# On the (In)security of optimized Stern-like signature schemes

André Chailloux

Inria de Paris, COSMIQ team, andre.chailloux@inria.fr

**Abstract.** Stern's signature is an important code-based signature scheme. An important optimization is to generate pseudo-random vectors and permutation instead of random ones, and most proposals that are based on Stern's signature use this optimization. However, its security hasn't been properly analyzed. In this article, we study the security of this optimization. We first show that for some parameters, there is an attack that exploits this optimization and breaks the scheme in time $O(2^{\frac{\lambda}{2}})$ while the claimed security is $\lambda$ bits. This impacts in particular the recent Quasy-cyclic Stern signature scheme [BGS21]. Our second result shows that there is an efficient fix to this attack. By adding a string $salt \in \{0,1\}^{2\lambda}$ to the scheme, and changing slightly how the pseudo-random strings are generated, we prove not only that our attack doesn't work but that for any attack, the scheme preserves $\lambda$ bits of security, and this fix increases the total signature size by only $2\lambda$ bits. Additionally, we perform this analysis against quantum adversaries. This result not only has an impact on the Quasy-cyclic Stern signature scheme but also on other Stern-based signature scheme. While those aren't impacted by our attack, they have parameters that are too conservative so our result shows how they can improve the efficiency of their scheme without losing on the security.

## 1 Introduction

Stern's signature scheme [Ste94] is one of the main code-based signature schemes. As the NIST standardization process [NIS] for post-quantum cryptography advances, the need for building efficient and secure post-quantum signature schemes becomes increasingly important. There are many variants of Stern's signature schemes which have been proposed. Their security are all based on the Fiat-Shamir transform, which is well understood even against quantum adversaries [Cha19,GHHM21]. The problem with Stern's signature scheme is that in its original form, the signature sizes are extremely large. There are several optimizations that are done in order to make the signature sizes smaller, by adding some rounds, using the cut and choose technique, or changing the metric ([Vér97],[AGS11],[CVE11],[GRSZ14], [Beu20],[BBC$^+$20],[BGS21]). However, once we add these optimizations, it's unclear whether the security reduction still holds.

In this article we study the security of one of the main optimizations in Stern's signature scheme. In the original scheme, the signer must generate many

random vectors $\mathbf{u}$ as well as a random permutations $\pi$. In the signature, some of these have to be revealed and for others, we only have the commitment to these values. Revealing all these values is very costly and makes the signature very long.

The idea to improve this is to consider vectors and permutations that arise from a pseudo-random generator. For example, for constructing a vector $\mathbf{u} \in \mathbb{F}_q^n$, we use a function $E : \{0,1\}^{S_{\text{seed}}} \to \mathbb{F}_q^n$ that should behave like as a random function with $S_{seed}$ significantly smaller $n \log_2(q)$. Instead of constructing $\mathbf{u}$, we generate a random seed $\in S_{\text{seed}}$ and construct $\mathbf{u} = E(\text{seed})$. This greatly reduces the signature size since instead of revealing $\mathbf{u}$, we can only reveal seed.

This idea has been used in several proposals but has never been properly proven and we don't even know to what extent this optimization remains secure. In order to maintain $\lambda$ bits of security[1] should we have $S_{\text{seed}} = \{0,1\}^{2\lambda}$ or can we take $S_{\text{seed}} = \{0,1\}^{\lambda}$? As we said, Stern's scheme also uses a commitment scheme so with this optimization, can we use a deterministic commitment of size $2\lambda$ or should we use a randomized commitment scheme?

Strikingly, each proposal that uses this optimization doesn't use the same answers to these questions, some are quite conservative while others take the maximum amount of risk. This shows the necessity to study this problem and to give an answer for what is and what isn't possible.

Our first contribution is to show that taking the maximum risk, *i.e.* taking $S_{\text{seed}} = \{0,1\}^{\lambda}$ and a deterministic commitment size of size $2\lambda$ *doesn't preserve* $\lambda$ bits of security. More precisely, we present an attack on these parameters that recovers the secret key in time $O(2^{\frac{\lambda}{2}})$.

**Theorem 1.** *Consider an optimized version of Stern's signature scheme with deterministic commitments of size* $2\lambda$ *and a* $S_{\text{seed}} = \lambda$*. Then there is an attack that recovers the secret key in time* $O(2^{\frac{\lambda}{2}})$*.*

This theorem implies in particular that the security claims of [BGS21] are incorrect, and that there is an attack on their scheme that runs in time $O(2^{\lambda/2})$ (their scheme is a little bit different than the one we analyze here but our attack works the same way). The attack uses a combination of facts: the commitment function is deterministic and the seed size is small. This prompts 2 possible fixes to this attack:

1. *Make the commitment probabilistic.* This means to commit to a string $z$, instead of computing $\mathscr{H}_{comm}(z)$ for a hash function $\mathscr{H}_{comm}$, you first pick a random string $r$ of size $2\lambda$ and compute $\mathscr{H}_{comm}(z||r)$. While the commitment size is still $2\lambda$ you additionally must reveal $r$ when you reveal $z$ which adds a cost of $2\lambda$ which is very costly since you reveal many strings. This method was used in MQDSS signature scheme but we expect that this is unnecessary, and that our solution can be used in their setting which would significantly decrease the signature size.

[1] When we write maintain $\lambda$ bits of security, we mean that if the original scheme has $\lambda$ bits of security then the optimized scheme should also have $\lambda$ bits of security.

2. *Increase the seed size.* If we use seeds of size $2\lambda$ instead of $\lambda$ then the attack doesn't work, and we can actually prove security. This is again however very costly, as the signature contains a large number of seeds.

These 2 fixes are very costly. Our second contribution is to present a fix that will solve this problem by increasing the total signature size by only $2\lambda$ bits. The main idea is to make to change the generation of $\mathbf{u}$. Before, the protocol must generate many vectors $\mathbf{u}^1, \ldots, \mathbf{u}^R$ and use $\mathbf{u}^i = E(\text{seed}^i)$. Here we generate an extra string $salt \in \{0,1\}^{2\lambda}$ and use $\mathbf{u} = E(\text{seed}^i || salt || i)$ (we increase the input space of $E$ accordingly) so we add string $salt \in \{0,1\}^{2\lambda}$ randomly chosen at the beginning of the signing algorithm. What is important is that while we take a different seed for each $i$, the salt can remain the same for each generation of $\mathbf{u}^i$ so we only need to add one value *salt*. Our main contribution here is to show that if $E$ behaves like a random function then the protocol preserves $\lambda$ bits of security. Adding the index $i$ of the vector also slightly increases the security. With this change, we obtain the following:

**Theorem 2.** *There is a small modification to all optimized Stern's signatures schemes that preserve $\lambda$ bits of security with commitments of size $2\lambda$ and seed size $S_{\text{seed}} = \lambda$, that increases the total signature size by $2\lambda$ bits only.*

Finally, we also prove this statement in the quantum setting, where sign queries remain classical but we otherwise have a fully quantum adversary.

**Theorem 3.** *There is a small modification to all optimized Stern's signatures schemes that preserve $\lambda$ bits of security with commitments of size $3\lambda$ and seed size $S_{\text{seed}} = 2\lambda$, that increases the total signature size by $2\lambda$ bits only.*

## 2 Preliminaries

### 2.1 Signature schemes

**Definition 1.** *A signature scheme $S$ consists of* 3 *efficient algorithms* S.KEYGEN, S.SIGN *and* S.VERIFY *where:*

- $\text{S.KEYGEN}(1^\lambda) \to (pk, sk)$ *is the generation of the public key $pk$ and the secret key $sk$ from the security parameter $\lambda$.*
- $\text{S.SIGN}(m, pk, sk) \to \sigma_m$ *: generates the signature $\sigma_m$ of a message $m$ from $m, pk, sk$.*
- $\text{S.VERIFY}(m, \sigma, pk) \to \{0,1\}$ *verifies that $\sigma$ is a valid signature of $m$ using $m, \sigma, pk$. The output* 1 *corresponds to a valid signature.*

*A signature scheme has perfect correctness iff. when we run $(pk, sk) \leftarrow \text{S.KEYGEN}(1^\lambda)$, we have for each $m$*

$$\text{S.VERIFY}(m, \text{S.SIGN}(m, pk, sk), pk) = 1.$$

### 2.2 EUF-CMA Security

We consider the standard EUF-CMA security for signature schemes. To define the advantage of an adversary $\mathcal{A}$, we consider the following interaction with a challenger:

**Modeling an adversary $\mathcal{A}$ in the EUF-CMA model**

- Initialize. The challenger generates $(pk, sk) \leftarrow \text{S.KEYGEN}(1^\lambda)$ and sends $pk$ to $\mathcal{A}$.
- Query phase. $\mathcal{A}$ can perform sign queries by sending each time a message $m$ to the challenger who generates $\sigma = \text{S.SIGN}(m, pk, sk)$ and sends $\sigma$ to $\mathcal{A}$. Let $m_1, \ldots, m_{q_S}$ the (not necessarily distinct) queries made by $\mathcal{A}$.
- Output. $\mathcal{A}$ outputs a pair $(m^*, \sigma^*)$.

**Definition 2.** *The EUF-CMA advantage for $S$ with an adversary $\mathcal{A}$ described above is the quantity*

$$\text{ADV}_{\text{S}}(\mathcal{A}) = \Pr\left[\text{S.VERIFY}(m^*, \sigma^*, pk) = 1 \wedge (\forall i \in [q_S],\ m^* \neq m_i) \middle| \begin{smallmatrix}(pk,sk)\leftarrow \text{S.KEYGEN}(1^\lambda)\\ (m^*,\sigma^*)\leftarrow A(pk)\end{smallmatrix}\right].$$

### 2.3 The syndrome decoding problem

The canonical hard problem used in code-based cryptography is the Syndrome Decoding problem. Given a parity matrix $\mathbf{H}$ of a code and a syndrome $\mathbf{s}$, the goal is to find an error vector $\mathbf{e}$ of fixed Hamming weight such that $\mathbf{He} = \mathbf{s}$.

*Problem 1.* Syndrome Decoding $\text{SD}(n, k, w)$
**Input.** A matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, a column vector (the syndrome) $\mathbf{s} \in \mathbb{F}_q^{n-k}$.
**Goal.** Find a column vector $\mathbf{e} \in \mathbb{F}_q^n$ s.t. $\mathbf{He} = \mathbf{s}$ and $|\mathbf{e}| = w$, where $|\mathbf{e}| = |\{i : e_i \neq 0\}|$.

The mostly studied problem is the Binary Syndrome Decoding problem which corresponds to the case $q = 2$. For any integer $q \geq 2$, the decision variant of this problem, where we ask whether a vector $\mathbf{e}$ of weight $w$ satisfying $\mathbf{He} = \mathbf{s}$ exists or not, is NP hard. Most variants of this problem are also believed to be hard on average even against quantum computers, when the input is taken from the following distribution, after removing the error $\mathbf{e}$.

$$\mathscr{D}_{n,k,w} : \mathbf{H} \leftarrow \mathbb{F}_q^{(n-k)\times n}, \mathbf{e} \leftarrow S_w,\ \text{return } (\mathbf{H}, \mathbf{e}, \mathbf{s} = \mathbf{He}). \tag{1}$$

The Syndrome Decoding problem is therefore a prime candidate for designing post-quantum cryptographic protocols.

## 3 Stern's signature scheme

We first present the ideal Stern's signature scheme without any optimization. We actually present the identification scheme and then briefly explain how to transform it into a signature scheme.

### 3.1 Ideal Stern's signature scheme

Stern's identification scheme for $\mathrm{SD}(n,k,w)$ $[S_{\mathrm{Ideal}}]$

**Key generation.** Sample $(\mathbf{H}, \mathbf{e}, \mathbf{s} = \mathbf{He}) \leftarrow D_{\mathrm{SD}(n,k,w)}$ where $D_{\mathrm{SD}(n,k,w)}$ is defined in Equation 1. Output $pk = (\mathbf{H}, \mathbf{s})$ and $sk = \mathbf{e}$.

**Protocol.**

**1.** For i from 1 to $R$: the prover picks a random permutation $\pi^i$ acting on $[n]$ and a random column vector $\mathbf{y}^i \in \{0,1\}^n$. Let $\mathbf{t}^i = \mathbf{H}\mathbf{y}^i$. Let also

$$z_1^i = (\pi^i || t^i),\ z_2^i = \pi^i(\mathbf{y}^i),\ z_3^i = \pi^i(\mathbf{y}^i + \mathbf{e}^i).$$

The prover then commits to each $z_1^i, z_2^i$ and $z_3^i$ separately by computing $comm_j^i = \mathscr{H}_{comm}(z_j^i)$ for $j \in \{1,2,3\}$ and sends all the $comm_j^i$ for $i \in [R]$ and $j \in \{1,2,3\}$.

**2.** The verifier sends a uniformly random challenge $c = (c^1, \ldots, c^R)$ where each $c^i \in \{1,2,3\}$.

**3.** For i from 1 to $R$: the prover reveals the following to the verifier:

- If $c^i = 1$, reveal $z_2^i = \pi^i(\mathbf{y}^i)$ and $z_3^i - z_2^i = \pi^i(\mathbf{e})$.
- If $c^i = 2$, reveal $\pi^i, \mathbf{y}^i + \mathbf{e}^i$.
- If $c^i = 3$, reveal $\pi^i, \mathbf{y}^i$.

**Verification.** For $i$ from 1 to $R$

- if $c^i = 1$, compute $z_3^i$, check that $|z_3^i - z_2^i| = w$ and that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 2$ and $j = 3$.
- if $c^i = 2$, compute $\mathbf{t}^i = \mathbf{H}(\mathbf{y}^i + \mathbf{e}^i) - \mathbf{s}$. Compute $z_1^i = (\pi^i || t^i),\ z_3^i = \pi^i(\mathbf{y}^i + \mathbf{e}^i)$ and check that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 1$ and $j = 3$.
- if $c^i = 3$, compute $\mathbf{t}^i = \mathbf{H}(\mathbf{y}^i)$, compute $z_1^i = (\pi^i || t^i),\ z_2^i = \pi^i(\mathbf{y}^i)$ and check that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 1$ and $j = 2$.

The ideal Stern's signature scheme, thereafter denoted $S_{\mathrm{Ideal}}$ is constructed taking the above identification scheme and applying the Fiat-Shamir transform. This means we replace the verifier's challenge by $\mathscr{H}(M_1, m)$ for a hash function $\mathscr{H}$ where $M_1$ is the first message sent by the prover in the identification scheme, and $m$ is the message to be signed. The signature $\sigma_m$ becomes then the whole transcript of the identification scheme and the verification remains unchanged, where we also verify that the challenge has been correctly constructed.

### 3.2 Optimized version of Stern's signature scheme

We present here an optimized version of Stern's signature. We consider a very simple optimized scheme, where only one vector $\mathbf{u}$ is replaced with a pseudo-random vector. The optimized identification scheme is presented below, differences with the ideal scheme are highlighted in blue.

Optimized Stern's identification scheme for $\mathrm{SD}(n,k,w)$ $[S_{\mathrm{OPT}}]$

**Key generation.** Sample $(\mathbf{H}, \mathbf{e}, \mathbf{s} = \mathbf{He}) \leftarrow D_{\mathrm{SD}(n,k,w)}$ where $D_{\mathrm{SD}(n,k,w)}$ is defined in Equation 1. Output $pk = (\mathbf{H}, \mathbf{s})$ and $sk = \mathbf{e}$.

**Protocol.**

**1.** For i from 1 to $R$: the prover picks a random permutation $\pi^i$ acting on $[n]$, **picks a random string** $\mathrm{seed}^i \in \{0,1\}^{S_{\mathrm{seed}}}$ **and computes the string** $\mathbf{u}^i = E(\mathrm{seed}^i)$ **and also defines** $\mathbf{y} = \pi^{-1}(\mathbf{u})$. Let also

$$z_1^i = (\pi^i || t^i),\ z_2^i = \mathbf{u^i} = \pi^i(\mathbf{y}^i),\ z_3^i = \pi^i(\mathbf{y}^i + \mathbf{e}^i).$$

The prover then commits to each $z_1^i, z_2^i$ and $z_3^i$ separately by computing $comm_j^i = \mathscr{H}_{comm}(z_j^i)$ for $j \in \{1,2,3\}$ and sends all the $comm_j^i$ for $i \in [R]$ and $j \in \{1,2,3\}$.

**2.** The verifier sends a uniformly random challenge $c = (c^1, \ldots, c^R)$ where each $c^i \in \{1,2,3\}$.

**3.** For i from 1 to $R$: the prover reveals the following to the verifier:

- If $c^i = 1$, reveal $\mathrm{seed}^i$ and $z_3^i - z_2^i = \pi^i(\mathbf{e})$.
- If $c^i = 2$, reveal $\pi^i$,$\mathbf{y}^i + \mathbf{e}^i$.
- If $c^i = 3$, reveal $\pi^i$, $\mathrm{seed}^i$.

**Verification.** For $i$ from 1 to $R$

- if $c^i = 1$, **compute** $z_2^i = E(\mathrm{seed}^i)$, compute $z_3^i$, check that $|z_3^i - z_2^i| = w$ and that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 2$ and $j = 3$.
- if $c^i = 2$, compute $\mathbf{t}^i = \mathbf{H}(\mathbf{y}^i + \mathbf{e}^i) - \mathbf{s}$. Compute $z_1^i = (\pi^i || t^i)$, $z_3^i = \pi^i(\mathbf{y}^i + \mathbf{e}^i)$ and check that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 1$ and $j = 3$.
- if $c^i = 3$, **compute** $z_2^i = E(\mathrm{seed}^i)$**, compute** $\mathbf{y}^i = \pi^{-1}(\mathbf{u}^i)$, compute $\mathbf{t}^i = \mathbf{H}(\mathbf{y}^i)$, compute $z_1^i = (\pi^i || t^i)$, and check that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 1$ and $j = 2$.

Again, we construct the signature scheme $S_{\mathrm{OPT}}$ by applying the Fiat-Shamir transform to the above identification scheme. We assume the pseudo-random generator $E$ behaves as a random function. In the next section, we prove that if $S_{\mathrm{seed}} = \lambda$, then the above scheme only has $\lambda/2$ bits of security. In actual optimized schemes, there are even more optimization but we show that this optimization already compromises the security of the scheme.

## 4 Attack on the $S_{\mathrm{OPT}}$ signature scheme in time $O(2^{\frac{\lambda}{2}})$

**Theorem 1.** *Consider the signature scheme $S_{\mathrm{OPT}}$ that uses any function $E : \{0,1\}^{\lambda} \rightarrow \{0,1\}^{\lceil n \log_2(q) \rceil}$ as the expansion function and a random function $\mathscr{H}_{comm} : \{0,1\}^* \rightarrow \{0,1\}^{2\lambda}$ as the commitment function. There exists an attack on $S_{\mathrm{OPT}}$ recovers the secret key in time $O\left(2^{\frac{\lambda}{2}}\right)$, doing $O\left(\frac{1}{R}2^{\frac{\lambda}{2}}\right)$ sign queries.*

*Proof.* We first present the attack and then prove its running time.

Attack $\mathcal{A}$ on the $S_{\mathrm{OPT}}$ signature scheme

**while** $\mathcal{A}$ doesn't recover the secret key $\mathbf{e}$:

1. Perform $q_S = \lceil \frac{1}{R}2^{\frac{\text{seed}}{2}} \rceil$ signature queries (with any message) and gather all the signatures $\sigma[1], \ldots, \sigma[q_S]$. From each $\sigma[k]$ gather the corresponding $comm_2^i[k] = (\mathscr{H}_{comm} \circ E)(\text{seed}^i[k])$.
2. Find all pairs of couples $((i,k),(i',k'))$ with $(i',k') \neq (i,k)$ st. $comm_2^i[k] = comm_2^{i'}[k']$.
3. For each such pair of couples $((i,k),(i',k'))$, if

$$\text{chall}^i[k] = 2 \ \wedge \ \text{chall}^{i'}[k'] \neq 2 \ \wedge \text{seed}^i[k] = \text{seed}^{i'}[k'] \tag{2}$$

then the following procedure recovers $\mathbf{e}$:

- From $\sigma[k]$, since $\text{chall}^i[k] = 2$, you can recover $z_1^i[k] = (\pi^i[k], t^i[k])$ and $z_3^i[k] = \pi^i[k](y^i[k] + \mathbf{e})$.
- From $\sigma[k']$, since $\text{chall}^{i'}[k'] \neq 2$, you can recover $\text{seed}_{k'}^{i'}$. But since $\text{seed}^i[k] = \text{seed}^{i'}[k']$, you can also recover $z_2^i[k] = E^{seed^i[k]} = E^{seed^{i'}[k']}$.
- From $z_1^i[k] = (\pi^i[k], t^i[k]), z_2^i[k] = \pi^i[k](y^i[k]), z_3^i[k] = \pi^i[k](y^i[k] + \mathbf{e})$, recover the secret key by computing:

$$\mathbf{e} = (\pi^i[k])^{-1}\left(z_3^i[k] - z_2^i[k]\right).$$

where $(\pi^i[k])^{-1}$ is the inverse of $(\pi^i[k])$.

We now prove that each iteration of the while loop will find the secret key with a constant probability. First notice that with $q_S = \lceil \frac{1}{R}2^{\frac{\text{seed}}{2}} \rceil$, the signing oracle generates $Rq_S \geq 2^{\frac{\text{seed}}{2}}$ random seeds $\text{seed}^i[k] \in 2^{\lambda}$ with $i \in [R]$ and $k \in [q_S]$. This means that with constant probability, we have a collision in these seeds so we have $(i,k)$ and $(i',k') \neq (i,k)$ st. $\text{seed}^i[k] = \text{seed}^{i'}[k']$. This implies $comm_2^i[k] = comm_2^{i'}[k']$ so the pair $((i,k),(i',k'))$ will be found by the algorithm in step 2. For this pair, the probability that $\text{chall}^i[k] = 2$ and $\text{chall}^{i'}[k'] \neq 2 = \frac{2}{9}$ since the challenges are generated from a random function. Therefore, the algorithm will find in step 2 a pair $((i,k),(i',k'))$ st. Equation 2 is satisfied with a constant probability which proves that each iteration of the while loop finds the secret key with a constant probability.

In order to conclude, notice first that each iteration of the while loop performs $q_S = \lceil \frac{1}{R} 2^{\frac{\text{seed}}{2}} \rceil$ sign queries. Regarding the running time, we have to read $O(2^{\frac{\text{seed}}{2}})$ strings to perform an iteration of the while loop. Notice also that most of pairs found in step 3 actually satisfy $\text{seed}^i[k] = \text{seed}^{i'}[k']$ - otherwise we found a collision in $E$ or in $\mathscr{H}_comm$ and such collisions occur with overwhelmingly small probability.

## 5 Fixing the scheme

We present a modification to the protocol that will unable the above attack to work. Changes with the previous protocol are marked in red. Even more, we show that this modified protocol preserves $\lambda$ bits of security.

Optimized Stern's identification scheme for $\text{SD}(n, k, w)$ $[S_{\text{OPT[Salt+Index]}}]$

**Key generation.** Sample $(\mathbf{H}, \mathbf{e}, \mathbf{s} = \mathbf{He}) \leftarrow D_{\text{SD}(n,k,w)}$ where $D_{\text{SD}(n,k,w)}$ is defined in Equation 1. Output $pk = (\mathbf{H}, \mathbf{s})$ and $sk = \mathbf{e}$.

**Protocol.**

**1.** **The prover picks a random *salt* $\in \{0,1\}^{2\lambda}$.** For i from 1 to $R$: the prover picks a random permutation $\pi^i$ acting on $[n]$, picks a random $\text{seed}^i \in S_{\text{seed}}$ and computes the string $\mathbf{u^i} = \mathbf{E}(\text{seed}^\mathbf{i}||salt||\mathbf{i})$ and also defines $\mathbf{y} = \pi^{-1}(\mathbf{u})$. Let also

$$z_1^i = (\pi^i||t^i),\ z_2^i = \mathbf{u}^i = \pi^i(\mathbf{y}^i),\ z_3^i = \pi^i(\mathbf{y}^i + \mathbf{e}^i).$$

The prover then commits to each $z_1^i, z_2^i$ and $z_3^i$ separately by computing $comm_j^i = \mathscr{H}_{comm}(z_j^i)$ for $j \in \{1,2,3\}$ and sends all the $comm_j^i$ for $i \in [R]$ and $j \in \{1,2,3\}$.

**2.** The verifier sends a uniformly random challenge $c = (c^1, \ldots, c^R)$ where each $c^i \in \{1,2,3\}$.

**3.** **The prover reveals *salt*.** For i from 1 to $R$: the prover reveals the following to the verifier:

- If $c^i = 1$, reveal $\text{seed}^i$ and $z_3^i - z_2^i = \pi^i(\mathbf{e})$.
- If $c^i = 2$, reveal $\pi^i$,$\mathbf{y}^i + \mathbf{e}^i$.
- If $c^i = 3$, reveal $\pi^i$, $\text{seed}^i$.

**Verification.** For $i$ from 1 to $R$

- if $c^i = 1$, compute $\mathbf{z_2^i} = \mathbf{E}(\text{seed}^\mathbf{i}||salt||\mathbf{i})$, compute $z_3^i$, check that $|z_3^i - z_2^i| = w$ and that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 2$ and $j = 3$.
- if $c^i = 3$, compute $\mathbf{t}^i = \mathbf{H}(\mathbf{y}^i + \mathbf{e}^i) - \mathbf{s}$. Compute $z_1^i = (\pi^i||t^i)$, $z_3^i = \pi^i(\mathbf{y}^i + \mathbf{e}^i)$ and check that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 1$ and $j = 3$.
- if $c^i = 2$, compute $\mathbf{z_2^i} = \mathbf{E}(\text{seed}^\mathbf{i}||salt||\mathbf{i})$, compute $\mathbf{y}^i = \pi^{-1}(\mathbf{u}^i)$, compute $\mathbf{t}^i = \mathbf{H}(\mathbf{y}^i)$, compute $z_1^i = (\pi^i||t^i)$, and check that $\mathscr{H}_{comm}(z_j^i) = comm_j^i$ for $j = 1$ and $j = 2$.

We have the following

**Theorem 2 (Informal).** *Consider the signature $S_{\mathrm{OPT[Salt+Index]}}$ that uses forE a function that behaves as a random function*[2]*. For any adversary $\mathcal{A}$ performing $q_S$ sign queries and $q_E$ queries to $E.(\cdot)$*

$$\mathrm{ADV}_{S_{\mathrm{OPT[Salt+Index]}}}(\mathcal{A}) \leq \mathrm{ADV}_{S_{\mathrm{Ideal}}}(\mathcal{A}) + \frac{q_S^2}{2^{s_{\mathrm{salt}}}} + \frac{q_E}{2^{s_{\mathrm{seed}}}}.$$

The proof of this theorem will be given in the full version. This shows that we can take seeds of size $\lambda$ and a salt of size $2\lambda$ (even less if we limit the number of sign queries), while preserving the same security as the ideal scheme.

Stern's signature scheme is actually important in the context of post-quantum cryptography since the underlying computational problem is also believed to be hard against quantum computers. It is therefore important to have the above claim also in the quantum setting. We have the following theorem, which we also prove in the full version.

**Theorem 3 (Informal).** *Consider the signature $S_{\mathrm{OPT[Salt+Index]}}$ that uses for $E$ a function that behaves as a random function. For any adversary quantum $\mathcal{A}$ performing $q_S$ (classical) sign queries and $q_E$ (possibly quantum) queries to $E.(\cdot)$*

$$\mathrm{ADV}_{S_{\mathrm{OPT[Salt+Index]}}}(\mathcal{A}) \leq \mathrm{ADV}_{S_{\mathrm{Ideal}}}(\mathcal{A}) + \frac{q_S^2}{2^{s_{\mathrm{salt}}}} + \frac{q_E^2}{2^{s_{\mathrm{seed}}}}.$$

Notice that the last term is $\frac{q_E^2}{2^{s_{\mathrm{seed}}}}$ instead of $\frac{q_E}{2^{s_{\mathrm{seed}}}}$ which means we need to use seeds of size $2\lambda$ if we want to preserve $\lambda$ bits of quantum security.

## 6 Discussion and Conclusion

The takeaway from our attack is that you shouldn't use a deterministic commitment scheme if you possibly commit to the same string several times. This is what happens in particular if you commit to random seeds of size $\lambda$, like in the signature scheme $S_{\mathrm{OPT}}$. How does our work impact the existing proposals of Stern-like signature schemes that use seeds? The first thing to say is that there is no consensus on what should be done. Some proposals seem to be too conservative while some proposals are vulnerable to our attack. Several proposed schemes do use similar optimizations and we now present how our results impact these schemes.

- [BBC+20] that relies on restricted syndrome decoding, uses seeds and hashes both of size 256 bits. We expect that if using seeds of size 128, the performance of their scheme could be improved.

[2] The precise statement of what condition we require will also appear in the full version

– [BGS21] is directly impacted by the presented attack but can be easily corrected with our fix.
– [CHR+20] the MQDSS signature scheme uses seeds of size $\lambda$ bits and commitments of size $2\lambda$. However, the commitments are probabilistic so our attack doesn't work but this is makes the scheme quite inefficient.
– [CVE11] uses 128 bit seeds, however it doesn't specifically commit to a single seed so our attack doesn't work immediately. Nevertheless, our attack at least shows there is a risk here and we still recommend adding the salt here.

For the schemes where we believe our [salt + index] method can be used to improve their efficiency, one needs to prove that this is indeed the case, since each optimized scheme has its own specificities but the proof should essentially follow the proof of Theorems 2,3. For the schemes that use seeds of size $\lambda$ and deterministic commitments of size $2\lambda$, we presented a way to preserve their security at a minimal cost.

## References


AGS11. Carlos Aguilar, Philippe Gaborit, and Julien Schrek. A new zero-knowledge code based identification scheme with reduced communication. In *2011 IEEE Information Theory Workshop*, pages 648–652, 2011.

BBC+20. Marco Baldi, Massimo Battaglioni, Franco Chiaraluce, Anna-Lena Horlemann-Trautmann, Edoardo Persichetti, Paolo Santini, and Violetta Weger. A new path to code-based signatures via identification schemes with restricted errors. *CoRR*, abs/2008.06403, 2020.

Beu20. Ward Beullens. Sigma protocols for mq, pkp and sis, and fishy signature schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020*, pages 183–211, 2020.

BGS21. Loïc Bidoux, Philippe Gaborit, and Nicolas Sendrier. Quasi-cyclic stern proof of knowledge, 2021.

Cha19. André Chailloux. Quantum security of the fiat-shamir transform of commit and open protocols. *IACR Cryptol. ePrint Arch.*, 2019:699, 2019.

CHR+20. Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. Mqdss specifications, 2020.

CVE11. Pierre-Louis Cayrel, Pascal Véron, and Sidi Mohamed El Yousfi Alaoui. A zero-knowledge identification scheme based on the q-ary syndrome decoding problem. In *SAC*, pages 171–186, 2011.

GHHM21. Alex B. Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight adaptive reprogramming in the qrom. Springer-Verlag, 2021.

GRSZ14. Philippe Gaborit, Olivier Ruatta, Julien Schrek, and Gilles Zemor. Ranksign: An efficient signature algorithm based on the rank metric. In Michele Mosca, editor, *Post-Quantum Cryptography*, pages 88–107, 2014.

NIS. NIST. Nist post-quantum standardization, https://csrc.nist.gov/projects/post-quantum-cryptography.

Ste94. Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *Advances in Cryptology — CRYPTO' 93*, pages 13–21, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

Vér97. Pascal Véron. Improved identification schemes based on error-correcting codes. 8(1):57–69, jan 1997.