldots \|c_{\delta-1}$
if $(m\|c, \cdot) \in \Pi : b \leftarrow \Pi(m\|c)$
else
  $b \xleftarrow{\mathcal{U}} \{0,1,2\}^\delta$
  $\Pi^S \leftarrow \Pi^S \cup \{(m\|c, b)\}$
  $\Pi \leftarrow \Pi^F \cup \Pi^S$
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 $b_i = 2 : r_i \leftarrow \sigma_i(u_i)\|\sigma_i(s)$
$r \leftarrow r_0\| \ldots \|r_{\delta-1}$
$\mathcal{L} \leftarrow \mathcal{L} \cup \{m\}$
return $c\|r$

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

$s' \xleftarrow{\mathcal{U}} \{x \in \{0,1\}^n : \mathrm{wt}(x) = \omega\}$
foreach $0 \leqslant i < \delta$:
  $b_i \xleftarrow{\mathcal{U}} \{0,1,2\}$
  $u'_i \xleftarrow{\mathcal{U}} \{0,1\}^n, \sigma'_i \xleftarrow{\mathcal{U}} S_n,$
  if $b_i = 0$:
    $c_{i,0} \leftarrow h(\sigma'_i\|Hu_i'^T)$
    $c_{i,1} \leftarrow h(\sigma'_i(u'_i))$
    $c_{i,2} \leftarrow h(\sigma'_i(u'_i \oplus s'))$
    $r_i \leftarrow \sigma'_i\|u'_i$
  if $b_i = 1$:
    $c_{i,0} \leftarrow h(\sigma'_i\|(Hu_i'^T \oplus y))$
    $c_{i,1} \leftarrow h(\sigma'_i(s'))$
    $c_{i,2} \leftarrow h(\sigma'_i(u'_i))$
    $r_i \leftarrow \sigma'_i\|u'_i$
  if $b_i = 2$:
    $c_{i,0} \leftarrow h(\sigma'_i\|H(u'_i \oplus s')^T)$
    $c_{i,1} \leftarrow h(\sigma'_i(u'_i \oplus s'))$
    $c_{i,2} \leftarrow h(\sigma'_i(u'_i))$
    $r_i \leftarrow \sigma'_i(u'_i \oplus s')\|\sigma'_i(s')$
  $c_i \leftarrow c_{i,0}\|c_{i,1}\|c_{i,2}$
$c \leftarrow c_0\| \ldots \|c_{\delta-1}$
$r \leftarrow r_0\| \ldots \|r_{\delta-1}$
if $(m\|c, \cdot) \in \Pi^F$: **abort**
$\Pi^S \leftarrow \Pi^S \cup \{(m\|c, b)\}$
$\Pi \leftarrow \Pi^F \cup \Pi^S$
$\mathcal{L} \leftarrow \mathcal{L} \cup \{m\}$
return $c\|r$

In the 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

$$P_{a_1,a_2,a_3,a_4,a_5,a_6} = \mathsf{P}\left[\sigma_i = a_1, Hu_i^{\mathrm{T}} = a_2, \sigma_i(u_i) = a_3, \sigma_i(u_i \oplus s) = a_4, \sigma_i = a_5, u_i \oplus s = a_6\right] =$$
$$= \mathbb{I}\big[a_1 = a_5, H(a_6 \oplus s)^{\mathrm{T}} = a_2, a_1(a_6) = a_4\big]\ \mathsf{P}\left[\sigma_i = a_1, s = a_1^{-1}(a_3) \oplus a_6, u_i = a_1^{-1}(a_3)\right],$$

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

$$\widehat{P}_{a_1,a_2,a_3,a_4,a_5,a_6} = \mathsf{P}\left[\sigma'_i = a_1, Hu_i'^{\mathrm{T}} \oplus y = a_2, \sigma'_i(s') = a_3, \sigma'_i(u'_i) = a_4, \sigma'_i = a_5, u'_i = a_6\right] =$$
$$= \mathbb{I}\big[a_1 = a_5, Ha_6^{\mathrm{T}} \oplus y = a_2, a_1(a_6) = a_4\big]\ \mathsf{P}\left[\sigma'_i = a_1, s' = a_1^{-1}(a_3), u'_i = a_6\right].$$

As far as $H(a_6 \oplus s)^{\mathrm{T}} = Ha_6^{\mathrm{T}} \oplus y$, 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:

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

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

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

and distributions are indistinguishable.

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

$$P_{a_1,a_2,a_3,a_4,a_5,a_6} = \mathsf{P}\left[\sigma_i = a_1, Hu_i^{\mathrm{T}} = a_2, \sigma_i(u_i) = a_3, \sigma_i(u_i \oplus s) = a_4, \sigma_i(u_i) = a_5, \sigma_i(s) = a_6\right] =$$
$$= \mathbb{I}\big[a_3 = a_5, a_3 \oplus a_6 = a_4, H(a_1^{-1}(a_3))^{\mathrm{T}} = a_2\big]\ \mathsf{P}\left[\sigma_i = a_1, u_i = a_1^{-1}(a_3), s = a_1^{-1}(a_6)\right]$$

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

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

Reasoning similar to above, we get

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

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 Π^F. Let us consider the

worst case and suppose that $\mathcal{A}$ makes q_f queries to the hashing oracle F first. We denote by p' the probability that for some message at one of q_s queries the following condition was satisfied:

$$p' = \mathsf{P}\left[c \in \Pi_c^F\right],$$

where

$$\Pi_c^F = \{c \in \{0,1\}^{3\delta\ell} : \exists m \in \{0,1\}^* \, \exists \beta \in \{0,1,2\}^\delta \ \left((m\|c,\beta) \in \Pi^F\right)\}.$$

For a string c and a set $\Pi \subset \{0,1\}^* \times \{0,1\}^{3\delta\ell} \times \{0,1,2\}^\delta$, we can introduce the projection of this set to the part of the string:

$$\Pi_{c,(0,1)} = \{c_{0,1} : \exists m \in \{0,1\}^* \, \exists c \in \{0,1\}^{3\delta\ell} \, \exists \beta \in \{0,1,2\}^\delta \ ((m\|c,\beta) \in \Pi)\}.$$

Similarly, for a tuple $c_{*,1} = (c_{0,1}, \ldots, c_{\delta-1,1})$ we define

$$\Pi_{c,(*,1)} = \{c_{*,1} : \exists m \in \{0,1\}^* \, \exists c \in \{0,1\}^{3\delta\ell} \, \exists \beta \in \{0,1,2\}^\delta \ ((m\|c,\beta) \in \Pi)\}.$$

Then for $\Pi \subset \{0,1\}^* \times \{0,1\}^{3\delta\ell} \times \{0,1,2\}^\delta$ we can claim

$$p' \leqslant \sum_{\Pi} \mathsf{P}[\Pi^F = \Pi \wedge c_{*,1} \in \Pi_{c,(*,1)}].$$

Note that the first event is determined by $\mathcal{A}$'s random oracle whereas the second is determined by the signing one. That is, the events are independent and

$$p' \leqslant \sum_{\Pi} \mathsf{P}[\Pi^F = \Pi] \ \mathsf{P}[c_{*,1} \in \Pi_{c,(*,1)}].$$

Since each c_i is a function of independent random variables for $i \in \{0, \delta-1\}$, then events $c_{i,1} \in \Pi^F_{c,(i,1)}$ are also independent. Hence

$$p' \leqslant \sum_{\Pi} \mathsf{P}[\Pi^F = \Pi] \prod_{i=0}^{\delta-1} \mathsf{P}[c_{i,1} \in \Pi_{c,(i,1)}] \leqslant \sum_{\Pi} \mathsf{P}[\Pi^F = \Pi] \left(\max_{i\in\{0,\delta-1\}} \mathsf{P}[c_{i,1} \in \Pi_{c,(i,1)}]\right)^\delta \leqslant$$

$$\leqslant \left(\max_{\Pi} \max_{i\in\{0,\delta-1\}} \mathsf{P}[c_{i,1} \in \Pi_{c,(i,1)}]\right)^\delta \sum_{\Pi} \mathsf{P}[\Pi^F = \Pi] =$$

$$= \left(\max_{\Pi} \max_{i\in\{0,\delta-1\}} \mathsf{P}[c_{i,1} \in \Pi_{c,(i,1)}]\right)^\delta \leqslant \left(\max_{\Pi} \max_{i\in\{0,\delta-1\}} \sum_{y\in\Pi_{c,(i,1)}} \mathsf{P}[c_{i,1} = y]\right)^\delta \leqslant$$

$$\leqslant \left(\max_{\Pi} \max_{i\in\{0,\delta-1\}} \left(\left|\Pi_{c,(i,1)}\right| \max_{y\in\Pi_{c,(i,1)}} \mathsf{P}[c_{i,1} = y]\right)\right)^\delta \leqslant \left(q_f \max_{i\in\{0,\delta-1\}} \max_{y\in\{0,1\}^\ell} \mathsf{P}[h(x_i) = y]\right)^\delta \leqslant$$

$$\leqslant \left(q_f \sum_{i\in\{0,\delta-1\}} \max_{y\in\{0,1\}^\ell} \mathsf{P}[h(x_i) = y]\right)^\delta \leqslant \left(\delta q_f \max_{y\in\{0,1\}^\ell} \mathsf{P}[h(x) = y]\right)^\delta.$$

We can denote $p_h = \max\limits_{y\in\{0,1\}^\ell} \mathsf{P}[h(x) = y]$. Then there exists an algorithm $\mathcal{C}$ that evaluates hash function collision by the following steps: $\mathcal{C}$ chooses t random inputs x_i and evaluates their hash values. It stops after finding a collision. Then the probability that $\mathcal{C}$ fails decomposes into two incompatible events: there is no such $x \in \{x_i : i = 1, \ldots, t\}$ that $h(x) = y$ and there is only one such x. We claim the algorithm makes not more than $14/p_h$ steps and has the success probability at least $1 - 1/e$.

To prove this fact, we separate to cases: $p_h \leqslant 0.25$ and $p_h > 0.25$. In the first one, the number of steps can be taken equal to $t_1 = 3/p_h$. Then

$$\mathsf{P}[\mathcal{C} \text{ failes}] = (1-p_h)^{t_1} + \sum_{i=1}^{t_1} p_h(1-p_h)^{t_1-1} = (1-p_h)^{t_1} + t_1 p_h (1-p_h)^{t_1-1} \leqslant$$
$$\leqslant e^{-3} + \frac{(3/p_h)\, p_h e^{-3}}{1-p_h} = e^{-3}\left(1 + \frac{3}{1-p_h}\right) \leqslant 5e^{-3} < e^{-1}.$$

Here we used the fact that for $z > 0$ the inequality $(1+w)^z \leqslant e^{wz}$ holds.

If $p_h > 0.25$, we can fix $t_2 = 14$ and

$$\mathsf{P}[\mathcal{C} \text{ failes}] = (1-p_h)^{t_2} + \sum_{i=1}^{t_2} p_h(1-p_h)^{t_2-1} = (1-p_h)^{t_2} + t_2 p_h (1-p_h)^{t_2-1} \leqslant$$
$$\leqslant 0.75^{t_2} + t_2 p_h \cdot 0.75^{t_2-1} = 0.75^{t_2-1}(0.75 + t_2 p_h) < 0.75^{t_2-1}(1+t_2) < e^{-1}.$$

The complexity in the first case is $3\tilde{c}/p_h$ and in the second one it is $14\,\tilde{c}$, where $\tilde{c}$ is a constant depending on the model of computation and corresponding to the complexity of one hash function evaluation. So we can estimate the resulting complexity of the whole algorithm as $\check{T} = 14\tilde{c}/p_h$. Note that if $p_h > 0$, then the algorithm stops with probability equal to 1. Indeed, $\mathsf{P}[\mathcal{C} \text{ failes}]$ tends to zero as t tends to infinity.

Then for the complexity of the optimal algorithm T_{Coll} solving the problem $\text{Coll}(h)$ with probability at least $1 - 1/e$ it holds that

$$T_{\text{Coll}} \leqslant \check{T} \quad \text{and} \quad p_h \leqslant \frac{14\,\tilde{c}}{T_{\text{Coll}}}.$$

Finally, we obtain

$$p' \leqslant \left(\frac{14\,\tilde{c}\,\delta\, q_f}{T_{\text{Coll}}}\right)^{\delta}.$$

Let us denote by G the event that the condition on the line 23 was never satisfied. On the other hand, if the event $\overline{G}$ happened, then the condition was satisfied at least once during q_f $\mathcal{A}$'s queries. If $m^i \| c^i$ are strings formed in signature generating algorithm, then

$$\mathsf{P}[\overline{G}] = \mathsf{P}[m^1\|c^1 \in \Pi^F \vee \cdots \vee m^{q_s}\|c^{q_s} \in \Pi^F] \leqslant \mathsf{P}[c^1 \in \Pi_c^F \vee \cdots \vee c^{q_s} \in \Pi_c^F] =$$
$$= 1 - \mathsf{P}[c^1 \notin \Pi_c^F \wedge \cdots \wedge c^{q_s} \notin \Pi_c^F] = 1 - \prod_{i=1}^{q_s} \mathsf{P}[c^i \notin \Pi_c^F] =$$
$$= 1 - \prod_{i=1}^{q_s} (1 - \mathsf{P}[c^i \in \Pi_c^F]) = 1 - (1-p')^{q_s}.$$

Then, using the Bernoulli's inequality together with the fact $p' \leqslant 1$ and $q_s > 0$, we obtain

$$\mathsf{P}[\overline{G}] \leqslant 1 - 1 + q_s p' \leqslant q_s \left(\frac{14\,\tilde{c}\,\delta\, q_f}{T_{\text{Coll}}}\right)^{\delta}.$$

Since

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

then

$$\mathsf{P}[\mathbf{Exp}^1(\mathcal{A}) \Rightarrow 1] - \mathsf{P}[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1] \leqslant \mathsf{P}[\overline{G}] = q_s \left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^{\delta}.$$

Now, based on the adversary $\mathcal{A}$, we construct an adversary $\mathcal{B}$ that makes an existential forgery in the EUF-NMA model. It simulates oracles F and Sign using algorithms Sim$F$ 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 : $\Pi^F \leftarrow \varnothing$
3 : $(m, c\|r) \leftarrow\$\ \mathcal{A}^{\text{SimSign},\text{Sim}F}(y)$
4 : **if** $m \in \mathcal{L}$: **return** 0
5 : **return** $(m, c\|r)$

Oracle Sim$F(\alpha)$

1 : $\beta \leftarrow F^*(\alpha)$
2 : $\Pi^F \leftarrow \Pi^F \cup \{(\alpha, \beta)\}$
3 : **return** β

Now let us denote by $\text{Out}(\mathcal{A})$ and $\text{Out}(\mathcal{B})$ pairs $(m, c\|r)$ that are outputs of the adversary $\mathcal{A}$ in the experiment $\mathbf{Exp}^2$ and the adversary $\mathcal{B}$ in the experiment EUF-NMA, respectively. $\text{Out}(\mathcal{A})_{m,c}$ and $\text{Out}(\mathcal{B})_{m,c}$ are projections of the adversaries outputs to $m\|c$. Note that the projection is not defined if an adversary returns zero (in case $m \in \mathcal{L}$). But further we consider only $\mathbf{Exp}^2$ and EUF-NMA experiments with output equal to one, which excludes this case. We also define

$$\text{V}(G, m, c, r) = \text{Ver}(y, m, c\|r),$$

where Ver is the signature verification algorithm that uses function G. Then for $m \in \{0,1\}^*$, $c \in \{0,1\}^{3\delta\ell}$ we define

$$\begin{aligned} p_{m,c} &= \mathsf{P}[\mathbf{Exp}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) \Rightarrow 1 \wedge \text{Out}(\mathcal{B})_{m,c} = m\|c] = \\ &= \mathsf{P}[\mathbf{Exp}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) \Rightarrow 1 \wedge \text{Out}(\mathcal{B})_{m,c} = m\|c \wedge m\|c \in \Pi^F] + \\ &+ \mathsf{P}[\mathbf{Exp}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) \Rightarrow 1 \wedge \text{Out}(\mathcal{B})_{m,c} = m\|c \wedge m\|c \notin \Pi^F]. \end{aligned}$$

If $m\|c \notin \Pi^F$, then the adversary does not know the correct hash-value for this string and has to guess it. This is possible with the probability equal to $3^{-\delta}$. Also, note that the result of the EUF-NMA experiment equals the result of the signature verification algorithm, so it holds

$$\mathbf{Exp}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) = \text{V}(F, m, c, r). \tag{3}$$

Hence,

$$p_{m,c} = \mathsf{P}[\text{V}(F, m, c, r) \wedge \text{Out}(\mathcal{B})_{m,c} = m\|c \wedge m\|c \in \Pi^F] + 3^{-\delta}.$$

For $m\|c \in \Pi^F$ it holds that

$$\text{V}(F^*, m, c, r) = \text{V}(F, m, c, r)$$

as $\mathcal{A}$'s hashing oracle F is strictly determined by $\mathcal{B}$'s oracle F^*. Then

$$p_{m,c} = \mathsf{P}[\text{V}(F^*, m, c, r) \wedge \text{Out}(\mathcal{B})_{m,c} = m\|c \wedge m\|c \in \Pi^F] + 3^{-\delta}.$$

Similar to (3) we claim

$$\mathbf{Exp}^2(\mathcal{A}) = \mathrm{V}(F^*, m, c, r).$$

The adversary $\mathcal{B}$ always returns $\mathcal{A}$'s output so

$$\mathrm{Out}(\mathcal{B})_{m,c} = \mathrm{Out}(\mathcal{A})_{m,c}.$$

Finally, the $\mathcal{A}$'s strategy in the case $m\|c \notin \Pi^F$ is the same as $\mathcal{B}$'s and therefore has the same success probability $(1/3)^\delta$.

From the proceeding argument it becomes clear that

$$\begin{gathered} p_{m,c} = \mathsf{P}[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1 \wedge \mathrm{Out}(\mathcal{A})_{m,c} = m\|c \wedge m\|c \in \Pi^F] + \\ + \mathsf{P}[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1 \wedge \mathrm{Out}(\mathcal{A})_{m,c} = m\|c \wedge m\|c \notin \Pi^F] = \\ = \mathsf{P}[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1 \wedge \mathrm{Out}(\mathcal{A})_{m,c} = m\|c]. \end{gathered}$$

Thus,

$$\mathsf{P}[\mathbf{Exp}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) \Rightarrow 1] = \mathsf{P}[\mathbf{Exp}(\mathcal{B}) \Rightarrow 1 \wedge \bigvee_{(m,c)} \mathrm{Out}(\mathcal{B})_{m,c} = m\|c] =$$

$$= \sum_{(m,c)} p_{m,c} = \sum_{(m,c)} \mathsf{P}[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1 \wedge \mathrm{Out}(\mathcal{A})_{m,c} = m\|c] = \mathsf{P}[\mathbf{Exp}^2(\mathcal{A}) \Rightarrow 1].$$

Consequently,

$$\mathsf{Adv}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) \geqslant \mathsf{Adv}^{\text{EUF-CMA}}_{\text{Stern}}(\mathcal{A}) - q_s \left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^\delta.$$

The adversary $\mathcal{B}$ runs $\mathcal{A}$ and simulates q_f queries to the oracle F and q_s queries to the oracle Sign. Note that the complexity of the oracle Sign does not exceed the complexity of the original signing algorithm. Hence, $\mathcal{B}$'s complexity is no more than $T + c''(q_f + q_s T^{\text{Sig}}_{\text{Stern}})$. ■

Corollary 1. Let $\mathcal{A}$ be an adversary in the EUF-CMA model for the 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 it holds that

$$\begin{gathered} \mathsf{Adv}^{\text{EUF-CMA}}_{\text{Stern}}(\mathcal{A}) \leqslant \\ \leqslant \max\left\{ 15 q_f \sqrt[3]{\frac{\delta^2 (T + \tilde{\tilde{c}}(2q_f + q_s T^{\text{Sig}}_{\text{Stern}}))}{\min\{T_{\text{SD}}, T_{\text{Coll}}\}}} + \left(\frac{2}{3}\right)^\delta (1 + q_f) + q_s \left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^\delta, \right. \\ \left. \left(\frac{2}{3}\right)^\delta \left(1 + q_f \left(1 + 2\delta \cdot 1.1^\delta\right)\right) + q_s \left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^\delta \right\}, \end{gathered}$$

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

Proof. The complexity $\hat{T}$ of an adversary in the EUF-NMA model with one query to the hashing oracle from Theorems 1–3 does not exceed $T + \tilde{\tilde{c}}(2q_f + q_s T^{\text{Sig}}_{\text{Stern}})$, where T is the complexity of an adversary in the EUF-CMA model and $\tilde{\tilde{c}} = \max\{c', c''\}$.

Also, for an adversary $\mathcal{B}$ in the EUF-NMA model making at most q_f queries to the hashing oracle follows that

$$\mathsf{Adv}_{\text{Stern}}^{\text{EUF-NMA}}(\mathcal{B}) \leqslant$$

$$\leqslant \max\left\{15q_f\sqrt[3]{\frac{\delta^2\hat{T}}{\min\{T_{\text{SD}},T_{\text{Coll}}\}}}+\left(\frac{2}{3}\right)^{\delta}(1+q_f),\left(\frac{2}{3}\right)^{\delta}\left(1+q_f\left(1+2\delta\cdot 1.1^{\delta}\right)\right)\right\},$$

$$\mathsf{Adv}_{\text{Stern}}^{\text{EUF-CMA}}(\mathcal{A}) \leqslant \mathsf{Adv}_{\text{Stern}}^{\text{EUF-NMA}}(\mathcal{B}) + q_s\left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^{\delta} \leqslant$$

$$\leqslant \max\left\{15q_f\sqrt[3]{\frac{\delta^2\hat{T}}{\min\{T_{\text{SD}},T_{\text{Coll}}\}}}+\left(\frac{2}{3}\right)^{\delta}(1+q_f)+q_s\left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^{\delta},\right.$$

$$\left.\left(\frac{2}{3}\right)^{\delta}\left(1+q_f\left(1+2\delta\cdot 1.1^{\delta}\right)\right)+q_s\left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^{\delta}\right\}.$$

Corollary 1 is proven. ■

5. Parameters

In this Section, we mention different constraints on the signature parameters and introduce a parameter set.

5.1. Choosing general parameters

J. 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 δ should be chosen to satisfy the condition

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

where λ is the security parameter.

As it was mentioned above, the complexity of the best collision-finding algorithm is $\mathcal{O}\left(2^{\ell/2}\right)$, where ℓ is the length of the hash value. Since we want to maximize this complexity, it is worth using hash functions with the maximal ℓ. 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 $T_{\text{Coll}} \approx 2^{256}$ bit operations. Further by default we suppose that hash function Streebog-512 is used.

In order to maximize the value of $\min\{T_{\text{SD}}, T_{\text{Coll}}\}$, it is necessary to choose the length of the code so that $T_{\text{SD}} \geqslant T_{\text{Coll}}$. From the fact that $T_{\text{SD}} \approx 2^{0.0885n}$ [29], a lower estimate for the length of the code n can be found. 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 $\omega \approx 0.11n$.

5.2. Public data and signature sizes

The public key is a vector y of $n-k$ 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 $\delta(3\ell + n + n\log_2 n)$ bits.

5.3. Example parameter set

Parameters q_f, q_s, T, and $T^{\text{Sig}}_{\text{Stern}}$ on practice depend on the desired level of security and the realization of hash functions. We assume that the maximal complexity T of the adversary does not exceed 2^{70} bit operations. The structure of function f we specify as follows:

$$f(x) = \left\lfloor \frac{h'(x)\, 3^{\delta}}{2^{256}} \right\rfloor,$$

where h' is Streebog-256 hash function and the output is considered as a ternary vector. So the complexity of h' dominates in the complexity of f. 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. In this case, according to [30], the complexity of Streebog is about 2^{25} CPU cycles or more than 2^{30} bit operations. Signing oracle has to evaluate at least hash function f, thus $q_s \leqslant q_f$. As a result, an adversary with complexity T is able to do no more that 2^{40} queries to each oracle.

The value $T^{\text{Sig}}_{\text{Stern}}$ consists of the complexity of function f and the complexity of δ computations of the signature components. The complexity of one component (c_i, r_i) computation includes the triple calculation of the hash function h, which can be estimated as $2^{23.5}$ bit operations using Streebog realization from [30], and other calculations, the complexity of which in total equals 2^{22} bit operations. Finally, the computation of δ components requires 2^{31} bit operations, and $T^{\text{Sig}}_{\text{Stern}}$ is about $2^{31.5}$ bit operations.

The table presents an example parameter set for signature with 70-bit security.

λ	n	k	ω	δ	ℓ	H, MB	y, KB	ζ, MB
70	2896	1448	318	137	512	0.25	0.18	0.62

The adversary's advantage in this case equals approximately $1.5 \cdot 10^{-4}$.

Although the introduced parameter set guarantees 70 bits of security, we presume that our estimate is rather rough and in fact one can count on larger security level.

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 70-bit security of the signature. As a direction for further research, we consider the refinement of the obtained estimate and the extension of the security proof to a model with quantum access to the random oracle.

7. Acknowledgments

The authors thank Lev Vysotsky, Liliya Akhmetzyanova, Alexandra Babueva and Kirill Tsaregorodtsev for helpful discussions.

REFERENCES

1. *Shor P. V.* Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Computing, 1997, vol. 26, no. 5, pp. 1484–1509.

2. https://csrc.nist.gov/Projects/post-quantum-cryptography/Post-Quantum-Cryptography-Standardization/Call-for-Proposals — NIST PQC Call for Proposals, 2016.
3. *Lee W., Kim Y.-S., Lee Y.-W., and 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, https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/pqsigRM.zip.
4. *Fukushima K., Roy P. S., Xu R., et al.* Supporting documentation of RaCoSS (Random Code-based Signature Scheme). First round submission to the NIST post-quantum cryptography call, 2017, https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/RaCoSS.zip.
5. *Aragon N., Gaborit P., Hauteville A., et al.* RankSign — a signature proposal for the NIST's call. First round submission to the NIST post-quantum cryptography call, 2017, https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/RankSign.zip.
6. *Debris-Alazard T. and Tillich J.-P.* Two attacks on rank metric code-based schemes: RankSign and an IBE scheme. LNCS, 2018, vol. 11272, pp. 62–92.
7. *Lee Y., Lee W., Kim Y. S., and No J.-S.* Modified pqsigRM: RM code-based signature scheme. IEEE Access, 2020, vol. 8, pp. 177506–177518.
8. *Roy P. S., Morozov K., Fukushima K., et al.* Code-based signature scheme without trapdoors. IEICE Tech. Rep., 2018, vol. 118, no. 151, pp. 17–22.
9. *Xagawa K.* Practical Attack on RaCoSS-R. Cryptology ePrint Archive, 2018, Report 2018/831, http://eprint.iacr.org/
10. *Kabatianskii G., Krouk E., and Smeets B.* A digital signature scheme based on random error-correcting codes. LNCS, 1997, vol. 1355, pp. 161–167.
11. *Cayrel P.-L., Otmani A., and Vergnaud D.* On Kabatianskii — Krouk — Smeets signatures. LNCS, 2007, vol. 4547, pp. 237–252.
12. *Stern J.* Can one design a signature scheme based on error-correcting codes? LNCS, 1995, vol. 917, pp. 424–426.
13. *Courtois N., Finiasz M., and Sendrier N.* How to achieve a McEliece-based digital signature scheme. LNCS, 2001, vol. 2248, pp. 157–174.
14. *McEliece R. J.* A public-key cryptosystem based on algebraic coding theory. DSN Progress Report, 1978, vol. 42–44, pp. 114–116.
15. *Niederreiter H.* Knapsack-type cryptosystems and algebraic coding theory. Problems Control Inform. Theory, 1986, vol. 15, no. 2, pp. 159–166.
16. *Dallot L.* Towards a concrete security proof of Courtois, Finiasz and Sendrier signature scheme. LNCS, 2008, vol. 4945, pp. 65–77.
17. *Debris-Alazard T., Sendrier N., and Tillich J.-P.* Wave: a new family of trapdoor one-way preimage sampleable functions based on codes. LNCS, 2019, vol. 11921, pp. 21–51.
18. *Fiat A. and Shamir A.* How to prove yourself: practical solutions to identification and signature problems. LNCS, 1987, vol. 263, pp. 186–194.
19. *Stern J.* A new identification scheme based on syndrome decoding. LNCS, 1994, vol. 773, pp. 13–21.
20. *Jain A., Krenn S., Pietrzak K., and Tentes A.* Commitments and efficient zero-knowledge proofs from learning parity with noise. LNCS, 2012, vol. 7658, pp. 663–680.
21. *Cayrel P.-L., Véron P., and El Y. A. S. M.* A zero-knowledge identification scheme based on the q-ary SD problem. LNCS, 2010, vol. 6544, pp. 171–186.
22. *Lyubashevsky V.* Lattice signatures without trapdoors. LNCS, 2012, vol. 7237, pp. 738–755.

23. *Aragon N., Blazy O., Gaborit P., et al.* Durandal: a rank metric based signature scheme. LNCS, 2019, vol. 11478, pp. 728–758.
24. *Overbeck R. and Sendrier N.* Code-based cryptography. Post-Quantum Cryptography, 2009, pp. 95–145.
25. *Roy P. S., Morozov K., Fukushima K., and Kiyomoto S.* Evaluation of Code-Based Signature Schemes. Cryptology ePrint Archive, 2019, Report 2019/544, `https://eprint.iacr.org/`
26. *El Y. A. S. M., Cayrel P.-L., El B. R., and Hoffmann G.* Code-based identification and signature schemes in software. LNCS, 2013, vol. 8128, pp. 122–136.
27. *Pointcheval D. and Stern J.* Security proofs for signature schemes. LNCS, 1996, vol. 1070, pp. 387–398.
28. *Berlekamp E., McEliece R., and van Tilborg H.* On the inherent intractability of certain coding problems (Corresp.). IEEE Trans. Inform. Theory, 1978, vol. 24, no. 3, pp. 384–386.
29. *Both L. and May A.* Decoding linear codes with high error rate and its impact for LPN security. LNCS, 2018, vol. 10786, pp. 25–46.
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., 2013, vol. 4, no. 2, pp. 73–80.