Moroccan Journal of Algebra and Geometry with Applications
x (xx) (2026), 1–11
Supported by Sidi Mohamed Ben Abdellah University, Fez, Morocco
ISSN: 2820-7114

Cryptanalysis of the Sharafi–Daghigh digital signature scheme

Nour-eddine Rahmani[1]*, Taoufik Serraj[1] and Abdelmalek Azizi[1,2]

[1] *ACSA Laboratory, Department of Mathematics, Faculty of Sciences, Mohammed I University, BV Mohammed VI, 60000 Oujda, Morocco.*
email: nour-eddine.rahmani@ump.ac.ma; t.serraj@ump.ac.ma; abdelmalekazizi@yahoo.fr.
[2] *SupMTI, 67, Immeuble Hallou, Boulevard Med V, 60000 Oujda, Morocco.*

Communicated by Abderrahmane Nitaj
(Received 28 April 2026, Revised 19 June 2026, Accepted 22 June 2026)

Abstract. This paper is devoted to the cryptanalysis of the Ring-LWE-based digital signature scheme proposed by Sharafi and Daghigh. The Sharafi and Daghigh scheme is inspired by the Lindner–Peikert encryption paradigm and adopts a hash-and-sign approach via the Fiat–Shamir transformation. The security claims rely on the assumed hardness of the Ring-LWE and Ring-SIS problems, whose definitions and properties have been well studied in the last two decades. We demonstrate that the scheme admits a public-key-only forgery attack whenever the sum of two attacker-sampled error polynomials remains inside the decoding radius. We also give experimental ratios for the parameter set proposed in the original work.
Key Words: LWE, SIS, Digital Signature, Cryptanalysis, Quantum Safe.
2020 MSC: Primary 94A60; Secondary 68Q17, 11T71.

1 Introduction

The impending threat of quantum computing [19, 4, 15] has made it necessary to move from traditional to post-quantum cryptographic methods. Among the various possible new families, lattice-based cryptography has attracted much attention because of its strong worst-case hardness promises and flexibility in algorithmic use cases. In this field, the Learning With Errors (LWE) problem [16] and its ring-based extension, Ring-LWE [9, 8], are primitive hardness assumptions that form the foundation for numerous contemporary constructions. Lattice-based cryptography has shown a promising alternative to classical public-key constructions with further applications, like Fully-Homomorphic Encryption (FHE), constructing privacy-preserving systems, and much more. NIST has standardized two lattice-based constructions, namely Kyber [5] and Dilithium [2], whose standardized versions are ML-KEM [10] and ML-DSA [11], respectively.

In this paper, we focus on the detailed analysis and breakdown of a Ring-LWE-based digital signature scheme, proposed by Sharafi and Daghigh [18], inspired from the Lindner–Peikert cryptosystem. The scheme is in the hash-and-sign paradigm with the Fiat–Shamir transformation, providing efficiency and compactness. With the use of structured lattices via cyclotomic number fields, it reduces key and signature sizes while it is conjectured to be quantum-resistant in the Quantum Random Oracle Model. The main contribution of this paper is to introduce a flaw-by-design attack that compromises unforgeability of the scheme in chosen-message settings. Rather than depending on inconsistent choice of parameters, noise distributions, or simply the Fiat–Shamir transform, this attack exploits a more profound issue: the signing algorithm lacks sufficient algebraic binding to the secret key. As a result, an adversary having access only to a public key can compute candidate signatures

* Corresponding Author

for any input message, and these candidates verify whenever the aggregate error remains inside the decoding radius.

The purpose of this paper is to clarify and systematically record this intrinsic weakness through theory as well as experimental evidence. Because the attack depends on the decoding behavior of the error sum $e_3 + e_4$, we include concrete success ratios for the original parameter set rather than relying only on an informal small-noise heuristic.

We begin by providing the mathematical and algorithmic prerequisites to understand the scheme and our proposed attack in Section 2. We then restate the signature scheme proposed by Sharafi and Daghigh in Section 3, its correctness and verification, and deconstruct its security reduction in Section 4. A proof of concept implementation of the scheme and our attack is given in Section 5. An extension to plain and module variants is discussed in Section 6. Lastly, a conclusion is given in Section 7.

2 Preliminaries

In this section we revisit some lattice basics and lattice hard problems that most of lattice-based cryptography relies on.

Definition 2.1. A *lattice* $\mathcal{L}$ formally is a discrete subgroup of $\mathbb{R}^n$ for a norm $\|\cdot\|$. A lattice can be represented by a basis $B = \{b_1, \ldots, b_m\}$, for $b_i \in \mathbb{R}^n$.

It is known that, for every lattice $\mathcal{L}$, there is a non-zero vector in $\mathcal{L}$ of minimal Euclidean norm, and finding such a vector is the well-known Shortest Vector Problem.

Definition 2.2 (Shortest Vector Problem (SVP))**.** Given a basis $B \in \mathbb{Z}^{n\times m}$ of a lattice $\mathcal{L}$, find $z \in \mathbb{Z}^m \setminus \{0\}$ such that $\|Bz\|$ is minimal.

Notably, lattice bases are not unique; any unimodular transformation preserves the lattice structure [9]. The quality of a lattice basis is measured on how near-orthogonal and short the basis vectors are.

The process of transforming an initial lattice basis into one with improved geometric properties, termed lattice reduction [7, 17], employs various algorithmic strategies that trade computational efficiency for output quality. For instance, the LLL algorithm [7] achieves polynomial-time complexity but yields vectors with exponential approximation factors relative to the shortest vector. In contrast, HKZ reduction guarantees a basis containing the shortest non-zero vector, albeit at exponential runtime. A detailed comparison of these methods and their trade-offs can be found in the survey [13] and the recent preprint [6]. Notably, advancements in lattice reduction remain an active research area, with recent breakthroughs such as the Progressive BKZ framework [14] and highly optimized implementations like the General Sieve Kernel (G6K) [3] pushing the boundaries of practical efficiency.

For more details and discussion of the hardness of the above problems, we refer the reader to [13, 16].

2.1 Lattice-Based Cryptography

Lattice-based cryptography has emerged as a cornerstone of post-quantum security due to its resilience against quantum algorithms. The foundational work by Ajtai [1] established a critical link between average-case and worst-case hardness of lattice problems, enabling cryptographic constructions based on robust computational assumptions.

2.2 Learning With Errors (LWE)

The Learning With Errors (LWE) problem, introduced by Regev [16], underpins many lattice-based schemes. Let χ be an error distribution over $\mathbb{Z}_q$. For a secret $s \in \mathbb{Z}_q^n$, the LWE distribution $\mathcal{A}_{s,\chi}$ generates samples $(a, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where $b = \langle s, a \rangle + e \bmod q$ with $e \leftarrow \chi$. Two variants exist:

- **Search-LWE**: Recover s given polynomially many samples.
- **Decision-LWE**: Distinguish LWE samples from uniform ones.

LWE's security relies on the conjectured intractability of solving noisy linear equations.

2.3 Ring-LWE

To improve efficiency, Lyubashevsky et al. [8] proposed Ring-LWE, an algebraic variant operating over polynomial rings. Let $R = \mathbb{Z}[X]/\langle f(X) \rangle$ for a cyclotomic polynomial $f(X) = X^n + 1$, where typically n is a power of 2, and let $R_q = \mathbb{Z}_q[X]/\langle f(X) \rangle$. For a secret $s \in R_q$, a Ring-LWE sample $(a, b) \in R_q \times R_q$ satisfies $b = s \cdot a + e \bmod q$, where $e \leftarrow \chi$. The decisional variant requires distinguishing such samples from uniform pairs. Cyclotomic rings are preferred due to their algebraic structure and proven hardness guarantees [8].

2.4 Short Integer Solution (SIS) and Ring-SIS

The SIS problem, introduced by Ajtai [1], involves finding a short non-zero vector $z \in \mathbb{Z}^m$ satisfying $Az = 0 \bmod q$ for a random matrix $A \in \mathbb{Z}_q^{n \times m}$; more precisely, for a small $\beta > 0$, one asks for $z \in \mathbb{Z}^m \setminus \{0\}$ such that $\|z\| \leq \beta$ and $Az = 0 \bmod q$. Its ring analogue, Ring-SIS [9, 8], operates over R_q: given $\mathbf{a} = (a_1, \ldots, a_m) \in R_q^m$, find $z \in R^m \setminus \{0\}$ with $\|z\| \leq \beta$ satisfying $\sum_{i=1}^m a_i z_i = 0 \bmod q$ for a small $\beta > 0$. Both problems are reducible to worst-case lattice problems, forming the basis for collision-resistant hash functions and digital signatures.

3 Review of the Digital Signature Scheme of Sharafi and Daghigh

Most of the content of this section is from the original paper of Sharafi and Daghigh; we recall the most relevant parts to our attack and refer the reader to their original paper for further explanations [18].

The signature scheme under study is built upon the Ring-LWE assumption and inspired by the Lindner–Peikert encryption system. It follows a "hash-and-sign" paradigm using the Fiat–Shamir transformation and operates over a cyclotomic ring $R = \mathbb{Z}[X]/(X^n + 1)$ where n is a power of two. Let $R_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ for a modulus q. The encoding and decoding mechanisms play a critical role in the correctness and security of the scheme.

3.1 Digital Signature Syntax and Security Notion (UF-CMA)

A *digital signature scheme* is a triple of probabilistic polynomial-time algorithms $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ where $\mathsf{KeyGen}(1^\lambda)$ outputs a key pair (pk, sk), $\mathsf{Sign}(sk, m)$ outputs a signature σ, and $\mathsf{Verify}(pk, m, \sigma) \in \{\mathsf{accept}, \mathsf{reject}\}$. Correctness requires that for all messages m,

$$\Pr[\mathsf{Verify}(pk, m, \mathsf{Sign}(sk, m)) = \mathsf{accept}] = 1 - o(1).$$

Unforgeability under chosen-message attack (UF-CMA) is defined by the standard game in which an adversary given pk and access to a signing oracle must not output a valid signature on a fresh message with non-negligible probability.

3.2 Encoding and Decoding: Domain, Range, and Tolerance

Let Σ denote the message space, namely bitstrings of the length used in the original proposal. The scheme uses deterministic maps

$$\text{encode} : \Sigma \to R_q, \qquad \text{decode} : R_q \to \Sigma,$$

where $R_q = \mathbb{Z}_q[X]/\langle X^n + 1\rangle$. Writing a polynomial $g \in R_q$ as $g = \sum_{i=0}^{n-1} g_i X^i$ with centered representatives $g_i \in (-q/2, q/2]$, we use

$$\|g\|_\infty = \max_{0 \le i < n} |g_i|$$

for the coefficient infinity norm. A standard coefficient-wise instantiation of the maps encodes each bit as one of two well-separated residues, for example $0 \mapsto 0$ and $1 \mapsto \lfloor q/2 \rfloor$, and decodes each coefficient to the nearest representative class. The only property needed in our attack is the following error tolerance: there exists a threshold t such that for all $m \in \Sigma$ and all errors e with $\|e\|_\infty \le t$,

$$\text{decode}(\text{encode}(m) + e) = m.$$

In [18], the decoding threshold is set to $t = \lfloor q/4 \rfloor$.

3.3 Key Generation

The signer samples two secret elements $r_1, r_2 \leftarrow \chi_k$ from a discrete Gaussian distribution χ_k over R_q, and a public element $a \in R_q$ is sampled uniformly at random. The public key is computed as:

$$p = r_1 - a r_2 \in R_q.$$

Algorithm 1 Key Generation

1: **Input:** Ring R_q, error distribution χ_k
2: Sample $r_1, r_2 \leftarrow \chi_k$
3: Sample $a \leftarrow R_q$ uniformly
4: Compute $p = r_1 - a r_2$
5: **Output:** Public key (a, p), Secret key r_2

3.4 Signing Procedure

Given a message $m \in \{0,1\}^n$, the signer hashes it and encodes it as $\bar{m} = \text{encode}(H(m)) \in R_q$, using an injective encoding function. The signer samples ephemeral noise terms $e_i \leftarrow \chi_e$ for $i = 1, 2, 3, 4$ and computes:

$$\begin{aligned} C_1 &= p\, e_1 + e_2, \\ C_2 &= p\, a\, e_1 + e_3 + \bar{m}, \\ C_3 &= a\, e_2 + e_4. \end{aligned}$$

Then, the signer computes:

$$\omega = \text{decode}(a r_2 e_1), \quad h = H(m \| H(\omega)).$$

The signature is the tuple (C_1, C_2, C_3, h).

Algorithm 2 Signing

1: **Input:** Message m, secret key r_2, public parameters (a, p)
2: Sample $e_1, e_2, e_3, e_4 \leftarrow \chi_e$
3: Compute $\bar{m} \leftarrow \text{encode}(H(m))$
4: Compute $C_1 \leftarrow p\,e_1 + e_2, \quad C_2 \leftarrow p\,a\,e_1 + e_3 + \bar{m}, \quad C_3 \leftarrow a\,e_2 + e_4$
5: Compute $\omega \leftarrow \text{decode}(ar_2e_1)$
6: Compute $h \leftarrow H(m \| H(\omega))$
7: **Output:** Signature (C_1, C_2, C_3, h)

3.5 Verification Procedure

Given a message m and a signature (C_1, C_2, C_3, h), the verifier computes $\bar{m} = \text{encode}(H(m))$ and checks the following two conditions:

1. $\text{decode}(C_2 - aC_1 + C_3) = \bar{m}$,
2. $h \stackrel{?}{=} H(m \| H(\text{decode}(-C_1)))$.

Algorithm 3 Verification

1: **Input:** Message m, signature (C_1, C_2, C_3, h), public key (a, p)
2: Compute $\bar{m} \leftarrow \text{encode}(H(m))$
3: Compute $\omega' \leftarrow \text{decode}(-C_1)$
4: Compute $h' \leftarrow H(m \| H(\omega'))$
5: Check: $\text{decode}(C_2 - aC_1 + C_3) \stackrel{?}{=} \bar{m}$ **and** $h' \stackrel{?}{=} h$
6: **Accept** if both conditions hold; otherwise **Reject**

Remark 3.1 (On $H(m \parallel H(\omega))$ vs $H(m \parallel \omega)$)**.** The original construction computes the Fiat–Shamir challenge as $h \leftarrow H(m \parallel H(\omega))$. Using $H(\omega)$ provides a fixed-length, canonical input to the hash function and avoids serialisation ambiguities. This choice does not affect our cryptanalysis: in a forgery, the adversary can compute $\omega' = \text{decode}(-C_1)$ from its own C_1 and hence can compute $H(m \parallel H(\omega'))$ just as the verifier does.

3.6 Design Choices: Encoding, Error Distribution, and Noise Thresholds

The encode/decode functions are designed such that

$$\text{decode}(\text{encode}(m) + e) = m$$

for all $m \in \{0,1\}^n$ and small enough noise e in R_q. The error terms are sampled from discrete Gaussian distributions χ_k for secrets and χ_e for ephemeral randomness, both centered at zero. The success of decoding, and hence correctness of verification, is guaranteed with high probability provided the noise remains below a threshold $t = \lfloor q/4 \rfloor$.

These choices aim to balance correctness and security. However, as we demonstrate in the next section, the tight coupling of noise-dependent correctness and Fiat–Shamir commitments creates a critical structural vulnerability.

4 Cryptanalysis: A Flaw-by-Design Attack

4.1 Description of the Attack

Our attack is not based on solving any underlying hard lattice problem such as Ring-LWE or Ring-SIS. Instead, it is a structural attack that exploits the algebraic relationships within the signature generation and verification process. It allows an adversary to generate valid signatures on arbitrary messages without querying the signer, provided that the aggregate error produced by the adversary lies inside the decoding radius.

Our attack replicates the structure of a valid signature without access to the secret key. The adversary proceeds as follows:

1. Choose an arbitrary message $m \in \{0,1\}^n$ and define $\bar{m} = \text{encode}(H(m))$.
2. Sample $e_1, e_2, e_3, e_4 \leftarrow \chi_e$ from the same small-error distribution used by the signer.
3. Compute the following components:

$$C_1 = p\,e_1 + e_2, \qquad C_2 = p\,a\,e_1 + e_3 + \bar{m}, \qquad C_3 = a\,e_2 + e_4.$$

4. Instead of computing ω as the signer does, the adversary computes directly $\omega = \text{decode}(-C_1)$.
5. Compute $h = H(m \| H(\omega))$.

The forged signature (C_1, C_2, C_3, h) passes verification, because:

$$\begin{aligned} C_2 - aC_1 + C_3 &= (pae_1 + e_3 + \bar{m}) - a(pe_1 + e_2) + ae_2 + e_4 \\ &= (pae_1 - ape_1) + e_3 + \bar{m} + e_4 = \bar{m} + (e_3 + e_4). \end{aligned}$$

Thus the first verification condition is reduced to a decoding condition on $e_3 + e_4$. The attack succeeds whenever

$$\|e_3 + e_4\|_\infty \le t,$$

as if this aggregate error had been produced directly by the noise distribution tolerated by the decoding algorithm. Under this condition,

$$\text{decode}(C_2 - aC_1 + C_3) = \bar{m}.$$

Thus, the signature is accepted by the verifier, even though the message was never signed by the legitimate party.

4.2 Proof of Insecurity

We now formally prove that the scheme fails to satisfy the UF-CMA security requirement. Specifically, we show that a probabilistic polynomial-time adversary can generate a valid signature on an arbitrary message without querying the signing oracle.

Theorem 4.1. The Ring-LWE-based signature scheme of Sharafi and Daghigh is insecure in the UF-CMA model. There exists a polynomial-time adversary that forges signatures on arbitrary messages with success probability

$$\Pr_{e_3, e_4 \leftarrow \chi_e} [\|e_3 + e_4\|_\infty \le t],$$

and the adversary can amplify this probability by re-sampling its own error terms.

Proof. Let $\mathcal{A}$ be an adversary who does not make any signing queries. On input the public key (a, p), $\mathcal{A}$ chooses a message $m \in \{0,1\}^n$ to be signed and computes $\bar{m} = \text{encode}(H(m))$. Then it samples $e_1, e_2, e_3, e_4 \leftarrow \chi_e$ and computes:

$$C_1 = p\,e_1 + e_2, \qquad C_2 = p\,a\,e_1 + e_3 + \bar{m}, \qquad C_3 = a\,e_2 + e_4.$$

$\mathcal{A}$ checks whether $\text{decode}(C_2 - aC_1 + C_3)$ equals $\bar{m}$ or not. If not, $\mathcal{A}$ repeats from step 2. This loop terminates with probability governed exactly by the event $\|e_3 + e_4\|_\infty \leq t$, because $C_2 - aC_1 + C_3 = \bar{m} + e_3 + e_4$.

Next, the adversary computes $\omega = \text{decode}(-C_1)$ and outputs the signature (C_1, C_2, C_3, h) where $h = H(m \| H(\omega))$. This signature passes verification because:

$$\text{decode}(C_2 - aC_1 + C_3) = \text{decode}(\bar{m} + e_3 + e_4) = \bar{m},$$

provided that $e_3 + e_4$ lies within the decoding radius. For the original parameters used in Section 5, this event occurred in all reported trials.

The second verification condition is satisfied by construction since $\omega = \text{decode}(-C_1)$, so the verifier recomputes $h' = h$. Thus, the adversary succeeds in forging a signature on a new message without access to any valid signature or the private key, violating the UF-CMA security definition. □

5 Experimental Proof of Concept

To confirm the viability of the attack in practice, we implemented a prototype of the signature scheme and the flaw-by-design forgery algorithm (Algorithm 4) in SageMath. The primary goal was to demonstrate that for any chosen message, it is possible to generate a valid signature that passes verification without access to the private key.

Algorithm 4 Flaw-by-Design Forgery Attack

Require: Public key (a, p), message m to be signed, decoding function decode, error distribution χ_e
Ensure: Forged signature (C_1, C_2, C_3, h) on message $m \in \{0,1\}^n$

1: Compute $\bar{m} \leftarrow \text{encode}(H(m))$
2: Sample $e_1, e_2, e_3, e_4 \leftarrow \chi_e$
3: Compute $C_1 \leftarrow p\,e_1 + e_2$
4: Compute $C_2 \leftarrow p\,a\,e_1 + e_3 + \bar{m}$
5: Compute $C_3 \leftarrow a\,e_2 + e_4$
6: Compute $h \leftarrow H(m \| H(\text{decode}(-C_1)))$
7: **return** (C_1, C_2, C_3, h)

Experimental platform

All experiments were conducted on a single machine with the following specifications:

CPU	Intel Core i7-class processor
RAM	16 GB
OS	Ubuntu 22.04 LTS (64-bit)
SageMath	9.6
Python	3.10

Reported runtimes correspond to average wall-clock time over the stated trials.

Setup

We used the parameter set recommended in the original proposal:

- Ring dimension: $n = 256$
- Modulus: $q = 7681$
- Error distribution: centered discrete Gaussian χ_e with standard deviation $\sigma = 2.5$
- Decoding threshold: $t = \lfloor q/4 \rfloor = 1920$

We generated public keys using the original scheme key generation algorithm, but all signatures were produced independently using the attack with only the public key and a random message as inputs.

Procedure

For each of 2000 randomly selected messages and each tested dimension:

1. We computed $\bar{m} = \text{encode}(H(m))$.
2. Sampled noise vectors $e_1, e_2, e_3, e_4 \leftarrow \chi_e$.
3. Computed (C_1, C_2, C_3) using the public key and the formulas of the attack.
4. Checked whether $\text{decode}(C_2 - aC_1 + C_3) = \bar{m}$.
5. Recorded the coefficient norm $\|e_3 + e_4\|_\infty$.

Code availability

A fully reproducible proof-of-concept implementation of the attack is provided as supplementary material in the notebook: `https://github.com/ndrahmani/attack_sharafi_daghigh`.

Results

Table 1 gives the observed success ratios. The success criterion is exactly the condition used in the proof, namely $\|e_3 + e_4\|_\infty \leq t$ with $t = 1920$.

Table 1: Empirical success ratios for the forgery condition $\|e_3 + e_4\|_\infty \leq t$, with $q = 7681$, $t = 1920$, and $\sigma = 2.5$.

Dimension n	Trials	Successful forgeries	Success ratio
128	2000	2000	1.000000
256	2000	2000	1.000000
512	2000	2000	1.000000
1024	2000	2000	1.000000

In the same experiments, the largest observed value of $\|e_3 + e_4\|_\infty$ was 19, far below the decoding threshold $t = 1920$. This explains why the vector condition is not negligible for the original parameters: each component of $e_3 + e_4$ is the sum of two very small centered discrete-Gaussian samples, while the accepted interval is several orders of magnitude wider. If an implementation used a much

narrower decoding radius or a substantially wider error distribution, the ratio in Table 1 would have to be recomputed for that implementation.

These results confirm that the attack is not merely theoretical but also practical and efficient. It does not rely on any oracle access, nor does it attempt to recover the secret key. Instead, it forges valid signatures by exploiting the deterministic algebraic structure of the scheme.

Interpretation

The experimental findings corroborate the theoretical proof of insecurity: the attacker can produce an arbitrary number of valid signatures on any messages of their choosing. Since the verification condition only relies on smallness of the decoding error, and the error is attacker-controlled, the correctness mechanism becomes a source of forgery rather than security.

This shows that the signature scheme is not only mathematically broken for the stated condition but also practically exploitable for the original parameter set.

6 Remarks on Plain and Module Variants

The algebraic cancellation used in the attack is not specific to a secret-key recovery problem or to a reduction from a lattice problem. If a plain-LWE or module-LWE variant keeps the same verification pattern, the same formal cancellation gives

$$C_2 - aC_1 + C_3 = \bar{m} + e_3 + e_4$$

coordinate-wise.

However, the probability of a successful forgery in such variants depends on the full vector condition that every relevant coordinate of $e_3 + e_4$ lies inside the decoding radius. Therefore, for a concrete plain or module instantiation, the success probability must be measured or bounded using its actual dimension, decoding rule, and error distribution. The experimental ratios in Table 1 support this condition for the Ring-LWE parameter set considered in the original Sharafi–Daghigh proposal; they should not be read as a universal probability statement for all possible module dimensions and parameter choices.

Consequently, our unconditional conclusion concerns the published Ring-LWE scheme under the tested parameters. The same algebraic weakness should be checked carefully before reusing this verification pattern in plain-LWE or module-LWE variants.

7 Conclusion

We have demonstrated a key-only attack arising from a structural flaw in the design of a Ring-LWE-based digital signature scheme that invalidates its unforgeability for the published parameter set. Our attack exploits the fact that an attacker does not need to know the signing keys to generate valid signatures: the generated error terms can be computed by the attacker knowing only the verification key, since these error terms are not bound to the secret signing key.

The decisive condition is whether $e_3 + e_4$ remains inside the decoding radius. We therefore supported the proof with concrete success ratios, which showed success in all reported trials for the original parameters. Similar constructions over plain or module lattices require separate parameter-specific experiments or bounds, because the full vector decoding condition changes with the dimension and the decoding rule.

Declaration of Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Ajtai, Generating hard instances of lattice problems (extended abstract), *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (STOC '96), pp. 99–108, Association for Computing Machinery, New York (1996).

[2] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler and D. Stehlé, CRYSTALS-Dilithium: A lattice-based digital signature scheme, *Transactions on Cryptographic Hardware and Embedded Systems* **2018** (2018).

[3] M.R. Albrecht, L. Ducas, G. Herold, E. Kirshanova, E.W. Postlethwaite and M. Stevens, The general sieve kernel and new records in lattice reduction, in: Y. Ishai and V. Rijmen (eds.), *Advances in Cryptology – EUROCRYPT 2019*, pp. 717–746, Springer, Cham (2019).

[4] L.K. Grover, A fast quantum mechanical algorithm for database search, *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (STOC '96), pp. 212–219, Association for Computing Machinery, New York (1996).

[5] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J.M. Schanck, P. Schwabe, G. Seiler and D. Stehlé, CRYSTALS-Kyber: A CCA-secure module-lattice-based KEM, *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 353–367 (2018).

[6] K. de Boer and W. van Woerden, Lattice-based cryptography: A survey on the security of the lattice-based NIST finalists, Cryptology ePrint Archive, Paper 2025/304 (2025). https://eprint.iacr.org/2025/304

[7] A.K. Lenstra, H.W. Lenstra and L.M. Lovász, Factoring polynomials with rational coefficients, *Math. Ann.* **261** (1982), 515–534.

[8] V. Lyubashevsky, C. Peikert and O. Regev, On ideal lattices and learning with errors over rings, *J. ACM.* **60**(6) (2013), Article 43.

[9] D. Micciancio and O. Regev, Lattice-based cryptography, in: D.J. Bernstein, J. Buchmann and E. Dahmen (eds.), *Post-Quantum Cryptography*, pp. 147–191, Springer, Berlin, Heidelberg (2009).

[10] National Institute of Standards and Technology, Module-Lattice-Based Key Encapsulation Mechanism Standard, NIST FIPS 203, Department of Commerce, Washington, D.C. (2024).

[11] National Institute of Standards and Technology, Module-Lattice-Based Digital Signature Standard, NIST FIPS 204, Department of Commerce, Washington, D.C. (2024).

[12] A. Nitaj and M. Seck, A new public key encryption scheme based on the cubic Pell curve using encoding functions, Moroccan J. Algebra Geom. Appl. 4(1) (2025), 89–107.

[13] C. Peikert, A decade of lattice cryptography, *Found. Trends Theor. Comput.* **10**(4) (2016), 283–424.

[14] Z. Zhao and J. Ding, Practical improvements on BKZ algorithm, in: S. Dolev, E. Gudes and P. Paillier (eds.), *Cyber Security, Cryptology, and Machine Learning*, pp. 273–284, Springer, Cham (2023).

[15] O. Regev, An efficient quantum factoring algorithm, *Journal of the ACM* **72**(1) (2025), Article 10.

[16] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing* (STOC '05), pp. 84–93, Association for Computing Machinery, New York (2005).

[17] C.P. Schnorr, A hierarchy of polynomial time lattice basis reduction algorithms, *Theor. Comput. Sci.* **53**(2) (1987), 201–224.

[18] J. Sharafi and H. Daghigh, A Ring-LWE-based digital signature inspired by Lindner–Peikert scheme, *J. Math. Cryptol.* **16**(1) (2022), 205–214.

[19] P.W. Shor, Algorithms for quantum computation: discrete logarithms and factoring, *Proceedings 35th Annual Symposium on Foundations of Computer Science* (1994) pp. 124–134.

[20] A. Tall, A fast and SPA secure scalar multiplication for elliptic curve cryptography, Moroccan J. Algebra Geom. Appl. 2(2) (2023), 209–217.