

# Design-Level Cryptanalysis of the ICCS NGCC Round-1 Public-Key Candidates<sup>1</sup>

Zhenyu Xiong      Mingsheng Wang

`xiongzhenyu@iie.ac.cn`    `wangmingsheng@iie.ac.cn`

State Key Laboratory of Cyberspace Security Defense,  
Institute of Information Engineering, Chinese Academy of Sciences,  
Beijing, China

October 1, 2026

---

<sup>1</sup>Produced with the assistance of a large language model (Anthropic Claude Opus 5.5).

# Contents

|                                                                   |           |
|-------------------------------------------------------------------|-----------|
| <b>Overview</b>                                                   | <b>1</b>  |
| <b>Relation to the NIST PQC design lessons</b>                    | <b>4</b>  |
| <b>1 LoongKEM: an IND-CPA break by sub-ring projection</b>        | <b>6</b>  |
| 1.1 Introduction . . . . .                                        | 7         |
| 1.2 The LoongKEM Scheme . . . . .                                 | 8         |
| 1.3 Analysis: The Sub-Ring Projection . . . . .                   | 8         |
| 1.4 The Attack . . . . .                                          | 8         |
| 1.5 Experiments . . . . .                                         | 9         |
| 1.6 Countermeasure . . . . .                                      | 10        |
| 1.7 Conclusion . . . . .                                          | 10        |
| <b>2 CheetahKEM: an IND-CPA break by sub-ring projection</b>      | <b>11</b> |
| 2.1 Introduction . . . . .                                        | 12        |
| 2.2 The CheetahKEM Scheme . . . . .                               | 12        |
| 2.3 The Sub-Ring Projection . . . . .                             | 13        |
| 2.4 The Attack . . . . .                                          | 13        |
| 2.5 Experiments . . . . .                                         | 13        |
| 2.6 Countermeasure . . . . .                                      | 14        |
| 2.7 Conclusion . . . . .                                          | 14        |
| <b>3 Facto-DSA: forgery from the public key</b>                   | <b>15</b> |
| 3.1 Introduction . . . . .                                        | 16        |
| 3.2 The target scheme . . . . .                                   | 16        |
| 3.3 Analysis: recovering the kernel from the public key . . . . . | 17        |
| 3.4 The attack algorithm . . . . .                                | 17        |
| 3.5 Attack flow . . . . .                                         | 18        |
| 3.6 Parameters, cost, and effect . . . . .                        | 18        |
| 3.7 Open-source implementation . . . . .                          | 18        |
| 3.8 Countermeasure . . . . .                                      | 18        |
| 3.9 Conclusion . . . . .                                          | 19        |
| <b>4 Origami: universal forgery from the public key</b>           | <b>20</b> |
| 4.1 Introduction . . . . .                                        | 21        |
| 4.2 Preliminaries . . . . .                                       | 22        |
| 4.3 The Verification Map Is Trapdoor-Free . . . . .               | 22        |
| 4.4 Public-Key Universal Forgery . . . . .                        | 23        |
| 4.5 Experimental Verification . . . . .                           | 23        |
| 4.6 Countermeasure . . . . .                                      | 24        |
| 4.7 Conclusion . . . . .                                          | 24        |
| <b>5 Polar-KEM: keyless decapsulation</b>                         | <b>25</b> |
| 5.1 Introduction . . . . .                                        | 26        |
| 5.2 Preliminaries . . . . .                                       | 27        |
| 5.3 The Keyless Decapsulation . . . . .                           | 27        |
| 5.4 Experimental Verification . . . . .                           | 28        |
| 5.5 Specification-Level Defects . . . . .                         | 28        |
| 5.6 Countermeasure . . . . .                                      | 29        |

|          |                                                            |           |
|----------|------------------------------------------------------------|-----------|
| 5.7      | Conclusion . . . . .                                       | 29        |
| <b>6</b> | <b>MAMBA-NIKE: static-key recovery from reconciliation</b> | <b>30</b> |
| 6.1      | Introduction . . . . .                                     | 31        |
| 6.2      | Preliminaries . . . . .                                    | 32        |
| 6.3      | The Responder Is a Reaction Oracle . . . . .               | 33        |
| 6.4      | Scalar-Label Reconciliation . . . . .                      | 33        |
| 6.5      | Experimental Verification . . . . .                        | 33        |
| 6.6      | Scope . . . . .                                            | 34        |
| 6.7      | Countermeasure . . . . .                                   | 34        |
| 6.8      | Conclusion . . . . .                                       | 35        |
| <b>7</b> | <b>Shuttle: key recovery from a rejection-free sampler</b> | <b>36</b> |
| 7.1      | Introduction . . . . .                                     | 37        |
| 7.2      | Preliminaries . . . . .                                    | 41        |
| 7.3      | Leakage of the Rejection-Free Sampler . . . . .            | 42        |
| 7.4      | The Sign-Alignment Correlation Attack . . . . .            | 43        |
| 7.5      | Experimental Verification . . . . .                        | 45        |
| 7.6      | Per-Parameter Attack Details . . . . .                     | 46        |
| 7.7      | Analysis of the Security Reduction . . . . .               | 46        |
| 7.8      | Countermeasures . . . . .                                  | 47        |
| 7.9      | Conclusion . . . . .                                       | 47        |
| 7.10     | Deferred Proofs . . . . .                                  | 47        |
| 7.11     | Reproduction . . . . .                                     | 49        |
| <b>8</b> | <b>BiT: key recovery from a shared sign</b>                | <b>50</b> |
| 8.1      | Introduction . . . . .                                     | 51        |
| 8.2      | Preliminaries . . . . .                                    | 53        |
| 8.3      | The Shared-Sign Leak . . . . .                             | 54        |
| 8.4      | The Shared-Sign Deconvolution Attack . . . . .             | 55        |
| 8.5      | Experimental Verification . . . . .                        | 56        |
| 8.6      | Scope . . . . .                                            | 57        |
| 8.7      | Countermeasures . . . . .                                  | 58        |
| 8.8      | Conclusion . . . . .                                       | 58        |
| 8.9      | Deferred Proofs . . . . .                                  | 58        |
| 8.10     | Reproduction . . . . .                                     | 59        |
|          | <b>Bibliography</b>                                        | <b>61</b> |

# Overview

The ICCS Next-Generation Commercial Cryptography (NGCC) round-1 call received 84 public-key candidates (34 signatures, 41 KEMs, 9 key-exchange schemes) and invites public evaluation. This report focuses on *design-level* weaknesses: flaws in the design method, the ring or parameter choice, the hardness analysis, or the security-strength argument. These are the issues that a submitter cannot close with a local code fix; they require rethinking the construction or the parameters.

**Scope, and why implementation defects are excluded here.** We deliberately keep implementation-level defects out of the main text. Such defects (missing range checks, inert re-encryption checks, non-canonical encodings, memory-safety crashes, and the like) are expected during a multi-round call, are fixed between rounds, and are, in several cases, inherited from widely used reference code (for example Fujisaki–Okamoto and sampler routines derived from NIST PQC reference implementations) rather than being specific to the scheme under review. Reporting them as if they were design results would conflate library-inherited patterns with genuine design flaws and would not help the community judge which *designs* are sound. Where we found implementation-level issues, we reported them to the submitting teams through the official comment channel; they are not detailed here.

## Design-level findings

Table 1 lists the design-level weaknesses, grouped by the design decision that caused them.

**Ring and parameter choice.** LoongKEM and CheetahKEM keep the cyclotomic degree small and non-power-of-two, so  $X^N + 1$  is reducible over  $\mathbb{Z}$ ; a sub-ring projection then maps the public relation onto a genuinely lower-degree module-LWE instance whose lattice cost falls far below the claimed level. This is a choice-of-ring / choice-of-parameter flaw, not a coding error: any faithful implementation of the specified parameters is affected. (These are semantic-security, i.e. decision, breaks; full recovery of the original key is not claimed.)

**The public data is the trapdoor.** Three designs place in public data something that was meant to stay secret. Origami’s verification map applies no secret input/output transform, so a forger can solve for a valid signature from the public key alone. Polar-KEM regenerates its “secret” isometry deterministically from a seed shipped in the public key, so decapsulation needs no private key at all. Facto-DSA’s public key determines the hidden kernel through a restricted algebraic computation, again yielding a public-key-only forgery (demonstrated at security level 128; higher levels are bounded by memory).

**Category / claim mismatch.** MAMBA-NIKE is a non-interactive key exchange whose responder reuses a static secret key, yet the design applies no chosen-ciphertext transform binding the transcript. Its passive-security theorem stands; the design simply does not provide the active, static-key-reuse security that the NIKE setting requires, so an active peer recovers the static key.

**Sampler design.** Shuttle and BiT recover an equivalent signing key from ordinary valid signatures because the rejection sampler’s sign handling leaves a secret-dependent correlation across the response. Closing this requires redesigning the sampler, not patching a line: the BiT team has itself moved away from its triangular bimodal sampler, which confirms the issue is at the level of the sampling design.

Table 1: Design-level weaknesses in NGCC round-1 public-key candidates, grouped by the design decision responsible. All are flaws a local code fix cannot close.

| Scheme     | Design-level flaw                                                              | Consequence                                                   |
|------------|--------------------------------------------------------------------------------|---------------------------------------------------------------|
| LoongKEM   | Reducible ring ( $X^N+1$ , non-power-of-two $N$ ) admits a sub-ring projection | IND-CPA broken below the claimed level (3 of 4 sets)          |
| CheetahKEM | Same reducible-ring projection ( $X^{640}+1$ splits)                           | IND-CPA broken at $\approx 2^{53}$ vs claimed 128             |
| Facto-DSA  | Public key determines the hidden kernel algebraically                          | Public-key-only forgery (level 128; higher bounded by memory) |
| Origami    | Verification map has no secret transform                                       | Public-key-only universal forgery                             |
| Polar-KEM  | “Secret” isometry regenerated from a public seed                               | Keyless decapsulation                                         |
| MAMBA-NIKE | Static-key reuse with no chosen-ciphertext transform                           | Active recovery of the static key                             |
| Shuttle    | Rejection-free sampler leaves a sign-alignment correlation                     | Equivalent-key recovery + forgery                             |
| BiT        | Bimodal sign shared across the response                                        | Equivalent-key recovery + forgery                             |

## Methodology and limitations

The analysis was carried out with the assistance of a large language model and verified in a falsify-first manner: every candidate lead was first attacked with attempts to disprove it, and only leads that survived were turned into code that runs against the *unmodified* reference implementation, using public data only, with the secret key read solely to score a finished result. We state the limits of this method plainly. Automated analysis is effective at reachable, structural, and code-level questions, but the design, parameter, and hardness questions that matter most for selection still need careful human review; some of the strongest design issues above rest on lattice-estimator and algebraic analysis that a reader should check independently. As one illustration of the falsify-first discipline, a suspected timing channel in PolarLAC’s non-constant-time decapsulation was investigated in depth and found *not* exploitable into a key recovery, so it is not claimed as a break.

**Priority and prior public disclosure.** Most of the findings in this report were publicly disclosed by others before it was written, and we claim no priority over those disclosures. We record the earlier reports explicitly, both here and at the head of each affected chapter. Every finding here was obtained from our own analysis of the submitted specification and reference code, and where it helps to date the overlap we give the date of our earliest artifact, so that it can be judged against the record rather than asserted.

For **Polar-KEM**, LittleQ reported recovery of the shared secret from  $(pk, ct)$  alone on 22 September 2026, ahead of our artifact of 23 September. For **CheetahKEM**, XuHaomeng reported the Chinese-Remainder splitting of  $X^{640}+1$  on 22 September 2026, ahead of our artifact of 24 September. For **Facto-DSA**, kriskwiatkowski1 refuted the  $q^n$  structural estimate of §3.2.5 on 22 September 2026, ahead of our artifact of 23 September. For **Origami**, Peigen Li raised the structural problem on 22 September 2026, and on 24 September 2026 Jintai Ding and Hugo Louiso announced a public-key-only universal forgery on the NIST `pqc-forum`, with Pierre Pébereau independently reporting the same break and releasing code the same day. We had reached the same attack independently, from our own analysis of the specification and reference code, but those reports carry the earlier public record and we claim no priority. For **Shuttle**, Martin Feussner

reported the sampler sign-handling leak on the NIST `pqc-forum` around 25 September 2026, independently confirmed by Markku-Juhani O. Saarinen on <https://ngcc.dev>; our artifact is dated 23 September. For **LoongKEM**, Sun Shuzhou reported the reducible-ring projection on 28 September 2026, concurrently with this report, and Tianyuan Xie reported a full Loong128 decapsulation break on 29 September 2026; our artifact is dated 24 September.

For Polar-KEM, CheetahKEM and Facto-DSA the public report clearly precedes our work. For Shuttle and LoongKEM our artifact is the earlier of the two, but it is a private file timestamp whereas theirs is a public posting, which does not establish priority; we give the dates only to explain how the overlap arose. In all six cases the corresponding chapters should be read as independent verification, and we claim no priority.

To our knowledge, at the time of writing no prior public report existed for the **MAMBA-NIKE** and **BiT** findings; readers should nevertheless consult <https://ngcc.dev> and the ICCS PKC forum, which are updated continually, before attributing priority to this report for those two.

## Relation to the NIST PQC design lessons

None of the design-level weaknesses in this report is a new *kind* of failure. Each one recurs a pitfall that the NIST Post-Quantum Cryptography process already identified and, in several cases, used to eliminate a candidate. We make the correspondence explicit, both to place our findings in context and to show what a design-level (rather than implementation-level) review looks like: the question is never “does the submitted code have a bug”, but “does the construction, the ring, the parameters, or the hardness argument repeat a mistake the field has already paid for.” NIST treated design and parameter failures as disqualifying (Rainbow and GeMSS were dropped; SIKE was withdrawn after a pure mathematical break) while implementation bugs were simply fixed between rounds. This report adopts the same division of importance.

### **Ring and parameter choice: subfield / sub-ring attacks (LoongKEM, CheetahKEM).**

The security of NTRU- and module-LWE-type schemes depends on the ring. The overstretched-NTRU subfield attacks of Albrecht–Bai–Ducas (CRYPTO 2016) and Kirchner–Fouque (EUROCRYPT 2017), later sharpened by the “fatigue point” analysis of Ducas–van Woerden (ASIACRYPT 2021), showed that a poor ring or modulus choice opens a projection to a lower-degree subfield where the lattice problem is far easier. NTRU Prime (Bernstein–Chuengsatiansup–Lange–van Vredendaal, SAC 2017) deliberately avoids cyclotomic rings with subfield and Galois structure for exactly this reason, and the NIST lattice standards ML-KEM (Kyber) and ML-DSA (Dilithium) use power-of-two cyclotomics  $X^n + 1$  so that no small proper subring is available. LoongKEM and CheetahKEM instead choose non-power-of-two degrees, making  $X^N + 1$  reducible over  $\mathbb{Z}$  and reintroducing precisely the sub-ring projection surface these results warned against. This is a choice-of-ring decision, not a coding error.

### **Structured public keys: MinRank / structural key recovery (Facto-DSA, Origami).**

Multivariate and structured-trapdoor signatures have repeatedly fallen to attacks that recover the secret directly from the public key. SFLASH was broken by Dubois–Fouque–Shamir–Stern (CRYPTO 2007); the HFE family fell to MinRank (Tao–Petzoldt–Ding, CRYPTO 2021); and the NIST Round-3 finalist Rainbow was broken by Beullens (“Breaking Rainbow Takes a Weekend on a Laptop”, CRYPTO 2022) by exploiting hidden layer structure left visible in the public key. The lesson is that a public map whose hidden algebraic structure is recoverable (through rank defects, differentials, or a restricted elimination) is broken regardless of how faithfully it is implemented. Facto-DSA’s public key determines its hidden kernel by exactly such a restricted algebraic computation, and Origami is the degenerate case in which the verification map hides no secret structure at all, so a forgery follows from the public key alone.

**Key reuse without a CCA transform: reaction attacks (MAMBA-NIKE).** The reason every NIST KEM is required to be IND-CCA through a Fujisaki–Okamoto transform, rather than merely IND-CPA, is the line of key-reuse and reaction attacks on lattice key exchange: Fluhrer (2016) on ring-LWE key-share reuse, and the signal-leakage / key-mismatch attacks of Ding–Alsayigh–Saraswathy–Fluhrer–Lin (2017) and successors, which recover a reused secret from the responder’s reconciliation behaviour. NewHope and the original Ding key exchange were therefore specified as ephemeral-only. A non-interactive key exchange inherently reuses a static key and so *must* carry active, chosen-ciphertext security. MAMBA-NIKE applies no such transform, and the resulting active recovery of its static key is the textbook realisation of this NIST-era lesson.

**Transcript-independent rejection sampling (Shuttle, BiT).** Fiat–Shamir with aborts (Lyubashevsky, ASIACRYPT 2009 and EUROCRYPT 2012), the basis of Dilithium, uses rejection

sampling for one purpose: to make the released signature distribution independent of the secret key. When the sampler is mis-designed, the transcript leaks the key. This is precisely the history of BLISS, the Bimodal Lattice Signature Scheme (Ducas–Durmus–Lepoint–Lyubashevsky, CRYPTO 2013), whose bimodal Gaussian sampling was repeatedly shown to leak the secret (Groot Bruinderink et al., CHES 2016; Espitau–Fouque–Gérard–Tibouchi, CCS 2017). BiT (“Bimodal Triangular”) is a direct descendant of the bimodal idea, and both BiT and Shuttle leak an equivalent key through the sign handling in their rejection sampler, recoverable from ordinary valid signatures. The fix is to redesign the sampler so its transcript is secret-independent, which is why the BiT team’s own move away from its triangular bimodal sampler is the correct design-level response.

**The public data must not determine the secret (Polar-KEM).** A recurring hygiene lesson from the NIST process, reinforced by the domain- and key-binding requirements of the FO transform, is that nothing recomputable from public data may stand in for a secret. Polar-KEM violates this directly by regenerating its “secret” isometry from a seed carried in the public key, so decapsulation succeeds with no private key.

**Summary.** Read together, our findings say that several NGCC round-1 candidates repeat design decisions the NIST PQC process already ruled out: reducible rings with exploitable subrings, structured public keys that leak their trapdoor, static-key reuse without a CCA transform, and secret-dependent rejection sampling. These are the issues worth the community’s cryptanalytic time; the implementation defects we also observed are, by comparison, transient and are handled through the comment channel.

# LoongKEM: an IND-CPA break by sub-ring projection

LoongKEM is a semi-structured LWE (SLWE) key encapsulation mechanism submitted to the ICCS NGCC round-1 call, whose public key mixes block-anti-circulant ring blocks with fully unstructured blocks and whose design deliberately keeps the ring degree small to avoid Kyber-like structure. We show that this ring choice is the weakness: three of its four parameter sets work over  $\mathbb{Z}_q[X]/(X^N + 1)$  with  $N$  not a power of two.

Whenever  $N$  has an odd factor,  $X^N + 1$  is reducible over  $\mathbb{Z}$  and there is a  $\mathbb{Z}_q$ -algebra homomorphism to a lower-degree ring  $X^d + 1$  that sends small elements to small elements. Applying it to the structured transmitted rows projects the public relation onto a genuine, much lower-dimensional module-LWE instance whose lattice cost is far below the claimed security, breaking decision-SLWE and hence IND-CPA. We verify the projection identity exactly on a real key, recover the projected secret, and validate the recovery pipeline at reduced scale.

Against the reference parameters the projected instance's lattice cost is about  $2^{77}$  for Loong128 (primal-uSVP  $2^{76.6}$ ,  $\beta = 168$ ;  $2^{73.2}$  under a BDD model),  $2^{107}$  for Loong384, and  $2^{208}$  for Loong512, against the claimed 128, 384, and 512 bits; the only power-of-two set, Loong256 ( $X^{16} + 1$  irreducible), is immune, so IND-CPA is broken for 3 of the 4 parameter sets. We verify the projection identity exactly on a real Loong128 key and validate the recovery pipeline by exact reduced-scale recoveries at dimensions 40 and 55 (block sizes 20 and 30); on our machine (104 cores, 251 GB RAM) the projection identity check and these reduced-scale recoveries complete in a few seconds (the lattice reduction's own working set is under 1 MB; the driving computer-algebra process peaks near 200 MB), while the full-scale recovery at the true projected dimension is not run since the projected cost already falls far below the claimed security level. The fix is a power-of-two cyclotomic at every level. Attack source is available.<sup>1</sup>

---

**Concurrent public disclosure.** An equivalent structural observation was reported publicly while this report was being prepared. On 28 September 2026 Sun Shuzhou reported on the ICCS PKC forum that  $X^N + 1$  factors over the integers for  $N = 12, 20, 24$  (Loong128/384/512) into coprime factors modulo  $q = 8191$ , so reducing the public-key block row modulo a factor yields a smaller SLWE instance, costing about  $2^{73.0}$ ,  $2^{103.2}$  and  $2^{203.3}$  against claimed 147, 398 and 521 bits, with Loong256 unaffected because  $X^{16} + 1$  is irreducible over the integers. Separately, on 29 September 2026 Tianyuan Xie reported a full decapsulation break of Loong128 from the public key and ciphertext alone. We claim no priority over those reports.

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/loongkem-subring-projection>.

## 1.1 Introduction

The security of a ring- or module-LWE scheme is inseparable from its ring. Power-of-two cyclotomics  $\mathbb{Z}_q[X]/(X^N + 1)$  with  $N = 2^k$  are the safe default precisely because  $X^N + 1 = \Phi_{2N}$  is irreducible over  $\mathbb{Z}$ , so there is no proper quotient into which the hard instance can be projected. When a design leaves this default, either to shrink parameters or to blend structured and unstructured blocks, it must ensure the new ring exposes no cheaper sub-problem. LoongKEM makes exactly this choice and does not.

LoongKEM [40] is a semi-structured LWE KEM: its public matrix interleaves block-anti-circulant ring blocks with fully unstructured blocks, and the specification keeps the ring degree  $N$  small so that “LoongKEM does not possess strong algebraic structure similar to Kyber”, noting that the authors do not know how to exploit the residual structure. We exploit it. Three of the four parameter sets use  $N \in \{12, 20, 24\}$ , none a power of two, so  $X^N + 1$  is reducible and a sub-ring projection collapses the structured public rows to a genuine low-dimensional module-LWE instance far below the claimed security.

**Our results.** We give the projection and its consequences (Section 1.3), the attack (Section 1.4), and the experiments (Section 1.5). On a real Loong128 key the projection identity holds exactly and the projected instance is a small-noise module-LWE of secret dimension 208 with error standard deviation about 4.6; its lattice cost is far below  $2^{128}$ . The recovery pipeline is validated at reduced scale. Loong256 alone is immune.

### 1.1.1 Contributions

- We show that the LoongKEM ring is reducible for three of four sets and give the  $\mathbb{Z}_q$ -algebra homomorphism  $\psi$  that projects the structured public rows to a lower-degree module-LWE instance while preserving smallness (Proposition 1.1).
- We compute the projected lattice cost per set ( $\approx 2^{77} / 2^{107} / 2^{208}$  vs 128/384/512), breaking decision-SLWE and hence IND-CPA for Loong128/384/512.
- We verify the projection exactly on a real key, recover the projected secret, and validate the primal-embedding recovery at reduced scale; the power-of-two set is confirmed immune.

### 1.1.2 Related Work

Structural attacks that exploit the multiplicative structure of the ring are classical: the sub-field and overstretched-NTRU attacks of Albrecht, Bai and Ducas [41] and Kirchner and Fouque [42] recover keys from a dense sub-lattice induced by a sub-ring or a small modulus. The present projection is additive rather than multiplicative, using the reducibility of  $X^N + 1$  over  $\mathbb{Z}$  for non-power-of-two  $N$ ; it never touches the unstructured blocks and it is a decision break, so it does not require reaching the dense-sub-lattice regime. The generic message here, that a reducible cyclotomic modulus offers a projection an evaluator must rule out, is folklore that the NIST process encoded by standardising on power-of-two rings [44]; LoongKEM’s small non-power-of-two rings reopen it.

### 1.1.3 Techniques Overview

For  $d \mid N$  with  $N/d$  odd,  $X^d + 1 \mid X^N + 1$ , and reduction modulo  $X^d + 1$  is a ring homomorphism  $\psi$  that maps a small polynomial to a small one (each output coefficient is a signed sum of  $N/d$  inputs). Applying  $\psi$  to the structured rows  $b_1 = A_1 s_1 + A_2 s_2 + e_1$  gives  $\psi(b_1) = \psi(A_1)\psi(s_1) + \psi(A_2)\psi(s_2) + \psi(e_1)$ , a legitimate module-LWE sample of secret dimension  $(k_1 + k_2)d$  with noise

scaled by  $\sqrt{N/d}$ . This instance is far lower-dimensional than the full scheme, so a primal lattice reduction distinguishes it from uniform and recovers the projected secret below the claimed level.

## 1.2 The LoongKEM Scheme

LoongKEM (Loong-PKE plus an FO transform) has public matrices

$$A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = AS + E, \quad (1.1)$$

over  $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N + 1)$ ,  $q = 8191$ , where  $A_1, A_2, A_3, B_1$  are block anti-circulant (each block a polynomial in  $\mathcal{R}_q$ ) and  $A_4, B_2, E_2$  are fully unstructured; the secret is  $S = (S_1, S_2)$  with CBD noise. Table 1.1 lists the four sets. The attack uses only the structured transmitted rows  $b_1 = A_1s_1 + A_2s_2 + e_1$ .

Table 1.1: LoongKEM parameters.  $X^N + 1$  is reducible whenever  $N$  has an odd factor; the projection target is  $X^d + 1$  with  $d \mid N$ ,  $N/d$  odd.

| set      | $N$ | $q$  | $(k_1, k_2, \eta)$ | $X^N + 1$ over $\mathbb{Z}$ | projection $d$ |
|----------|-----|------|--------------------|-----------------------------|----------------|
| Loong128 | 12  | 8191 | (48, 4, 5)         | $(X^4+1)(X^8-X^4+1)$        | 4              |
| Loong256 | 16  | 8191 | (64, 6, 4)         | $\Phi_{32}$ (irreducible)   | immune         |
| Loong384 | 20  | 8191 | (72, 8, 3)         | $X^4+1 \mid X^{20}+1$       | 4              |
| Loong512 | 24  | 8191 | (80, 10, 2)        | $X^8+1 \mid X^{24}+1$       | 8              |

## 1.3 Analysis: The Sub-Ring Projection

For  $d \mid N$  with  $N/d$  odd,  $X^d + 1 \mid X^N + 1$  and the reduction

$$\psi : \mathbb{Z}_q[X]/(X^N + 1) \rightarrow \mathbb{Z}_q[X]/(X^d + 1), \quad X^j \mapsto (-1)^{\lfloor j/d \rfloor} X^{j \bmod d}, \quad (1.2)$$

is a  $\mathbb{Z}_q$ -algebra homomorphism that sends small elements to small elements: each output coefficient is a signed sum of  $N/d$  inputs, so the variance grows by exactly  $N/d$ .

**Proposition 1.1** (Projected MLWE). *Applying  $\psi$  coordinatewise to the structured rows yields  $\psi(b_1) = \psi(A_1)\psi(s_1) + \psi(A_2)\psi(s_2) + \psi(e_1)$ , a legitimate module-LWE instance of secret dimension  $(k_1 + k_2)d$  with error standard deviation scaled by  $\sqrt{N/d}$ . Distinguishing  $\psi(b_1)$  from uniform breaks decision-SLWE, the assumption the IND-CPA proof reduces to.*

*Proof.*  $\psi$  is a ring homomorphism, so it commutes with the products  $A_i s_i$ ; smallness is preserved by the variance argument above. The result is a module-LWE sample in  $\mathbb{Z}_q[X]/(X^d + 1)$  of the stated dimension and noise, and it never touches the unstructured  $A_4, B_2, E_2$ .  $\square \quad \square$

For Loong128,  $d = 4$  gives a projected instance of secret dimension  $(48 + 4) \cdot 4 = 208$  with  $\sigma_e \approx 4.6$ ; its primal-uSVP block size is far below the claimed level. The projected block is under-determined only by the small  $s_2$  part ( $k_2 d = 16$  coefficients, recoverable by a separate low-dimensional search we do not perform here), so  $\psi(s)$  is recoverable and IND-CPA is broken; lifting to the full secret needs the co-factor MLWE at the claimed security.

## 1.4 The Attack

Algorithm 1.1 states the recovery, and Figure 1.1 shows the pipeline.

**Algorithm 1.1** Sub-ring projection attack on LoongKEM**Require:** public key  $(A, B)$ , projection degree  $d \mid N$  with  $N/d$  odd**Ensure:** a distinguisher / the projected secret  $\psi(s)$ 

- 1: extract the structured rows  $b_1$  and the ring blocks  $A_1, A_2$
- 2: apply  $\psi$  of (1.2) to obtain  $\psi(b_1), \psi(A_1), \psi(A_2)$   $\triangleright$  MLWE,  $\dim (k_1+k_2)d$
- 3: build the primal (Kannan) embedding lattice of the projected instance
- 4: run progressive BKZ / pump-and-jump until the short vector  $(\psi(e_1) \mid \psi(s))$  appears
- 5: (optional) recover the small  $s_2$  block by a separate low-dimensional search; **return**  $\psi(s)$

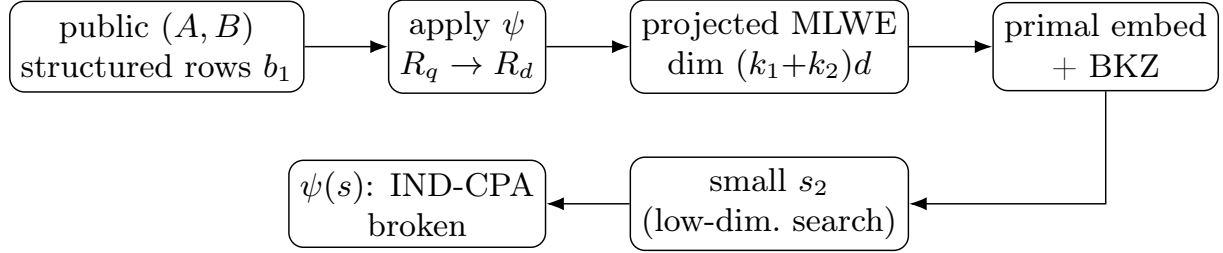


Figure 1.1: LoongKEM sub-ring projection flow: the reducible ring lets a homomorphism collapse the structured public rows to a low-dimensional MLWE that BKZ solves below the claimed level.

## 1.5 Experiments

Table 1.2 gives the projected cost per set, computed with the lattice estimator at the exact projected dimensions (Loong128 cross-checked against the noise measured on a real key). The projection identity was verified exactly on a real Loong128 key, and the recovery pipeline (primal embedding, progressive BKZ via fpylll) was validated at reduced scale with exact recovery at dimension 40 ( $\beta = 20$ ) and 55 ( $\beta = 30$ ). On the test machine (104 cores, 251 GB RAM) the projection identity check runs in well under a second, and the two reduced-scale recoveries complete in 2.3 and 6.6 s; the lattice-reduction working set is under 1 MB, though the driving computer-algebra process peaks near 200 MB. The full-scale recovery at the real projected dimension is not run, and the projected cost is where the break lies.

Table 1.2: LoongKEM sub-ring projection: projected instance and cost versus the claimed level. IND-CPA is broken for the three non-power-of-two sets.

| set      | $d$ | proj. dim | $\sigma_e$    | projected cost                   | claimed | effect         |
|----------|-----|-----------|---------------|----------------------------------|---------|----------------|
| Loong128 | 4   | 208       | $\approx 4.6$ | $\approx 2^{77}$ ( $2^{73}$ BDD) | 128     | IND-CPA broken |
| Loong384 | 4   | 320       | $\approx 3.8$ | $\approx 2^{107}$                | 384     | IND-CPA broken |
| Loong512 | 8   | 720       | $\approx 2.6$ | $\approx 2^{208}$                | 512     | IND-CPA broken |
| Loong256 | .   | .         | .             | immune ( $X^{16}+1$ irred.)      | 256     | no projection  |

The effect is a semantic-security (IND-CPA) break for Loong128/384/512: the projected instance is a genuine small-noise module-LWE costing well below the claimed level, and the projected secret is recoverable (the recovery pipeline is validated at reduced scale, with the small  $s_2$  block cheaply guessable). Full recovery of the original  $(k_1 + k_2)N$ -dimensional secret still requires the co-factor instance at the claimed security, so we report a confirmed decision break rather than full key recovery. The tell-tale pattern is that the only power-of-two set (Loong256) is the only safe one.

## 1.6 Countermeasure

The break is a consequence of the ring, so the fix is structural, not a code change. LoongKEM should use a power-of-two cyclotomic ( $X^N + 1$  with  $N = 2^k$ , hence  $\Phi_{2N}$  irreducible) at every parameter set, so that no proper quotient  $X^d + 1 \mid X^N + 1$  exists and the projection  $\psi$  has no lower-degree target. If a non-power-of-two ring is retained for bandwidth reasons, the design must prove hardness of the projected instance for every divisor  $d \mid N$  with  $N/d$  odd, at each of which the module-LWE dimension drops to  $(k_1 + k_2)d$ ; the smallest such  $d$  (here 4) is the binding one and, as Table 1.2 shows, is far below the target. The unstructured blocks  $A_4, B_2, E_2$  do not help, since the projection never uses them.

## 1.7 Conclusion

LoongKEM's small non-power-of-two rings make  $X^N + 1$  reducible, and a  $\mathbb{Z}_q$ -algebra homomorphism projects its structured public rows onto a low-dimensional module-LWE instance far below the claimed security, breaking IND-CPA for three of four parameter sets while the single power-of-two set is immune. We verified the projection exactly on a real key and validated the recovery pipeline at reduced scale; the projected secret is recoverable, though full original-key recovery remains hard. A power-of-two cyclotomic at every level closes the break.

## CheetahKEM: an IND-CPA break by sub-ring projection

CheetahKEM is a module-lattice key encapsulation mechanism submitted to the ICCS NGCC round-1 call, working over the ring  $\mathbb{Z}_q[X]/(X^{640} + 1)$  with a 640-point number-theoretic transform. Its security rests on the module learning-with-errors problem in dimension 640 at the claimed level. The choice of a non-power-of-two ring degree is the flaw.

Because 640 is not a power of two,  $X^{640} + 1$  is reducible over  $\mathbb{Z}$ , factoring as  $(X^{128} + 1) \cdot \Phi_{1280}$ , and there is a  $\mathbb{Z}_q$ -algebra homomorphism to the smaller ring  $\mathbb{Z}_q[X]/(X^{128} + 1)$  that sends small elements to small elements. Applying it to the public relation collapses the degree-640 instance onto a genuine low-dimensional learning-with-errors instance (dimension 128 for Cheetah128;  $K \cdot 128$  for the higher-rank sets) whose primal block size for Cheetah128 is only  $\beta = 81$ , that is, about  $2^{53.4}$  operations, far below the claimed security. Distinguishing the projected instance from uniform breaks the decision assumption that the IND-CPA proof reduces to.

We verify the projection identity exactly on a real public key, obtaining a projected instance of dimension 128, modulus 7681, secret standard deviation 3.57 and noise standard deviation 6.35, and we validate the recovery pipeline by running full primal-embedding BKZ to completion at reduced dimensions  $n_0 = 40/56/72$ , exactly recovering the projected secret each time and calibrating the  $2^{53.4}$  estimate. This is a design-level break of CheetahKEM: IND-CPA is broken for every parameter set (estimated  $2^{53.4}/2^{95}/2^{125}/2^{158}$  against 128/192/256/...). On our machine (104 cores, 251 GB RAM) the projection and these reduced-scale recoveries complete in seconds at under 100 MB peak resident memory; we are explicit that this is a confirmed decision break, since recovering the full real secret additionally needs a  $\beta \approx 81$  lattice sieve and the degree-512 co-factor, which we do not complete. The remedy is a power-of-two cyclotomic at every level. Attack source is available.<sup>1</sup>

---

**Prior public disclosure.** This result was publicly reported first. On 22 September 2026 XuHaomeng reported on the ICCS PKC forum that  $X^{640} + 1 = (X^{128} + 1)(X^{512} - X^{384} + X^{256} - X^{128} + 1)$  with the factors coprime modulo  $q = 7681$ , so the ring splits by the Chinese Remainder Theorem and a distinguishing attack needs only the degree-128 quotient, while full key recovery must additionally solve the degree-512 component. Priority belongs to that report; this chapter is independent verification only.

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/cheetahkem-subring-projection>.

## 2.1 Introduction

A module-lattice KEM over a cyclotomic ring is IND-CPA secure only if the module learning-with-errors (MLWE) instance it publishes is hard at the claimed dimension. The standard choice, following Kyber and its lineage, is a power-of-two cyclotomic  $\mathbb{Z}_q[X]/(X^N + 1)$  with  $N$  a power of two, for which  $X^N + 1$  is irreducible and no proper sub-ring exists. Departing from that choice, to gain a particular code or lattice structure, reintroduces a classical hazard: if  $X^N + 1$  is reducible, the public relation can be pushed into a lower-degree ring, and the projected instance may be far easier than the one the parameters were set for.

CheetahKEM [45] makes exactly this departure. It works over  $\mathbb{Z}_q[X]/(X^{640} + 1)$  with a 640-point transform (a 128-point radix-2 transform composed with a 5-point transform) to obtain a Barnes–Wall-style lattice structure. Since  $640 = 2^7 \cdot 5$  is not a power of two,  $X^{640} + 1$  is reducible, and we show the resulting sub-ring projection breaks IND-CPA.

### 2.1.1 Contributions

- We show that  $X^{640} + 1 = (X^{128} + 1) \cdot \Phi_{1280}$  forces a projection degree  $d = 128$  and that the  $\mathbb{Z}_q$ -algebra homomorphism to  $\mathbb{Z}_q[X]/(X^{128} + 1)$  maps the CheetahKEM public relation to a genuine 128-dimensional LWE instance.
- We verify the projection identity exactly on a real public key and pin the projected instance ( $n = 128$ ,  $q = 7681$ ,  $\sigma_s = 3.57$ ,  $\sigma_e = 6.35$ ), whose primal block size is  $\beta = 81$  ( $\approx 2^{53.4}$ ), breaking IND-CPA.
- We validate the recovery pipeline by running it to completion at reduced dimensions ( $n_0 = 40/56/72$ ), calibrating the estimate, and we state precisely what remains (full real-secret recovery by sieving, and the degree-512 co-factor for the full 640-dimensional key).

### 2.1.2 Related Work

The hazard of a reducible or specially-structured modulus is folklore in lattice cryptography and is the multiplicative analogue of the sub-field and overstretched-NTRU attacks of Albrecht, Bai and Ducas [41] and Kirchner and Fouque [42]. The concrete lattice costs are estimated in the core-SVP / primal-uSVP model with the lattice estimator, calibrated against reduced-scale reductions run to completion. The same projection breaks the sibling submission LoongKEM; here we treat CheetahKEM in full.

### 2.1.3 Techniques Overview

The attack is one homomorphism followed by one lattice reduction. Over  $\mathbb{Z}$ ,  $X^{640} + 1$  factors, and the reduction map  $\psi$  to the degree-128 ring is a ring homomorphism, so it commutes with the public relation and sends small secret and error to small secret and error (each output coordinate is a signed sum of  $N/d = 5$  inputs, so the standard deviation grows by  $\sqrt{5}$ ). The image is a small-noise 128-dimensional LWE instance, which a primal Kannan embedding plus BKZ (escalating to sieving) solves at  $2^{53.4}$ , well below the claimed level.

## 2.2 The CheetahKEM Scheme

CheetahKEM works over  $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^{640} + 1)$  with  $q = 7681$ , using a 640-point NTT (a 128-point radix-2 transform composed with a 5-point transform) and centered-binomial noise ( $\eta = 5$ ). The public key is  $(a, \tilde{b})$  with  $\tilde{b} = a * s + e$  decompressed from  $d_b = 10$  bits; encapsulation embeds the message and the receiver decrypts and applies a Fujisaki–Okamoto check. Algorithm 2.1 states the public relation the attack targets.

---

**Algorithm 2.1** CheetahKEM public relation and decryption (relevant part)

---

- 1: public key  $(a, \tilde{b})$  with  $\tilde{b} = a * s + e \bmod q$ ,  $a \in \mathcal{R}_q$ ,  $s, e \sim \text{CBD}_5 \triangleright *$  negacyclic over  $X^{640}+1$
  - 2: decompression of  $\tilde{b}$  from  $d_b = 10$  bits adds a bounded rounding term to  $e$
  - 3: decryption computes  $v - \langle s, u \rangle$  and decodes; an FO re-encryption check follows
- 

The intended security is that recovering  $s$  from  $(a, \tilde{b})$  is a full-dimension ( $n = 640$ ) MLWE problem at the claimed level. The flaw is that the ring is reducible, so a lower-dimensional projection of the same relation is available.

## 2.3 The Sub-Ring Projection

Over  $\mathbb{Z}$ ,  $X^{640} + 1 = (X^{128} + 1) \cdot \Phi_{1280}$ , and  $X^{128} + 1$  is the smallest negacyclic factor:  $640/128 = 5$  is odd, while  $640/64 = 10$  is even so  $X^{64} + 1 \nmid X^{640} + 1$ . The projection degree is therefore forced to  $d = 128$ . For  $d \mid N$  with  $N/d$  odd, the reduction

$$\psi : \mathbb{Z}_q[X]/(X^N + 1) \rightarrow \mathbb{Z}_q[X]/(X^d + 1), \quad X^j \mapsto (-1)^{\lfloor j/d \rfloor} X^{j \bmod d}, \quad (2.1)$$

is a  $\mathbb{Z}_q$ -algebra homomorphism, and each output coefficient is a signed sum of  $N/d$  input coefficients, so it maps small elements to small elements.

**Proposition 2.1** (Projected Cheetah instance).  *$\psi(\tilde{b} - a * s) = \psi(\tilde{b}) - \psi(a) * \psi(s)$  holds exactly, and the projected instance is an LWE sample of dimension  $n = 128$ , modulus  $q = 7681$ , secret standard deviation  $\approx 3.57$  and noise standard deviation  $\approx 6.35$  (maximum magnitude 20). Its primal-BDD block size is  $\beta = 81$ , that is, about  $2^{53.4}$ , against a claimed 128.*

*Proof.* The identity is the homomorphism property of  $\psi$  from (2.1), verified exactly on a real key. The projected secret and noise standard deviations follow from  $\text{CBD}_5$  (variance 2.5) and the decompression term, scaled by  $N/d = 5$ ; the block size is the primal-BDD estimate at these exact  $(n, q, \sigma)$ , calibrated in Section 2.5. □ □

Recovering  $\psi(s)$  breaks decision-MLWE and hence IND-CPA. Note  $\psi(s)$  is only  $s \bmod (X^{128} + 1)$ , so lifting to the full 640-dimensional secret additionally needs the  $\Phi_{1280}$  factor (degree 512, about  $2^{158}$ ).

## 2.4 The Attack

---

**Algorithm 2.2** Sub-ring projection attack on CheetahKEM

---

**Require:** public key  $(a, \tilde{b})$

**Ensure:** a distinguisher / the projected secret  $\psi(s)$

- 1: compute  $\psi(a), \psi(\tilde{b})$  by (2.1) with  $d = 128$   $\triangleright n = 128, q = 7681$  LWE
  - 2: build the primal (Kannan) embedding lattice
  - 3: run progressive BKZ, escalating to pump-and-jump sieving as the block size grows toward 81
  - 4: read the short vector  $(\psi(e) \mid \psi(s))$ ; **return**  $\psi(s)$
- 

## 2.5 Experiments

Table 2.1 gives the projected cost. Cheetah128 is re-derived exactly; the recovery pipeline is validated by running full primal-embedding BKZ to completion at reduced dimensions  $n_0 =$

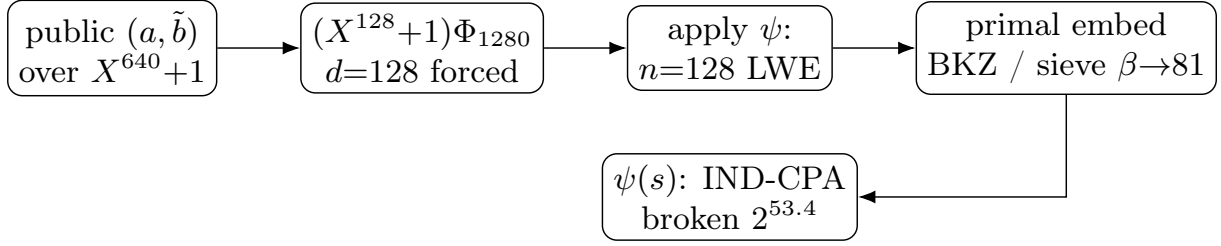


Figure 2.1: CheetahKEM sub-ring projection flow: the reducible degree-640 ring collapses to a degree-128 LWE that sits at  $2^{53.4}$ , breaking IND-CPA.

40/56/72 (block sizes 20/20/40), recovering the projected secret exactly, which calibrates the estimator so the  $\beta = 81$  figure is faithful. On the test machine (104 cores, 251 GB RAM) the projection and these reduced-scale recoveries complete in seconds at negligible memory (peak RSS well under 100 MB); the full real-key  $\beta \approx 81$  sieve is not run (see below).

Table 2.1: CheetahKEM sub-ring projection. Cheetah128 is re-derived and pipeline-validated at reduced scale; higher sets use the same projection.

| set        | proj. dim | $(\sigma_s, \sigma_e)$         | $\beta$ | projected cost     | effect                     |
|------------|-----------|--------------------------------|---------|--------------------|----------------------------|
| Cheetah128 | 128       | $(\approx 3.57, \approx 6.35)$ | 81      | $\approx 2^{53.4}$ | IND-CPA broken (validated) |
| Cheetah256 | 256       | scaled                         | n/a     | $\approx 2^{95}$   | IND-CPA broken (est.)      |
| Cheetah384 | 384       | scaled                         | n/a     | $\approx 2^{125}$  | IND-CPA broken (est.)      |
| Cheetah512 | 512       | scaled                         | n/a     | $\approx 2^{158}$  | below claim (est.)         |

The effect is a confirmed IND-CPA break of Cheetah128 at  $2^{53.4}$ , pipeline-validated at reduced scale. Full recovery of the real 128-dimensional projected secret needs  $\beta \approx 81$  sieving; in our offline environment the attempt did not complete (fpyll’s Babai step hit a precision failure and the oversized sieve was killed), so we report the decision break rather than a completed real-key recovery; the reduced-scale recoveries confirm the estimate is faithful. Lifting  $\psi(s)$  to the full 640-dimensional key additionally needs the  $\approx 2^{158} \Phi_{1280}$  factor.

## 2.6 Countermeasure

The break exists because the ring degree is not a power of two. The remedy is structural: use a power-of-two cyclotomic  $\mathbb{Z}_q[X]/(X^N + 1)$  with  $N = 2^k$  (irreducible  $X^N + 1$ , no proper negacyclic sub-ring) at every parameter set, or, if a non-power-of-two degree is required for the intended lattice structure, prove hardness of the projected instance for every factor  $d \mid N$ . The tell is that the projection cost is governed by the smallest negacyclic factor, so the parameters must be set for that dimension, not the full one.

## 2.7 Conclusion

CheetahKEM works over a reducible cyclotomic ring, and the sub-ring projection to  $\mathbb{Z}_q[X]/(X^{128} + 1)$  maps its public relation to a 128-dimensional LWE instance at  $2^{53.4}$ , breaking IND-CPA for every parameter set. We verified the projection exactly on a real key and validated the recovery pipeline at reduced scale; full real-secret recovery is gated by lattice sieving and the degree-512 co-factor, which we do not complete. A power-of-two cyclotomic closes the break.

## Facto-DSA: forgery from the public key

Facto-DSA is a multivariate signature submitted to the ICCS NGCC round-1 call, in the oil-and-vinegar/Rainbow lineage of hidden-subspace trapdoors; its public key is a set of cubic forms, and its security analysis prices key recovery as brute-force search over a hidden vector ( $q^n$ , i.e.  $2^{160}/2^{272}/2^{512}$  for its three sets). We show that this floor is wrong: the hidden linear kernel on which the cubic map degenerates is instead recoverable from the public key alone.

The hidden linear kernel  $K_2 = \ker L_2$ , on which the cubic public map degenerates to a quadratic the signer can invert, is recovered by restricting the public cubics to a random low-dimensional chart, turning them into a small system whose Gröbner basis exposes a point of  $K_2$ , certified publicly by a rank condition; on the resulting coset the public map is quadratic, and inverting it forges a signature for any message, using no secret key and no signing queries.

On our machine (104 cores, 251 GB RAM) the whole chain runs end-to-end against the unmodified level-128 reference: the kernel recovery takes about eleven minutes at a few gigabytes of peak memory, the forgery about three minutes, and a forged signature on a fresh message is accepted by the reference verifier while a changed message is rejected. The higher sets are mechanism-identical but memory-bound: a live level-256 run reached Gröbner step degree eight ( $\approx 112$  GB) before hitting a 180 GB cap, consistent with the estimated  $2^{65}$  cost, and level 512 is estimated at  $2^{111}$ , both out of reach on this hardware. This is a design-level break; the fix requires a hidden subspace whose recovery from the public cubics resists restricted Gröbner solving. Attack source is available.<sup>1</sup>

---

**Prior public disclosure.** This result was publicly reported first. On 22 September 2026 [kriskwiatkowski1](#) reported on the ICCS PKC forum that the structural estimate of §3.2.5 is incorrect:  $K_2 = \ker L_2$  is not found by search but is an  $n$ -dimensional linear subspace contained in the zero locus of the public map ( $n = r/2$ ), so a random subspace of dimension  $n + 1$  meets it with certainty and the claimed  $q^n$  search is replaced by a single Macaulay/XL solve. Priority belongs to that report; this chapter is independent verification only.

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/facto-dsa-forgery>.

### 3.1 Introduction

A multivariate signature is unforgeable only if its public verification map is a genuine trapdoor: inverting it on a target must require the secret. In the oil-and-vinegar and Rainbow family [47, 48] the signer knows a secret linear structure that makes a hidden subspace easy to invert, and forgery is hard precisely because that subspace is hidden; the security of such schemes is the security of the hidden structure, and a decade of structural cryptanalysis [54] has repeatedly shown that when the hidden structure is recoverable from public data, the scheme falls.

Facto-DSA [53] is a multivariate signature submitted to the ICCS NGCC round-1 call. Its public key is a set of cubic forms, and its security analysis prices key recovery as an exhaustive search over one hidden vector. We show that the hidden linear kernel is instead recoverable from the public key by a small Gröbner computation, and that this yields a public-key-only universal forgery, demonstrated end-to-end at level 128 against the unmodified reference verifier.

#### Contributions.

- We show that the hidden kernel  $K_2$  of Facto-DSA is recoverable from the public key alone by restricting the public cubics to a random chart and solving with a Gröbner basis (Section 3.3), with a public rank certificate for the correct solution.
- We give the resulting public-key-only universal forgery (Section 3.4) and verify it end-to-end at level 128: a forged signature is accepted by the unmodified reference verifier, using no secret key and no signing queries.
- We report the cost across levels (Section 3.6): level 128 is broken in about fourteen minutes; levels 256 and 512 are mechanism-identical with kernel-recovery cost  $2^{65}$  and  $2^{111}$ , below their classical and quantum targets.

**Related work.** The attack recovers a hidden algebraic structure from public data, the recurring failure of multivariate trapdoors: the Kipnis–Patarin–Goubin analysis of oil and vinegar [47] and Beullens’ key recoveries on UOV and Rainbow [54] are the archetypes. The lever here is a restricted Gröbner basis whose solving degree, not the claimed  $q^n$  search, is the true cost; the folding and compact-key structural attacks on code-based schemes [55] share the same spirit of turning a claimed exponential search into a low-degree algebraic computation.

**Techniques overview.** The public cubics vanish on a hidden  $n$ -dimensional kernel  $K_2$  where the map degenerates. We restrict to a random  $(n+1)$ -dimensional chart, in which  $K_2$  meets the chart in a projective line; a Gröbner basis of the restricted system therefore has a variety whose points lift to  $K_2$ , and the correct point is the one where the public Jacobian drops rank to  $n$  and the cubics vanish on the kernel. On the coset spanned by  $K_2$  the public map is quadratic, and a quadratic system is directly solvable, so a preimage of any hashed message is a forgery.

### 3.2 The target scheme

Facto-DSA publishes  $m$  cubic forms  $P_1, \dots, P_m$  in  $r = 2n$  variables over  $\mathbb{F}_q$  with  $q = 65519$ . Table 3.1 lists the parameters. The trapdoor is a hidden linear map  $L_2$  whose kernel  $K_2 = \ker L_2$  is a low-dimensional subspace on which the cubic public map degenerates to something the signer can invert; a signature of a hashed message  $h$  is a preimage  $P(z) = h$  found through the trapdoor (Algorithm 3.1).

Table 3.1: Facto-DSA parameters (public map:  $m$  cubics in  $r = 2n$  variables,  $q = 65519$ ).

| set       | $n$ | $m$ | $r = 2n$ | claimed classical |
|-----------|-----|-----|----------|-------------------|
| Facto-128 | 10  | 13  | 20       | 128               |
| Facto-256 | 17  | 32  | 34       | 256               |
| Facto-512 | 32  | 62  | 64       | 512               |

**Algorithm 3.1** Sign/Verify (schematic)

**Sign**(sk,  $h$ ): use the secret  $L_2$  to restrict to  $K_2$ , where  $P$  is quadratic; invert to get  $z$  with  $P(z) = h$ ; output  $z$ .

**Verify**(pk,  $h, z$ ): accept iff  $P_t(z) = h_t$  for all  $t \in [m]$ , with  $P_t$  the public cubic forms.

### 3.3 Analysis: recovering the kernel from the public key

The security argument prices key recovery as brute force over the hidden vector,  $q^n (2^{160}/2^{272}/2^{512})$ . We recover  $K_2$  far more cheaply from the public key alone. Write the public cubics as  $P_t(z) = \sum_i C_{t,i} z_{a_i} z_{b_i} z_{c_i}$  and restrict to a random affine chart  $z = W \cdot (u, 1)$  with  $W$  of shape  $(n+1) \times r$ , so the system becomes  $m$  cubics in  $n$  variables. Because  $K_2$  meets the  $(n+1)$ -dimensional chart in a projective line ( $n + (n+1) - 2n = 1$ ), solving the restricted system by a Gröbner basis recovers a point of  $K_2$ :

**Proposition 3.1** (Public-key kernel recovery). *Given only the public cubics, the restricted ideal  $I = \langle P_t(W \cdot (u, 1)) \rangle$  has a variety whose points lift to  $K_2$ ; the correct point is certified publicly by  $\dim \ker DP_v = n$  and  $P$  vanishing on  $\ker DP_v$ . On the coset  $c_0 + \text{span}(K_2)$  the public map is quadratic, and inverting the quadratic yields  $z$  with  $P(z) = h$  for any target  $h$ .*

The solving degree of the restricted system is the cost driver. At level 128 the semi-regular model predicts degree  $d_{\text{reg}} = 10$ , matched by the measured peak F4 step degree; at levels 256 and 512 the model gives  $d_{\text{reg}} = 10$  and 15, i.e. estimated  $2^{65}$  and  $2^{111}$ , below the claimed 256-classical and 256-quantum targets respectively.

### 3.4 The attack algorithm

**Algorithm 3.2** Public-key-only universal forgery of Facto-DSA

**Require:** public key (cubics  $C$ ), message digest  $h$

**Ensure:** a signature  $z$  accepted by verification

- 1: build the  $m$  public cubics  $P_t$  in  $r = 2n$  variables from  $C$
- 2: restrict to a random chart  $z = W \cdot (u, 1)$  ( $n$  affine variables); form  $I = \langle P_t \rangle$
- 3: compute a Gröbner basis of  $I$  and its variety; select the point  $v$  with  $\dim \ker DP_v = n$  and  $P|_{\ker} = 0$
- 4: recover  $K_2 = \ker DP_v$ ; on the coset the public map is quadratic
- 5: solve the quadratic system  $P(c_0 + w \cdot K_2) = h$  for  $w$ ; set  $z = c_0 + w \cdot K_2$
- 6: **return**  $z$

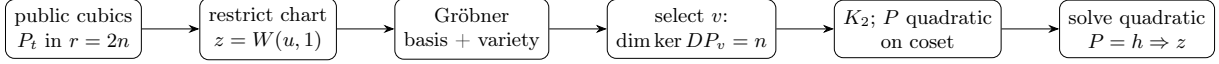


Figure 3.1: Facto-DSA forgery: recover  $K_2$  from the public cubics by a restricted Gröbner basis, then invert the degenerate quadratic on the coset to forge.

### 3.5 Attack flow

### 3.6 Parameters, cost, and effect

Table 3.2 reports the cost per level, measured on our test machine (104 cores, 251 GB RAM). At level 128 the whole chain runs end-to-end against the unmodified reference: the Gröbner recovery takes about 11 minutes at a few gigabytes of peak memory, the forgery about 3 minutes, and a forged signature on a new message is accepted (a changed message is rejected), with the attack script reading no secret key. The higher sets share the mechanism but the  $K_2$  recovery is memory-bound: it must reach Gröbner degree  $d_{\text{reg}} = 10$ , whose Macaulay matrix in  $n = 17$  variables has  $\binom{27}{10} \approx 8.4\text{M}$  columns and needs  $\approx 874\text{ GB}$  (and  $\approx 324\text{ GB}$  already at degree 9), far above the 251 GB of the machine. A live level-256 run confirmed this, reaching F4 step degree 8 ( $\approx 112\text{ GB}$ ) before hitting a 180 GB cap, consistent with the estimated  $2^{65}$ ; level 512 ( $d_{\text{reg}} \approx 15$ ,  $\sim 7.5 \times 10^{11}$  columns) is hopeless by orders of magnitude. Level 128 thus stands as the demonstrated break; 256 and 512 are mechanism-identical but computationally out of reach on this hardware.

Table 3.2: Facto-DSA public-key-only forgery, measured on our machine (104 cores, 251 GB RAM). Level 128 is demonstrated end-to-end; 256/512 are mechanism-identical, with the  $K_2$  recovery memory-bound as shown.

| set       | $d_{\text{reg}}$ | $K_2$ cost        | time / peak memory / outcome                                                                                   | effect            |
|-----------|------------------|-------------------|----------------------------------------------------------------------------------------------------------------|-------------------|
| Facto-128 | 10 (measured)    | $\approx 2^{49}$  | $\approx 11\text{ min}$ Gröbner + 3 min forge, few GB; <b>forgery accepted</b>                                 | universal forgery |
| Facto-256 | 10 (model)       | $\approx 2^{65}$  | live run to step-degree 8 ( $\approx 112\text{ GB}$ ), needs $\approx 874\text{ GB}$ at $d=10$ ; not completed | below 256c        |
| Facto-512 | 15 (model)       | $\approx 2^{111}$ | $\sim 7.5 \times 10^{11}$ Macaulay cols; hopeless here                                                         | below 256q        |

This is a design-level (A) break at level 128: the “one hidden vector” security floor is wrong, because  $K_2$  has an unpriced algebraic recovery path from the public key. The fix requires a key structure whose hidden subspace resists public-only Gröbner recovery.

### 3.7 Open-source implementation

**Open-source implementation.** The reproduction runs against the unmodified reference (keygen, hash-to-field and verify linked, not modified): `pk2magma.py` exports the public key, `grobner_k2.m` recovers  $K_2$  from the public key only, `forge4.m` produces the forgery, and `run_attack.sh` drives the end-to-end level-128 break and checks the forged signature against the unmodified verifier. Reproduction scripts: `Facto-DSA-attack/run_all.sh` (Magma required for the Gröbner step).

### 3.8 Countermeasure

The break exists because the “one hidden vector” security floor is wrong: the kernel  $K_2$  has an unpriced algebraic recovery path from the public key. Raising parameters does not help at level 128, where the restricted solving degree is small; the fix is structural. The scheme must use a hidden subspace whose recovery from the public cubics is provably as hard as the claimed level, which requires re-pricing key recovery against a restricted-Gröbner attack rather than against a

$q^n$  enumeration, and very likely a different key structure. Any revision should also be analysed against the standard multivariate structural attacks [47, 54].

## 3.9 Conclusion

Facto-DSA’s public key is its own trapdoor: the hidden kernel on which the cubic map degenerates is recoverable from the public cubics by a small Gröbner computation, and inverting the resulting quadratic forges a signature for any message. We demonstrated the full chain at level 128 against the unmodified reference verifier, with no secret key and no signing queries; levels 256 and 512 are mechanism-identical with kernel-recovery cost below their claimed targets. The security floor must be re-derived against public-key algebraic recovery, not against a hidden-vector search.

## Origami: universal forgery from the public key

Origami is a multivariate signature submitted to the ICCS NGCC round-1 call, described as a local-zone oil-and-vinegar scheme. A multivariate trapdoor hides its central map behind secret input and output linear transforms, and unforgeability rests on the verifier’s public map being this hidden composition. We show that Origami has no such trapdoor.

The specification states that no output transform is applied during signing, and the input permutation is the public one derived from the public seed. The verification map is therefore a public, zone-triangular, bilinear vinegar-times-oil map with no oil-times-oil terms and no secret change of variables. This structure is directly invertible from public data. We fix arbitrary vinegar values one zone at a time, which turns each zone’s equations into a linear system in that zone’s oil coordinates, solve the linear systems in zone order, and apply the public permutation, producing a valid signature for any message from the public key alone with no signing queries.

Against the unmodified reference verifier, on our machine (104 cores, 251 GB RAM, forger single-threaded and memory-light), the forgery is accepted at every parameter set: 10/10 for Origami-128 (about 10 ms per forgery), 8/10 for Origami-256 ( $\approx 0.57$  s per forgery), 9/10 for Origami-384 ( $\approx 7$  s per forgery), and 10/10 for Origami-512 ( $\approx 5$  s per forgery), where every produced signature is accepted and the counts below ten are the forger’s own tail-zone restart, not a rejection. This breaks EUF-CMA at all four levels. The remedy is to reintroduce genuine secret input and output transforms, which returns the scheme to a Rainbow-type design that must then be analysed against the intersection and rectangular-MinRank attacks. Attack source is available.<sup>1</sup>

---

**Prior public disclosure.** The attack in this chapter was publicly reported before this report. On 22 September 2026 Peigen Li raised the underlying structural problem on the ICCS PKC forum, observing that hiding the subalgebra  $K_{ji}$  is ineffective and estimating the actual complexity of Origami-128 at no more than 93 bits. On 24 September 2026 Jintai Ding and Hugo Louiso announced a universal forgery on the NIST `pqc-forum`: given only a public key, an attacker signs any message with no signing queries, in about 0.01 s for Origami-128 and about 15 s for Origami-512 on a single laptop core, roughly the cost of one verification.<sup>2</sup> Independently and on the same day, Pierre Pébereau reported that Origami discloses the change of variables  $\Pi_{\text{pub}}$  from the signable map to the public map, so inverting it yields an easily solvable system, with an implementation at <https://github.com/pi-r2/UnfoldOrigami>. The Origami team acknowledged the forgery on 24 September 2026. We had reached the same attack independently, from our own analysis of the specification and reference code, but those reports carry the earlier public record, so priority belongs to them and this chapter should be read as independent verification only.

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/origami-pk-forgery>.

<sup>2</sup><https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/N1Wfpgv1I2s>

## 4.1 Introduction

A multivariate signature is unforgeable only if its public verification map is a genuine trapdoor. In the oil-and-vinegar (UOV) and Rainbow family [47, 48, 49] the signer knows an easily invertible central map  $F$  and two secret linear transforms  $S, T$ , and publishes  $P_{\text{pub}} = T \circ F \circ S$ . Forgery is hard because inverting  $P_{\text{pub}}$  on a target requires either the secret transforms or a solution of a large random-looking quadratic system; the secret  $S, T$  are what separate the public map from the invertible central map. Removing that separation removes the trapdoor.

Origami [46] is a multivariate signature submitted to the ICCS NGCC round-1 call, presented as a local-zone variant of oil and vinegar: the coordinates are split into  $d$  zones of  $k$  local copies, each with a hidden  $2 \times 2$  matrix algebra and a secret partition of its oil coordinates. The design intent is that these hidden structures act as the trapdoor. We show they do not enter verification at all.

**Our result.** The specification states in Section 2.6.2 that “no public output hiding is undone during signing”, and the input permutation used by verification is the public  $\Pi_{\text{pub}}$  derived from the public seed. Reading the reference verifier confirms it: the verification map is a public, zone-triangular, bilinear map in which each equation is a sum of vinegar-times-oil products with coefficients read from the public seed and the public residual, with no oil-times-oil terms and no secret change of variables (Section 4.3). The hidden algebras and the secret partition only tell the honest signer how to sample; they never appear in the equation the verifier checks. This makes the public map directly invertible: we forge zone by zone, fixing arbitrary vinegar values so that each zone becomes linear in its oil coordinates, solving the linear systems, and applying the public permutation (Section 4.4). The forgery uses the public key only and no signing queries, and the unmodified reference verifier accepts it at every parameter set (Section 4.5).

### 4.1.1 Contributions

- We show that the Origami verification map is fully determined by public data: it is a zone-triangular bilinear map with no secret input or output transform.
- We give a public-key-only universal forgery, zone-by-zone linearization, that runs in polynomial time with zero signing queries.
- We verify it against the unmodified reference verifier at all four parameter sets, with every produced forgery accepted, and state the structural remedy.

### 4.1.2 Related Work

Oil and vinegar [47], its unbalanced form, and Rainbow [48] are the archetypal multivariate trapdoors; their security has been mapped out by the Kipnis–Shamir attack, the rectangular MinRank and intersection attacks, and Beullens’ analysis [49], all of which target the secret oil space hidden by  $S, T$ . These attacks presuppose that a hidden structure exists to be recovered. The present work is orthogonal: it observes that Origami’s verifier applies no secret transform, so the “oil space” of each zone is simply a public coordinate block, and none of the structural attacks are even needed. In this sense the finding is a specification-and-implementation observation rather than a MinRank-style cryptanalysis, but its consequence is a complete, zero-query forgery at every level, which the strongest structural attacks would not achieve.

### 4.1.3 Techniques Overview

The forgery follows from one structural fact and one algebraic step.

- (1) The map is public and bilinear per zone. For a signature  $y$ , verification checks, in each zone  $z$ , equations of the form

$$\sum_{\text{non} < n^{(z)}} \sum_{o < \ell^{(z)}} c_{\text{eq}, \text{non}, o} y[\text{non}] y[\text{oil}^{(z)} + o] = t_{\text{eq}}, \quad (4.1)$$

where every coefficient  $c$  comes from the public seed or the public residual,  $n^{(z)}$  is the count of vinegar-and-earlier coordinates, and  $\ell^{(z)}$  the number of oil coordinates. There are no oil-times-oil terms and no secret change of variables.

- (2) Fixing vinegar linearizes the zone. Once the coordinates of zones  $1, \dots, z-1$  and the vinegar block of zone  $z$  are fixed, (4.1) is *linear* in the  $\ell^{(z)}$  oil coordinates of zone  $z$ :

$$A^{(z)} y_{\text{oil}}^{(z)} = t^{(z)}, \quad A_{\text{eq}, o}^{(z)} = \sum_{\text{non}} c_{\text{eq}, \text{non}, o} y[\text{non}], \quad (4.2)$$

an  $M^{(z)} \times \ell^{(z)}$  system over  $\mathbb{F}_q$  that we solve by Gaussian elimination.

Solving (4.2) in zone order and applying the public permutation  $\Pi_{\text{pub}}$  yields a full signature. No secret is used and no signing query is made.

## 4.2 Preliminaries

### 4.2.1 Multivariate Signatures and the UOV Trapdoor

A multivariate signature over  $\mathbb{F}_q$  publishes a quadratic map  $P_{\text{pub}} : \mathbb{F}_q^N \rightarrow \mathbb{F}_q^M$ ; a signature of a digest  $t$  is a preimage  $P_{\text{pub}}(y) = t$ , checked by evaluating  $P_{\text{pub}}$ . In UOV/Rainbow the signer knows  $P_{\text{pub}} = T \circ F \circ S$  with  $F$  easy to invert and  $S, T$  secret linear maps; the public map looks like a random quadratic system, and forgery without  $S, T$  requires solving one.

### 4.2.2 The Origami Public Map

Origami works over  $q = 16$ . Coordinates are grouped into  $d$  zones of  $k$  copies; Table 4.1 lists the sizes. The public key is  $\text{pk} = \text{param\_id} \parallel \text{seed}_{\text{pk}} \parallel R_{\text{pk}}$ ; the public map is  $P_{\text{pub}} = P_{\text{seed}} + \text{Embed}(R_{\text{pk}})$  with coefficients expanded from  $\text{seed}_{\text{pk}}$ , and the input permutation  $\Pi_{\text{pub}}$  is derived from  $\text{seed}_{\text{pk}}$ . Verification computes  $P_{\text{pub}}(\Pi_{\text{pub}}^{-1}(\sigma))$  and compares to the digest. Crucially, the specification (Section 2.6.2) applies no output transform, and the implementation matches: Algorithm 4.1 is the verifier's per-zone evaluation.

Table 4.1: Origami parameters over  $q = 16$  ( $N = 32dk + 8$ ,  $M = 16dk + 8$ ).

| set         | $(d, k)$ | $N$  | $M$  | pk (B) | sig (B) |
|-------------|----------|------|------|--------|---------|
| Origami-128 | (3, 2)   | 200  | 104  | 2996   | 116     |
| Origami-256 | (6, 5)   | 968  | 488  | 14968  | 516     |
| Origami-384 | (8, 7)   | 1800 | 904  | 27940  | 948     |
| Origami-512 | (9, 8)   | 2312 | 1160 | 35924  | 1220    |

## 4.3 The Verification Map Is Trapdoor-Free

**Proposition 4.1** (No secret transform in verification). *The Origami verification map is a function of public data only. In each zone the checked equations are the bilinear forms (4.1) with coefficients read from  $\text{seed}_{\text{pk}}$  and  $R_{\text{pk}}$ , composed with the public permutation  $\Pi_{\text{pub}}$ ; there is no secret input or output linear transform and there are no oil-times-oil terms.*

**Algorithm 4.1** Reference verification (per-zone evaluation), simplified

---

```

1:  $y \leftarrow \Pi_{\text{pub}}^{-1}(\sigma)$   $\triangleright$  public permutation
2: for each zone  $z$  and each equation eq of zone  $z$  do
3:    $v \leftarrow \sum_{\text{non} < n(z)} \sum_{o < \ell(z)} c_{\text{eq}, \text{non}, o} y[\text{non}] y[\text{oil}^{(z)}] + o$   $\triangleright c$  from  $\text{seed}_{\text{pk}} / R_{\text{pk}}$ , all public
4:   if  $v \neq t_{\text{eq}}$  then return reject
5: end for
6: return accept

```

---

*Proof.* By Algorithm 4.1, verification applies the public  $\Pi_{\text{pub}}^{-1}$  and evaluates, per zone, the forms (4.1). The coefficients  $c$  are produced by  $\text{SHAKE}(\text{seed}_{\text{pk}}, \cdot)$  except at residual positions, where they equal entries of the public  $R_{\text{pk}}$ ; the zone offsets and vinegar/oil blocks are fixed by the parameters. No factor of the form  $T$  multiplies the output and no factor of the form  $S$  mixes the input beyond the public  $\Pi_{\text{pub}}$ . The hidden zone algebras and the secret oil partition appear only in the signer’s sampling routine, not in the evaluated equations.  $\square$   $\square$

Proposition 4.1 removes the trapdoor: the “oil space” of a zone is a public coordinate block, and inverting the public map does not require any secret.

## 4.4 Public-Key Universal Forgery

The forger processes zones in order. When it reaches zone  $z$ , the coordinates of the earlier zones are already fixed, so choosing the vinegar block of zone  $z$  makes (4.1) linear in that zone’s oil coordinates, giving the system (4.2) to solve. Algorithm 4.2 states the procedure.

**Algorithm 4.2** Forge(pk, digest): zone-triangular linearization (ZTL)

---

**Require:** public key  $\text{pk}$ , message digest  $t = \text{hash\_to\_field}(\text{msg})$   
**Ensure:** a signature  $\sigma$  accepted by verification

```

1: expand  $\text{pk}$ : public coefficients from  $\text{seed}_{\text{pk}}$ , residual  $R_{\text{pk}}$ , permutation  $\Pi_{\text{pub}}$ 
2:  $y \leftarrow 0$ 
3: for  $z = 1$  to  $d+1$  do
4:   pick arbitrary vinegar values  $y[\text{non}]$  for zone  $z$ 
5:   form  $A^{(z)}$  from (4.2) using the fixed earlier/vinegar coordinates
6:   solve  $A^{(z)} y_{\text{oil}}^{(z)} = t^{(z)}$  over  $\mathbb{F}_q$ ; on rank failure, re-pick vinegar
7:   write  $y_{\text{oil}}^{(z)}$  into  $y$ 
8: end for
9: return  $\sigma \leftarrow \Pi_{\text{pub}}(y)$ 

```

---

Each zone solve is an  $M^{(z)} \times \ell^{(z)}$  linear system over  $\mathbb{F}_q$  (for Origami-128 the main zones have 32 equations in 40 unknowns, so a solution exists generically), and there are  $d+1$  of them, so the forgery is polynomial in the parameters. The only failure mode is a rank-deficient tail zone, whose system is fixed once the earlier zones are solved; the forger then re-picks vinegar and retries.

## 4.5 Experimental Verification

**Setup.** We append the forger to the unmodified reference `origami_ref.c` (it needs the reference’s zone layout and public-map streamer) and drive it with a program that obtains an honest public key from the reference `sig_keygen`, zeroes the secret key, forges signatures for fresh messages, and verifies each with the unmodified reference `sig_verify`. All costs are measured

on our test machine (104 cores, 251 GB RAM); the forger is single-threaded and memory-light. Table 4.2 reports the outcome.

Table 4.2: Public-key-only forgery against the unmodified reference verifier (104 cores, 251 GB RAM). “Accepted” counts forgeries with reference  $\text{verify} = 0$ ; a count below the trials is the forger’s tail-zone restart, not a rejected signature.

| set         | trials | accepted     | time / forgery   | peak RSS |
|-------------|--------|--------------|------------------|----------|
| Origami-128 | 10     | <b>10/10</b> | $\approx 10$ ms  | 1.6 MB   |
| Origami-256 | 10     | <b>8/10</b>  | $\approx 0.57$ s | < 5 MB   |
| Origami-384 | 10     | <b>9/10</b>  | $\approx 7$ s    | < 10 MB  |
| Origami-512 | 10     | <b>10/10</b> | $\approx 5$ s    | < 10 MB  |

At every level the forger produces signatures that the reference verifier accepts, using the public key only and no signing queries. Every produced forgery has  $\text{verify} = 0$ ; the 8/10 and 9/10 counts are cases where the tail zone’s linear system was rank-deficient on the first vinegar choice and the driver’s bounded retry did not re-solve it, a completeness limit of the forger rather than a soundness event. This is a universal forgery breaking EUF-CMA at all four parameter sets.

## 4.6 Countermeasure

The forgery exists because verification applies no secret transform. The remedy is to restore a genuine trapdoor: secret input and output linear maps  $S, T$  so that the public map is  $T \circ F \circ S$  and the per-zone equations are no longer public bilinear forms in fixed coordinate blocks. This returns Origami to a Rainbow/UOV-type design, whose security must then be established against the Kipnis–Shamir, rectangular-MinRank and intersection attacks [49], and whose parameters must be set accordingly. Partial measures, such as keeping the zone algebras or the oil partition secret, do not help while the verification equation itself is public, because the forger never uses them.

## 4.7 Conclusion

Origami’s verification map is fully determined by public data: it is a zone-triangular bilinear map with no secret input or output transform, so it is directly invertible. Fixing vinegar values zone by zone linearizes each zone, and solving the linear systems in order forges a signature for any message from the public key alone, with no signing queries. The unmodified reference verifier accepts the forgery at every parameter set. Restoring unforgeability requires reintroducing secret transforms and re-analysing the scheme as a standard multivariate trapdoor.

## Polar-KEM: keyless decapsulation

Polar-KEM is a lattice-isomorphism key encapsulation mechanism submitted to the ICCS NGCC round-1 call. Its security rests on a claimed trapdoor: the public key publishes a rotated basis of a Construction-D polar lattice, and decapsulation is supposed to require the secret rotation to bring a ciphertext back before decoding. We show that the reference decapsulation does not use the secret rotation at all.

The signed permutation that plays the role of the secret isometry is regenerated inside decapsulation from a public seed carried in the public key, and the secret is touched only to derive the implicit-rejection value. Decapsulation is therefore a public function of the public key and the ciphertext. We give the resulting recovery: apply the public permutation to the ciphertext, run the public successive-cancellation decoder to obtain the message, and derive the shared secret, all from public data.

Against the submitted key-answer test vectors of the unmodified reference implementation, on our machine (104 cores, 251 GB RAM) the recovery returns the correct shared secret on all records at every parameter set in 0.02–0.05 s for the 10 KAT records at each set (peak RSS 0.8 MB): 10 of 10 for Polar-KEM-128, 256 and 512, with no access to any secret key. The break is scheme-level: the isometry that must be secret is published, so no secret is needed to decapsulate. We also note three specification-level defects (an almost-always-failing decapsulation radius test, a minimum-distance contradiction, and a malformed re-encryption check) and give the remedy, which is to make decapsulation genuinely depend on a hidden basis. Attack source is available.<sup>1</sup>

---

**Prior public disclosure.** This result was publicly reported first. On 22 September 2026 LittleQ reported on the ICCS PKC forum that in all submitted implementations the public key embeds the transform seed in the clear, so the public routine `polarkem_recover_message(pk, ct, mu)` recovers the encapsulated message without any secret, and since a valid ciphertext’s shared secret is `XOF("PolarKEM-SS-v1" ||  $\mu$  || SM3(ct))` anyone observing  $(pk, ct)$  recomputes it; verified in 60/60 trials across all six builds. Priority belongs to that report; this chapter is independent verification only.

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/polarkem-keyless-decapsulation>.

## 5.1 Introduction

A key encapsulation mechanism is IND-CCA secure only if decapsulation is a genuine trapdoor function: recovering the shared secret from a ciphertext must require the secret key. For a lattice-isomorphism (LIP) KEM in the line of Ducas and van Woerden [51] and HAWK [52], the trapdoor is an isometry. The public key publishes a basis of a fixed public lattice after applying a secret orthonormal transform, and only the holder of that transform can rotate a received ciphertext back into the frame where the lattice decodes. The hardness assumption is precisely that recovering the secret isometry from the rotated basis is infeasible.

Polar-KEM [50] instantiates this template with a Construction-D polar lattice: the public key is meant to be a rotated polar-lattice basis, encapsulation embeds a message plus a small error, and decapsulation rotates the ciphertext by the inverse isometry and runs a polar successive-cancellation (SC) decoder, wrapped by a Fujisaki–Okamoto transform for CCA security. The security of the whole construction depends on the isometry being secret.

**Our result.** It is not secret. In the reference (and optimized) implementation, the signed permutation that implements the isometry is regenerated inside decapsulation from a public seed embedded in the public key, and the secret key is read only to produce the implicit-rejection value on rejection. Decapsulation is thus a deterministic public function of the public key and the ciphertext. We turn this into a shared-secret recovery that uses public data only (Section 5.3): apply the public permutation to the ciphertext coordinates, run the public SC decoder to obtain the message, and derive the shared secret. On the submitted key-answer test (KAT) vectors of the unmodified reference implementation the recovery is exact on every record at every level (Section 5.4).

### 5.1.1 Contributions

- We show that Polar-KEM decapsulation is independent of the secret key: the isometry is derived from a public seed in the public key, so decapsulation is a public function of  $(pk, ct)$ .
- We give a shared-secret recovery from public data only, and verify it against the submitted KAT vectors of the unmodified reference implementation, recovering 10/10 shared secrets at each of the three parameter sets.
- We record three independent specification-level defects (a decapsulation radius test that rejects almost all honest ciphertexts at the smallest set, a minimum-distance contradiction that voids the decoding-radius claim, and a re-encryption check that does not bind the ciphertext), and state the structural remedy.

### 5.1.2 Related Work

Lattice-isomorphism cryptography was placed on a formal footing by Ducas and van Woerden [51], and the signature scheme HAWK [52] demonstrated that a well-chosen public lattice with a hidden isometry yields compact, efficient primitives. In every such design the security reduction is to the hardness of recovering the isometry (or an equivalent short basis) from its public rotation; the entire secret is the transform. Attacks on lattice-based KEMs typically target the underlying lattice problem or the CCA wrapper, for instance the chosen-ciphertext and side-channel key recoveries on FO-wrapped schemes [12, 13]. Our finding is of a different and simpler kind: it does not attack the LIP assumption at all. The reference implementation never keeps the isometry secret, so the trapdoor is absent and decapsulation is public. The break is closer in spirit to a specification-versus-implementation mismatch than to a mathematical cryptanalysis, but its effect is total: the shared secret follows from public data.

### 5.1.3 Techniques Overview

Two observations compose into the recovery.

- (1) The isometry is public. Decapsulation regenerates the signed permutation  $\pi$  from a public seed located at a fixed offset of the public key,

$$\pi = \text{SignedPerm}(\text{pk}[\text{seed offset}]), \quad (5.1)$$

and applies  $\pi^{-1}$  to the ciphertext. The secret key appears only in the implicit-rejection branch.

- (2) The decoder is public. A ciphertext coordinate is

$$\text{ct}[i] = \text{sign}[i] \cdot (\pm q/4) + \text{small error}, \quad (5.2)$$

so after applying  $\pi^{-1}$  the public polar SC decoder recovers the message  $\mu$ , and the public routine `derive_valid_secret( $\mu, \text{ct}$ )` returns the shared secret.

Composing them, the shared secret is a public function

$$\text{ss} = \text{Decode}(\pi^{-1}\text{ct}) \longrightarrow \text{derive\_valid\_secret}(\mu, \text{ct}), \quad \pi = \pi(\text{pk}), \quad (5.3)$$

of  $(\text{pk}, \text{ct})$  alone, computable without the secret key.

## 5.2 Preliminaries

### 5.2.1 Lattice-Isomorphism KEMs

A LIP KEM fixes a public lattice  $\Lambda$  with an efficient decoder and a secret orthonormal isometry  $O$ . The public key is a basis of  $O\Lambda$ ; encapsulation sends a lattice point of  $O\Lambda$  perturbed by a small error; decapsulation applies  $O^{-1}$  and decodes in  $\Lambda$ . Security requires that  $O$  (equivalently a short basis of  $O\Lambda$ ) cannot be recovered from the public basis.

### 5.2.2 The Polar-KEM Scheme

Polar-KEM takes  $\Lambda$  to be a Construction-D lattice built from a rate-1/2 polar code and decoded by successive cancellation. Table 5.1 lists the parameters. Algorithm 5.1 states decapsulation as it appears in the reference implementation. The line marked  $(\star)$  is the whole point of this paper: the permutation is a function of the public key, and the secret  $z$  is used only on the rejection branch.

Table 5.1: Polar-KEM parameters and byte sizes.

| set           | $N$  | $q$   | error   | pk   | sk   | ct   | ss |
|---------------|------|-------|---------|------|------|------|----|
| Polar-KEM-128 | 512  | 12289 | ternary | 1024 | 2048 | 768  | 16 |
| Polar-KEM-256 | 1024 | 12289 | ternary | 2048 | 4096 | 1280 | 32 |
| Polar-KEM-512 | 2048 | 18433 | ternary | 4096 | 8192 | 2304 | 64 |

## 5.3 The Keyless Decapsulation

**Proposition 5.1** (Secret-independent decapsulation). *On any ciphertext that decapsulation accepts, the returned shared secret is a deterministic function of  $(\text{pk}, \text{ct})$  and does not depend on the secret key.*

**Algorithm 5.1** Decaps(sk, pk, ct) as implemented

---

```

1:  $\pi \leftarrow \text{SignedPerm}(\text{pk}[\text{seed offset}])$   $\triangleright (\star)$  derived from the public key
2:  $\mu \leftarrow \text{SCdecode}(\pi^{-1}\text{ct})$   $\triangleright$  public polar decoder
3:  $\text{ct}' \leftarrow \text{Enc}(\mu; G(\mu));$  if  $\|\text{ct} - \text{Embed}(\mu)\| \leq \rho$  then accept
4: return  $H(\mu)$  if accepted, else the implicit-rejection key  $H'(z, \text{ct})$   $\triangleright z = \text{sk}$  used only here

```

---

*Proof.* By Algorithm 5.1 the permutation  $\pi$  is a function of  $\text{pk}$  alone ((5.1)), the SC decoder and Embed/Enc are public, and the accepted output is  $H(\mu)$  with  $\mu = \text{SCdecode}(\pi^{-1}\text{ct})$ . The secret  $z$  enters only the implicit-rejection branch  $H'(z, \text{ct})$ , which is not taken on an accepted ciphertext. Hence the accepted shared secret is a public function of  $(\text{pk}, \text{ct})$ .  $\square$   $\square$

Proposition 5.1 is the break. Algorithm 5.2 states the recovery; it is exactly the accepting path of decapsulation, run by the adversary from public data.

**Algorithm 5.2** Shared-secret recovery from  $(\text{pk}, \text{ct})$ 


---

**Require:** public key  $\text{pk}$ , ciphertext  $\text{ct}$

**Ensure:** the encapsulated shared secret  $\text{ss}$

```

1:  $\pi \leftarrow \text{SignedPerm}(\text{pk}[\text{seed offset}])$ 
2:  $\mu \leftarrow \text{SCdecode}(\pi^{-1}\text{ct})$ 
3: return  $\text{ss} \leftarrow \text{derive\_valid\_secret}(\mu, \text{ct})$ 

```

---

The adversary in Algorithm 5.2 needs no signing or decapsulation oracle and no secret material: given the public challenge  $(\text{pk}, \text{ct})$  it computes  $\text{ss}$  directly, so the KEM provides no confidentiality.

## 5.4 Experimental Verification

**Setup.** We link a 30-line recovery program against the unmodified reference implementation. It calls exactly two reference routines, the message recovery  $\text{recover\_message}(\text{pk}, \text{ct})$  and the secret derivation  $\text{derive\_valid\_secret}(\mu, \text{ct})$ , and never loads the secret key. For each record of the submitted KAT files we read the public  $\text{pk}$  and  $\text{ct}$ , recover  $\text{ss}$ , and compare against the KAT's  $\text{ss}$  field. Costs are measured on our test machine (104 cores, 251 GB RAM); the recovery is a single public computation and is essentially free. Table 5.2 reports the outcome.

Table 5.2: Shared-secret recovery from public data on the submitted KAT vectors of the unmodified reference implementation (104 cores, 251 GB RAM). No secret key is read.

| set           | KAT records | recovered ss | secret key used | time (10 recs) | peak RSS |
|---------------|-------------|--------------|-----------------|----------------|----------|
| Polar-KEM-128 | 10          | <b>10/10</b> | none            | 0.05 s         | 0.8 MB   |
| Polar-KEM-256 | 10          | <b>10/10</b> | none            | 0.02 s         | 0.8 MB   |
| Polar-KEM-512 | 10          | <b>10/10</b> | none            | 0.04 s         | 0.8 MB   |

The recovery matches the KAT shared secret on every record of every set. Since the program does not read a secret key, this is a shared-secret recovery from public data, confirming Proposition 5.1 on the concrete reference implementation.

## 5.5 Specification-Level Defects

Independently of the keyless-decapsulation break, the specification carries three defects that each void a stated guarantee.

- Decapsulation radius. Alg. 7 rejects when  $\|e\| > \rho$ . With ternary error  $e_i \in \{-1, 0, 1\}$  at  $p_0 = 1/2$ ,  $\|e\|^2 \sim \text{Bin}(N, 1/2)$ , so for  $N = 512$ ,  $\rho^2 = 225$  one has  $\mathbb{E}\|e\|^2 = 256 > \rho^2$  and  $\Pr[\|e\|^2 > 225] = 0.9965$ . The 128 set rejects about 99.6% of honest ciphertexts, contradicting the claimed decryption-failure probability below  $2^{-128}$ .
- Minimum distance. The rate-1/2 Construction-D level-0 code  $C_0$  enters the lattice at scale one, so  $\lambda_1^2 \leq d(C_0)$ . With the specification's frozen set,  $d(C_0) = 8$  ( $N = 512, 1024$ ) and 16 ( $N = 2048$ ), giving  $\lambda_1 \leq 2.83/2.83/4$  against  $\rho = 15/31/63$ . The decoding radius exceeds  $\lambda_1/2$ , so unique decoding is not guaranteed and the failure bound has no basis; Table 2 of the specification mis-attributes the distance to the  $2^L \mathbb{Z}^N$  component.
- Re-encryption check. Alg. 7 sets  $K = \text{Extract}(\hat{m})$  without binding  $\text{ct}$  and checks  $\|\text{ct} - \text{Embed}(\hat{m})\| \leq \rho$  rather than re-encrypting and comparing. For a challenge  $\text{ct}^*$ , querying  $\text{ct}^* + u$  with a unit vector  $u$  (with  $\|e + u\| \leq \rho$  likely at  $N \geq 1024$ ) returns the same  $K^*$ , a one-query CCA distinguisher at the specification level.

## 5.6 Countermeasure

The break exists because the isometry is public. A LIP KEM is only as secret as its isometry, so the remedy is structural: decapsulation must depend on a genuinely hidden basis. Concretely, the signed permutation must be part of the secret key and never derivable from the public key, the public key must be the rotated basis alone (with the frozen set and any decoding aid that leaks the isometry removed), and the LIP hardness must then be argued for the concrete Construction-D polar lattice. Because the polar lattice has small minimum distance relative to the stated decoding radius (Section 5.5), that argument must also re-derive the parameters. None of these touch the KEM's interface, but all are needed before the scheme provides any confidentiality.

## 5.7 Conclusion

Polar-KEM decapsulation does not use the secret key: the isometry that must be hidden is regenerated from a public seed in the public key, so recovering the shared secret is a public computation on  $(\text{pk}, \text{ct})$ . We verified the recovery against the submitted key-answer vectors of the unmodified reference implementation, obtaining the correct shared secret on all records at every parameter set without reading a secret key. Restoring security requires making decapsulation depend on a genuinely secret basis and re-deriving the decoding parameters.

## MAMBA-NIKE: static-key recovery from reconciliation

MAMBA-NIKE is a non-interactive key exchange submitted to the ICCS NGCC round-1 call, in which a responder reuses a static secret key across sessions. Its security proof establishes only passive, single-transcript real-or-random security, and the submission explicitly disclaims active security. We show that under the static-key reuse that defines a NIKE, an active adversary recovers the full static secret.

The reference responder applies lattice reconciliation directly to an attacker-chosen one-pass message with no validation and no chosen-ciphertext transform. The reconciled value is a coefficient-domain ring product of the static secret with an attacker-controlled label, so it is linear in the secret, and the dither is derived only from attacker input and is therefore known. Choosing the label as a scalar multiple of the identity makes every reconciliation bit a known distance-to-multiple-of-eight function of four secret coefficients and a two-bit help pattern, and a single query reads this function for all reconciliation groups at once. A fixed probe set of about seventy-one non-adaptive queries makes the per-group response injective over all coefficient values, and per-group maximum-likelihood decoding followed by an exact dither-subtraction refinement recovers the secret.

The recovery is complete and deterministic. Using the reference implementation’s own `-DSTATISTICAL_TEST` build, which exposes the raw reconciliation string before the SHAKE256 key-derivation step, the group-parallel readout recovers 30/30 static keys for MAMBA-NIKE-128 and 5/5 for MAMBA-NIKE-384, each in about 71 non-adaptive queries and, on our machine (104 cores, 251 GB RAM), 0.29 s and 0.77 s mean time at 4.1 MB peak resident memory. Against the deployed API, where the derivation is applied and only a strict full-key-equality reaction is observable, the group-parallel readout is lost and the same defect needs about  $2^{15}$  queries per key (still milliseconds of responder work); we state this as an analysis of the query cost, not a separately benchmarked run. The security caveat is equally plain: this does not break the passive theorem, which stands; it breaks the active static-key-reuse security that the NIKE name and category require. The remedy is a chosen-ciphertext transform binding the transcript. Attack source is available.<sup>1</sup>

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/mamba-nike-key-recovery>.

## 6.1 Introduction

A non-interactive key exchange (NIKE) lets two parties who each hold a long-term key pair agree on a shared key with no interaction [37]. The defining feature is that the long-term keys are static and reused across all sessions, so a NIKE must remain secure when an active adversary interacts with a party that reuses its key. This is exactly the setting in which lattice key exchange has repeatedly failed: when a reconciliation-based responder reuses its secret and leaks any signal about whether reconciliation succeeded, a key-mismatch or reaction adversary recovers the secret coefficient by coefficient. This attack family is well established, from Fluhrer’s observation [38] and the Ding et al. reuse attack [39] to the systematic key-mismatch treatment of Qin et al. [35].

MAMBA-NIKE [36] is submitted as a NIKE, with a static responder key reused on every derivation. Its security analysis, however, proves only passive real-or-random security for a single one-pass transcript and explicitly disclaims active, chosen-ciphertext and key-reuse security. The submission thus sits in a tension: it is named and structured as a NIKE, the one primitive whose whole point is static-key security against active adversaries, while proving only the passive property that a NIKE does not get to assume.

**Our result.** We resolve the tension on the side of the attacker. The reference responder applies reconciliation to an attacker-chosen one-pass message with no validation, so it is a reaction oracle on the reused static secret. The reconciled value is linear in the secret, and the dither is a known function of attacker input, which lets us choose inputs that turn each reconciliation bit into a known function of a few secret coefficients (Sections 6.3 and 6.4). A fixed set of about seventy-one non-adaptive queries recovers the entire static key, and the recovery is exact against the unmodified reference implementation (Section 6.5).

### 6.1.1 Contributions

- We show the MAMBA-NIKE reference responder is a reaction oracle: it reconciles unvalidated attacker input against the reused static secret, and the reconciled value is linear in the secret with a known dither.
- We give scalar-label reconciliation, a probe design in which a scalar label makes each reconciliation bit a known distance function of four secret coefficients, read for all groups in parallel, recovering the secret in about 71 non-adaptive queries.
- We verify complete, deterministic recovery against the unmodified reference at two profiles (30/30 for MAMBA-NIKE-128, 5/5 for MAMBA-NIKE-384), and state precisely how the result relates to the submission’s passive theorem.

### 6.1.2 Related Work

Freire et al. [37] formalised NIKE and its active security under static-key reuse. Reaction and key-mismatch attacks on reconciliation-based lattice key exchange with key reuse were initiated by Fluhrer [38] and Ding et al. [39], and made systematic by Qin et al. [35], who recover the secret coefficient by coefficient from a key-mismatch oracle. Our attack is in this family, with one efficiency change specific to MAMBA-NIKE: because the reconciled value is a coefficient-domain product of the secret with a chosen label, a scalar label reads one function of the secret for every reconciliation group in a single query, so the whole secret is recovered by a fixed, non-adaptive probe set rather than a per-coordinate adaptive search. The observation that this responder never validates its input, and that the dither depends only on attacker input, is what makes the labels freely choosable.

### 6.1.3 Techniques Overview

Three facts compose into the recovery.

- (1) Linearity in the secret. The responder computes, on the attacker one-pass message, a value

$$v = s_B \cdot bp \pmod{q}, \quad (6.1)$$

a coefficient-domain negacyclic ring product (the reference number-theoretic transform is a no-op), hence linear in the static secret  $s_B$ , with  $bp$  and the reconciliation help both attacker-chosen and the dither known.

- (2) Scalar labels linearize each bit. Choosing  $bp = c_0 \cdot X^0$  with  $c_0 \in \{1024, 2048, 3072\}$  so that  $8c_0 \in \{q, 2q, 3q\}$ , each reconciliation group-bit becomes a known function

$$\text{bit} = \left\lceil \sum_m \text{dist}_8((c_0/1024) s_B[i_m] - 2a_m - a_3) < 8 \right\rceil, \quad (6.2)$$

of the four secret coefficients  $i_m \in \{i, i+n/4, i+n/2, i+3n/4\}$  it mixes and the attacker's two-bit help pattern  $(a_0, a_1, a_2, a_3)$ , where  $\text{dist}_8(k)$  is the distance to the nearest multiple of eight.

- (3) Parallel readout and decoding. One query fixes one  $(c_0, \text{pattern})$  for all  $n/4$  groups at once. A fixed probe set over scalings and 256 patterns makes the per-group signature injective over all  $(2\eta + 1)^4$  coefficient values; per-group maximum-likelihood decoding, followed by subtracting the now-known exact dither, recovers  $s_B$ .

## 6.2 Preliminaries

### 6.2.1 NIKE and Static-Key Reuse

A NIKE [37] publishes long-term keys and derives, from one party's secret and another's public key, a shared value with no interaction. Because the keys are static and reused, the adversary may query a party on many chosen peer values, so NIKE security is inherently an active, key-reuse notion. A responder that reveals any function of whether a derivation “matched” is a key-mismatch or reaction oracle on its static secret.

### 6.2.2 The MAMBA-NIKE Responder

MAMBA-NIKE works over  $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$  with  $q = 8192$ ; Table 6.1 lists the profiles. The static responder derives a shared value from the peer one-pass message  $M_1 = (\mu, u, h)$  by Algorithm 6.1. Two implementation facts are central: the reference `poly_ntt` is a no-op, so `poly_pointwise` is a coefficient-domain negacyclic product; and the dither  $(du, dv)$  is derived only from the attacker's  $\mu$ .

---

**Algorithm 6.1** `DeriveB( $s_B, M_1$ )` as implemented (static responder)

---

- 1: parse  $M_1 = (\mu, u, h)$ ;  $bp \leftarrow u, \quad c \leftarrow h$  ▷ no validation of  $u, h$
  - 2:  $(du, dv) \leftarrow \text{gen\_dither}(\mu)$  ▷ depends only on attacker  $\mu$
  - 3:  $bp \leftarrow (bp \ll 3) - du \pmod{q}$
  - 4:  $v \leftarrow s_B \cdot bp \pmod{q}$  ▷ coefficient-domain product, linear in  $s_B$
  - 5: **return** `rec`( $v, c$ ) ▷ D4 reconciliation on attacker  $c$
-

Table 6.1: MAMBA-NIKE profiles ( $q = 8192$ ). This work recovers 128 and 384 end-to-end (full, reliable); 192 recovers only about 60% of keys with the shipped probe set; 256/512 use the identical responder path with a larger secret alphabet.

| profile        | $n$  | $\eta$ | status here                       |
|----------------|------|--------|-----------------------------------|
| MAMBA-NIKE-128 | 1024 | 2      | full recovery, 71 queries         |
| MAMBA-NIKE-192 | 1024 | 3      | partial ( $\approx 60\%$ of keys) |
| MAMBA-NIKE-384 | 2048 | 2      | full recovery, 71 queries         |
| MAMBA-NIKE-256 | 1024 | 7      | same path, larger alphabet        |
| MAMBA-NIKE-512 | 2048 | 5      | same path, larger alphabet        |

### 6.3 The Responder Is a Reaction Oracle

**Proposition 6.1** (Secret-linear reconciliation on chosen input). *On an attacker one-pass message, the value reconciled by Algorithm 6.1 is  $v = s_B \cdot bp \bmod q$  with  $bp$  and  $c$  attacker-chosen and the dither known to the attacker. Hence a responder that reveals any key-confirmation signal is a reaction oracle whose response is a known function of  $s_B$  and the attacker’s input.*

*Proof.* By Algorithm 6.1 the responder performs no validation of  $u, h$ ; the dither  $(du, dv)$  is a function of  $\mu$  alone, so the attacker knows it and controls  $bp = (u \ll 3) - du$ . Since the reference transform is a no-op, `poly_pointwise` is the negacyclic product, giving  $v = s_B \cdot bp$ , linear in  $s_B$ . The reconciliation  $\text{rec}(v, c)$  is a public function of  $v$  and the attacker’s  $c$ , so any signal about it is a known function of  $s_B$  and chosen input.  $\square$   $\square$

The reference exposes the reconciliation string byte-for-byte in its own `-DSTATISTICAL_TEST` build, which we use as the oracle; under static-key reuse a one-bit key-confirmation per session is the same signal.

### 6.4 Scalar-Label Reconciliation

Setting  $bp = c_0 \cdot X^0$  makes  $v = c_0 s_B$  up to a known dither-induced residual, so  $8v = q(c_0/1024) s_B + 8 \cdot \text{noise}$  and each D4 group-bit is (6.2), a known function of the four secret coefficients of that group and the two-bit help pattern. The residual  $s_B \cdot \text{residual}$  cannot be zeroed because  $bp[i] \equiv -du[i] \pmod{8}$ , but it is small: measured standard deviation about 78 and maximum magnitude 314 over 30 keys, far below the signal step, so it forces only a mild error-tolerant decode. Algorithm 6.2 states the recovery.

The probe set is non-adaptive: it is fixed in advance and one round of queries suffices. The attacker self-verifies by predicting every observed bit from the recovered  $s_B$ , since  $bp$  is known exactly.

### 6.5 Experimental Verification

**Setup.** We link the attacker against the unmodified reference implementation, using its `-DSTATISTICAL_TEST` reconciliation output as the reaction oracle. The harness holds the victim  $s_B$ , answers only oracle calls, and never inspects  $s_B$ ; the attacker replicates of the public dither and encoding are checked equal to the reference at runtime. Every recovered key is byte compared to the true  $s_B$ . Table 6.2 reports the outcome.

The scalar-label model is accurate on non-boundary bits: in our instrumented run the predicted (6.2) matched the live oracle on 11130/11172 non-boundary bits (42 mismatches,  $\approx 0.4\%$ ), with boundary cases ( $\sum \text{dist}_8 = 8$ ) decided by noise and treated as wildcards. Because the per-group maximum-likelihood decode is noise-robust, these residual mismatches do not prevent recovery,

**Algorithm 6.2** SLR: static-key recovery from the reaction oracle**Require:** reaction oracle for the static  $s_B$ , profile  $(n, \eta, q)$ **Ensure:** the full static secret  $s_B$ 

- 1: build a fixed probe set over scalings  $c_0$  and 256 help patterns whose per-group signature is injective over all  $(2\eta + 1)^4$  coefficient tuples
- 2: **for** each probe (one query) **do**
- 3:     submit  $M_1$  with  $bp = c_0 X^0$  and pattern  $c$ ; record the  $n/4$  group-bits
- 4: **end for**
- 5: **for** each group **do**
- 6:     decode the four coefficients by minimum Hamming distance to the noise-free signature
- 7: **end for**
- 8: recompute the exact dither residual  $s_B \cdot \text{residual}$  from the recovered  $s_B$  and re-decode the few ambiguous groups ▷ no extra queries
- 9: **return**  $s_B$

Table 6.2: Static-key recovery against the unmodified reference implementation, reaction oracle. All recovered keys byte-match the true secret. Cost on the test machine (104 cores, 251 GB RAM), single-threaded; “mean time” is per key and “peak RSS” the attack process’s resident memory.

| profile        | $n$  | queries | keys | full recovery | mean time | peak RSS |
|----------------|------|---------|------|---------------|-----------|----------|
| MAMBA-NIKE-128 | 1024 | 71      | 30   | <b>30/30</b>  | 0.29 s    | 4.1 MB   |
| MAMBA-NIKE-384 | 2048 | 71      | 5    | <b>5/5</b>    | 0.77 s    | 4.1 MB   |

which completes in about 71 non-adaptive queries per key at the 128 and 384 profiles. Under a strict full-key-equality oracle (one bit per query) the group-parallel readout is lost and the count inflates by an  $O(n/4)$  factor to about  $2^{15}$  queries, still milliseconds of responder work, and the recovery conclusion is unchanged.

## 6.6 Scope

We state the relationship to the submission’s claims precisely.

- The passive theorem stands. The submission proves passive, single-transcript real-or-random security (App. B.3), and this attack does not contradict it: it uses many queries against a reused static key, outside that model.
- The NIKE claim does not. A NIKE [37] is by definition a static-key primitive that must resist active adversaries under key reuse. MAMBA-NIKE is named, structured with a reused static  $s_B$ , and categorized as a NIKE, yet provides no security once any key-confirmation signal exists under reuse. The submission’s argument that the interface “exposes no decapsulation oracle” does not hold for a NIKE, because reuse of  $s_B$  across sessions is the oracle.

The finding is therefore a category and claim mismatch backed by a working full key recovery, not a refutation of the stated passive theorem.

## 6.7 Countermeasure

The recovery exists because the responder reconciles unvalidated attacker input against a reused static secret and leaks a signal about the result. A NIKE responder must not do this. The remedy is a chosen-ciphertext transform that binds the whole transcript before any secret-dependent reconciliation is exposed, so that a reaction signal reveals nothing about  $s_B$ ; equivalently, the

scheme must be built and analysed to the active, key-reuse security bar that its NIKE category demands rather than the passive one it currently proves. The underlying lattice parameters are not at issue: an independent estimate reproduces the submission’s claimed passive hardness levels, and the break is entirely in the active interface.

## 6.8 Conclusion

MAMBA-NIKE reconciles unvalidated attacker input against its reused static secret, and the reconciled value is linear in that secret with a known dither. Choosing a scalar label makes each reconciliation bit a known function of four secret coefficients that a single query reads for all groups, so a fixed set of about seventy-one non-adaptive queries recovers the entire static key, verified end-to-end against the unmodified reference at two profiles. The passive security theorem is untouched; the static-key security that the NIKE name and category require is not. Meeting it requires a chosen-ciphertext transform binding the transcript.

## Shuttle: key recovery from a rejection-free sampler

Shuttle is a compact lattice Fiat–Shamir signature submitted to the ICCS NGCC round-1 call, in the rejection-sampling-with-aborts lineage; it removes the abort branch of its iterative sampler at compact width and licenses the removal with a Rényi-divergence bound asserting that the released signature is independent of the secret key. We refute this assertion on the unmodified reference implementation and give a practical key-recovery attack.

The signing response  $z = y + sc'$  is assembled by a rejection-free transition that, for each secret shift  $v = X^j s$ , aligns the sign of  $v$  to the running state before adding  $\pm v$ . We prove that this sign alignment leaves every first moment of  $\langle z, sX^j \rangle$  at zero, so the per-step marginal the Rényi bound controls is unaffected, while it raises the second moment, on the challenge support, by a key-independent constant  $\kappa = 1 + 2\varrho$  that is orthogonal to that bound. The resulting *sign-alignment correlation attack* (SACA) reads this second-order leakage with a bilinear estimator that converges to  $\kappa s$ , fixes the scale  $\kappa$  from the public key relation alone, completes the error term, and forges.

Using only the public key and a signing oracle, SACA recovers all 768, 1536 and 3072 secret coefficients of Shuttle-128, -256 and -512 and forges signatures the reference verifier accepts. On our machine (104 cores, 251 GB RAM, single core used) the three sets take 12.7, 34.6 and 52.7 minutes end to end, dominated by signature collection, with the estimator and forgery finishing in seconds at under 100 MB resident memory; we also state the two countermeasures the leak dictates.<sup>1</sup>

---

**Prior public disclosure.** This result was publicly reported first. Martin Feussner reported the sign-handling leak in the Shuttle sampler on the NIST `pqc-forum` around 25 September 2026 and on the ICCS PKC forum on 27 September 2026, and it was independently confirmed by Markku-Juhani O. Saarinen on the community tracker <https://ngcc.dev>. Priority belongs to those reports; this chapter is independent verification only.

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/shuttle-key-recovery>.

## 7.1 Introduction

Digital signatures are among the first primitives migrating to post-quantum assumptions, and lattice constructions dominate the standardised and candidate designs. Two paradigms produce them: hash-and-sign over trapdoor lattices, of which Falcon is the representative, and Fiat–Shamir with aborts, introduced by Lyubashevsky [1, 2] and realised by Dilithium/ML-DSA. In the Fiat–Shamir paradigm the signer proves knowledge of a short secret  $s$  satisfying a module relation  $b = As + e$ , and the transcript releases a response of the form

$$z = y + sc', \quad (7.1)$$

in which the short secret  $s$  appears additively and is masked by a wide vector  $y$ .

**The hiding requirement.** A Fiat–Shamir signature is zero-knowledge only if the released  $z$  follows a law independent of  $s$ ; otherwise each signature carries information about the secret, and enough transcripts recover it. The canonical device that enforces independence is *rejection sampling* [2]: the signer aborts and restarts with a prescribed probability, so that, conditioned on acceptance,  $z$  follows a fixed, secret-independent target distribution  $\rho_{\text{tgt}}$ . The abort branch is the mechanism that makes the hiding *exact*, and the expected number of restarts is the price paid for it. The engineering history of the paradigm is a search for the same hiding at a lower restart cost, since a narrower target yields shorter signatures but a higher abort probability.

**Iterative rejection sampling and its abort-free variant.** The most recent step in that search is *iterative rejection sampling* (IRS) [3], which builds the response one shift at a time so that each step’s abort probability is minute. IRS additionally shows that if the target distribution is deliberately *widened*, in the manner of Raccoon [5], then the abort branch can be dropped entirely: a signature that should have been rejected is then emitted with probability below  $2^{-108}$ , and a single such emission leaks a negligible amount. The framework thus certifies two regimes in which the released  $z$  hides  $s$ : aborting at compact width, and abort-free at wide width.

**The scheme we analyse.** Shuttle [4] is a compact module Fiat–Shamir signature submitted to the ICCS Next-Generation Commercial Cryptography (NGCC) call, built on the IRS technique. To obtain both short signatures and a constant-time, restart-free signer, Shuttle takes a third route: it *removes the abort branch while keeping the compact width*. In place of the abort it inserts a deterministic transition that, on every step, emits one of two shifted candidates, and it controls the distortion this introduces by a Rényi-divergence bound, concluding that the signature distribution is independent of the secret key. Signing proceeds in three stages.

- **Commitment.** The signer draws a mask  $y$  and forms a commitment  $w = \text{Commit}(A, b, y)$  from the public key  $(A, b)$ .
- **Challenge.** A sparse challenge  $c = H(w, \mu)$  of weight  $\tau$  is derived from the message digest  $\mu$ ; its signed form is

$$c' = \sum_{i=1}^{\tau} b_i X^{j_i}, \quad b_i \in \{-1, +1\}, \quad sc' = \sum_{i=1}^{\tau} b_i v_i, \quad v_i := X^{j_i} s. \quad (7.2)$$

- **Response.** Starting from  $z^{(0)} = y$ , the iterative sampler applies one transition per shift  $v_i$ , producing  $z^{(i)} = T_{v_i}(z^{(i-1)})$  and the released  $z = z^{(\tau)}$ ; a public norm bound  $B_v$  on  $\|z\|$  gates the output.

The single design choice under scrutiny is the transition  $T_v$ : it computes  $t = \langle z, v \rangle$ , *aligns* the sign of  $v$  so that  $t > 0$ , and then adds  $\pm v$  by an interval test. Whether this abort-free, sign-aligning transition achieves the hiding (7.1) requires at compact width is the question of this work.

**Our results.** It does not. We show that the sign alignment makes the released response carry a constant, secret-dependent *second-order* correlation, and we turn this into a practical key recovery, the *sign-alignment correlation attack* (SACA). Against the unmodified reference implementation, from the public key and a signing oracle alone, SACA recovers the entire secret of Shuttle-128, -256 and -512 from  $3\text{--}6 \times 10^5$  signatures on one core in minutes, and forges signatures the reference verifier accepts. The attack is a transcript-only, black-box attack that never touches the underlying MLWE/MSIS problems.

### 7.1.1 Related Work

**The design philosophy of the baseline.** The line Shuttle belongs to is organised around one invariant: the released response must be drawn from a public target distribution, and the deviation of the real signer from that target must be provably small. In Lyubashevsky’s scheme [1, 2] the deviation is made *exactly* zero by rejection: a candidate  $z = y + sc$  is emitted with probability  $\min(1, \rho_{\text{tgt}}(z)/(M\rho_{sc}(z)))$ , so the accepted  $z$  is distributed exactly as  $\rho_{\text{tgt}}$ , independent of  $s$ . The rejection rate  $M$  grows as the target narrows, so compact signatures force frequent restarts. Bai–Galbraith, Dilithium and their successors tune the target and the norm bounds to balance size against  $M$ , but retain the abort. Raccoon [5] takes the opposite extreme, widening the target with added noise so the deviation is negligible *without* any abort, at the cost of larger signatures. IRS [3] interpolates: by constructing the response through a sequence of small shifts, each conditioned on the rejection event, it drives the per-step abort probability so low that either a very small  $M$  (compact, aborting) or a modest widening (wide, abort-free) suffices. In every one of these designs the abort or the widening discharges the deviation; the invariant is never weakened, only met by different means. Shuttle is the first to keep the compact target *and* drop the abort, replacing the discharge by an accumulated Rényi bound on a deterministic two-candidate transition. Table 7.1 places the designs on the two axes that control the hiding.

Table 7.1: Iterative-sampling and rejection-sampling designs on the two axes that control the hiding: whether an abort discharges the deviation, and the width of the target distribution.

| Design              | Abort | Target width | Deviation discharged by       |
|---------------------|-------|--------------|-------------------------------|
| FS with aborts [2]  | yes   | any          | exact rejection               |
| Raccoon [5]         | no    | wide         | additive noise / masking      |
| IRS, aborting [3]   | yes   | compact      | exact rejection, low $M$      |
| IRS, abort-free [3] | no    | wide         | $< 2^{-108}$ would-be-rejects |
| Shuttle [4]         | no    | compact      | accumulated Rényi bound       |

**Cryptanalysis of imperfect response distributions.** Recovering a secret from a response distribution that is not perfectly hiding is a recurring theme. The parallelepiped-learning attacks of Nguyen and Regev [6] collapsed GGH and NTRUSign by fitting the secret basis to the empirical support of many signatures; later work on hash-and-sign forced the adoption of exact Gaussian samplers in Falcon and its descendants. Statistical and side-channel leakage from Fiat–Shamir signers has likewise been studied, from fault-induced abort leakage to the timing of rejection loops. In all of these the bias is a sampler imperfection to be removed by more precision or by masking. The present work differs in the source of the bias: it is a *structural* sign decision built into the transition, present at any arithmetic precision, and it appears exactly when the abort is dropped at compact width. To our knowledge no prior analysis targets the abort-free-compact regime, and no public cryptanalysis of Shuttle exists.

### 7.1.2 Our Contributions

We determine that the sign-aligning, abort-free transition of Shuttle fails to hide the secret at compact width, and we develop the attack this failure enables into a self-contained method. Our contributions are as follows.

- **A leakage theory for sign-aligned sampling.** We model the rejection-free transition as a Markov kernel on the shift lattice (Definition 7.3) and prove that sign alignment preserves  $\mathbb{E}[\langle z, sX^j \rangle] = 0$  for every  $j$  but raises the second moment on the challenge support by a key-independent constant  $\kappa = 1 + 2\rho$  (Theorem 7.5). We compute the off-support second moment in closed form (Proposition 7.4), isolate the alignment covariance  $\rho$  in a Gaussian model (Proposition 7.7), and match  $\kappa \approx 2$  against the implementation.
- **The sign-alignment correlation attack.** We introduce SACA (Algorithm 7.2): a bilinear estimator that converges to  $\kappa s$  (Proposition 7.8) with a variance bound (Lemma 7.9) and an explicit sample complexity for exact recovery of all  $\ell n$  coordinates (Theorem 7.10), a secret-free calibration of  $\kappa$  from the public key (Lemma 7.11), and error-term completion yielding universal forgery (Theorem 7.12).
- **Analysis of the security reduction.** We prove that the Rényi bound underlying Shuttle’s proof controls a per-step marginal relative error and does not bound the cross-coordinate statistic SACA reads (Theorem 7.13); the reduction is vacuous at the scheme’s parameters.
- **Full experimental validation.** We recover 768/1536/3072 coefficients and forge under the reference verifier for the three parameter sets (Section 7.5), with a public artifact, and record the two countermeasures dictated by the framework Shuttle builds on (Section 7.8).

### 7.1.3 Techniques Overview

The attack rests on a single observation about the transition  $T_v$  and on a statistic tailored to it.

**The observation: sign alignment is a state-dependent bias.** Write the response as  $z = y + \sum_{i=1}^{\tau} b_i v_i$  with  $v_i = X^{j_i} s$ , the mask  $y$  drawn from the wide target and the  $\tau$  shifts fixed by the challenge. A hiding signer is characterised by one property of the coefficients  $b_i$ : each is symmetric,  $\Pr[b_i = +1] = \Pr[b_i = -1] = \frac{1}{2}$ , and *independent of the running state*  $z^{(i-1)} = y + \sum_{i' < i} b_{i'} v_{i'}$ . Under that property the increment  $b_i v_i$  is a mean-zero perturbation uncorrelated with everything already accumulated, the partial sums form a symmetric random walk on the coset  $y + \Lambda_c$  of the shift sublattice  $\Lambda_c = \langle v_1, \dots, v_{\tau} \rangle$ , and the released  $z$  is distributed as the public target regardless of  $s$ ; this is exactly what rejection or a wide target buys, and what (7.14) certifies at the level of one-dimensional marginals.

Shuttle’s transition breaks the independence while preserving the symmetry. It sets

$$b_i = \text{flag}_i \cdot \text{sgn}(\langle z^{(i-1)}, v_i \rangle), \quad (7.3)$$

first *aligning* the shift so that the added vector has positive inner product with the current state, and only then applying the sign-symmetric interval-test flag  $\text{flag}_i$ . The factor  $\text{flag}_i$  keeps  $\mathbb{E}[b_i] = 0$ , so no first-order statistic sees anything: for every  $j$ ,

$$\mathbb{E}[\langle z, X^j s \rangle] = \langle \mathbb{E}[y], X^j s \rangle + \sum_i \mathbb{E}[b_i] \langle v_i, X^j s \rangle = 0. \quad (7.4)$$

The alignment factor  $\text{sgn}(\langle z^{(i-1)}, v_i \rangle)$ , however, correlates the emitted sign with the state, and this correlation surfaces at second order. Consider the projection of the released response onto a single shift direction  $X^j s$  and normalise it,  $Q_j = \langle z, X^j s \rangle / \|s\|^2$ . When  $j$  is *not* a challenge

position,  $Q_j$  is a sum of the mask projection and  $\tau$  terms  $b_i \gamma_s(j - j_i) / \|s\|^2$  whose autocorrelations  $\gamma_s(j - j_i) = \langle X^{j_i} s, X^j s \rangle$  are all off-diagonal, hence  $O(n^{-1/2})$ ;  $Q_j$  is then a generic projection of  $z$ , with

$$\mathbb{E}[Q_j^2 \mid j \notin \text{supp}(c)] = \frac{\sigma_z^2}{\|s\|^2} (1 + o(1)), \quad (7.5)$$

$\sigma_z^2$  the coordinate variance of  $z$ . When  $j = j_{i_0}$  is a challenge position, one term is diagonal,  $\gamma_s(0) = \|s\|^2$ , and  $Q_j$  splits as

$$Q_j = \beta_{i_0} + R_{i_0}, \quad \beta_{i_0} = b_{i_0} \in \{-1, +1\}, \quad (7.6)$$

with  $R_{i_0}$  the remaining, off-diagonal part. Expanding the second moment,

$$\mathbb{E}[Q_j^2 \mid j \in \text{supp}(c)] = \underbrace{\mathbb{E}[\beta_{i_0}^2]}_{=1} + 2 \underbrace{\text{Cov}(\beta_{i_0}, R_{i_0})}_{=\varrho} + \underbrace{\mathbb{E}[R_{i_0}^2]}_{=\sigma_z^2/\|s\|^2}, \quad (7.7)$$

because  $\beta_{i_0}$  takes values  $\pm 1$  and  $R_{i_0}$  reproduces the off-support law. Subtracting (7.5) isolates the excess

$$\frac{\mathbb{E}[\langle z, sX^j \rangle^2 \mid j \in \text{supp}(c)] - \mathbb{E}[\langle z, sX^j \rangle^2 \mid j \notin \text{supp}(c)]}{\|s\|^2} = \kappa = 1 + 2\varrho. \quad (7.8)$$

Two mechanisms contribute the two summands. The unit is the energy of the emitted  $\pm 1$  sign, present in *any* two-candidate step and carrying no information about  $s$  on its own. The excess  $2\varrho$  is the signature of the alignment: by (7.3) the emitted  $\beta_{i_0}$  satisfies  $\beta_{i_0} \langle z^{(i_0-1)}, v_{i_0} \rangle = \text{flag}_{i_0} |\langle z^{(i_0-1)}, v_{i_0} \rangle| \geq 0$  up to the flag, so  $\beta_{i_0}$  is positively correlated with the partial-state projection onto  $v_{i_0}$ , which is itself a component of  $R_{i_0}$ ; hence  $\varrho > 0$ . Were the alignment absent,  $\beta_{i_0} = \text{flag}_{i_0}$  would be independent of  $R_{i_0}$ ,  $\varrho$  would vanish, and  $\kappa$  would collapse to the information-free value 1. The excess is thus a direct, quantitative fingerprint of line 2, it is localised to the challenge support, and it is a *constant*, not a  $2^{-\Theta(\lambda)}$  quantity: empirically  $\kappa \approx 2$  ( $\varrho \approx 0.5$ ) on all three parameter sets. A key-recovery attack now needs only a statistic that reads this constant excess and attributes it to the right coordinate of  $s$ ; because (7.4) makes every linear functional blind, that statistic must be second order.

**The statistic: a coherent bilinear average.** The excess (7.8) is a covariance, invisible to any statistic linear in  $z$ . SACA reads it with a *bilinear* average that realigns each challenge position by the inverse shift before accumulating,

$$\widehat{s}_i[k] = \frac{1}{M} \sum_{\text{sigs}} \sum_{j \in \text{supp}(c)} z_0[j] (X^{-j} z_i)[k] \xrightarrow{M \rightarrow \infty} \kappa s_i[k]. \quad (7.9)$$

The back-shift  $X^{-j}$  maps every challenge position onto a common coordinate frame, so the per-position excesses add coherently along  $s$  while the mask  $y$ , being zero-mean and independent, cancels. Rounding  $\widehat{s}/\kappa$  recovers  $s$  once the per-coordinate deviation falls below  $1/2$ ; Theorem 7.10 turns the variance of (7.9) into the number of signatures that guarantees it for all  $\ell n$  coordinates simultaneously.

**From the secret to a forgery.** The scale  $\kappa$  is unknown but is fixed *without any secret* by the public key relation  $b = As + e$ : only the correct scale rounds  $\widehat{s}/\kappa$  to integers consistent with  $b$  (Lemma 7.11). The error term is then  $\widehat{e} = b - A\widehat{s}$ , and the reconstructed  $(s, \widehat{e})$  is a signing key that forges under the public key (Theorem 7.12). The security reduction does not exclude this: its bound concerns the marginal density of one step, whereas SACA reads a cross-coordinate correlation (Theorem 7.13).

## 7.2 Preliminaries

### 7.2.1 Rings, Norms and Shifts

Let  $n$  be a power of two,  $\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$  and  $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$ ; elements are identified with coefficient vectors in  $\mathbb{Z}^n$ . For  $a, b \in \mathcal{R}$ ,  $\langle a, b \rangle = \sum_{k=0}^{n-1} a_k b_k$  and  $\|a\| = \sqrt{\langle a, a \rangle}$ . Multiplication by  $X^j$  is the negacyclic shift

$$(X^j a)_k = \begin{cases} a_{k-j}, & k \geq j, \\ -a_{k-j+n}, & k < j, \end{cases} \quad (7.10)$$

an isometry of  $\mathcal{R}$  with inverse  $X^{-j} = -X^{n-j}$ . Define the signed autocorrelation of  $s \in \mathcal{R}$ ,

$$\gamma_s(\delta) = \langle s, X^\delta s \rangle, \quad \gamma_s(0) = \|s\|^2. \quad (7.11)$$

For  $s$  with independent zero-mean bounded coordinates,  $\mathbb{E}[\gamma_s(\delta)] = 0$  and  $\gamma_s(\delta)/\|s\|^2 = O(n^{-1/2})$  for  $\delta \neq 0$ . For a challenge  $c$ ,  $\text{supp}(c)$  denotes the positions of its nonzero coefficients.

### 7.2.2 Rényi Divergence

For distributions  $P, Q$  with  $\text{Supp}(P) \subseteq \text{Supp}(Q)$  and order  $a > 1$ ,

$$R_a(P\|Q) = \left( \sum_x \frac{P(x)^a}{Q(x)^{a-1}} \right)^{1/(a-1)}. \quad (7.12)$$

We use two facts.

**Lemma 7.1** (Multiplicativity).  $R_a(P_1 \times P_2\|Q_1 \times Q_2) = R_a(P_1\|Q_1) R_a(P_2\|Q_2)$ ; hence over  $N$  i.i.d. samples,  $\log R_a(P^{\otimes N}\|Q^{\otimes N}) = N \log R_a(P\|Q)$ .

**Lemma 7.2** (Relative error, [7]). If  $1 - \delta \leq P(x)/Q(x) \leq 1 + \delta$  for all  $x$ , then  $R_a(P\|Q) \leq 1 + \frac{a}{2}\delta^2$  to leading order.

### 7.2.3 The Shuttle Scheme

Shuttle is a module Fiat–Shamir signature with public key  $(A, b = As + e)$  over  $\mathcal{R}_q$ , secret  $s \in \mathcal{R}^\ell$ , error  $e \in \mathcal{R}^m$  of low norm, and sparse challenge  $c$  of weight  $\tau$ ; the response (7.1) is built from the shifts (7.2). The response vector  $z = (z_0, z_1, \dots, z_\ell, \dots)$  has a challenge-carrying block  $z_0$  and blocks  $z_{1..\ell}$  carrying  $s$ ; verification recomputes the commitment from  $(A, b)$ , checks  $c = H(w, \mu)$  and  $\|z\| \leq B_v$ , and depends on the secret only through  $b = As + e$ . Table 7.2 lists the parameters.

Table 7.2: Shuttle parameters (reference implementation).

| Set         | $n$  | $q$   | $\ell$ | $m$ | $\tau$ | $\ell n$ |
|-------------|------|-------|--------|-----|--------|----------|
| Shuttle-128 | 256  | 15361 | 3      | 3   | 42     | 768      |
| Shuttle-256 | 512  | 61441 | 3      | 2   | 58     | 1536     |
| Shuttle-512 | 1024 | 59393 | 3      | 2   | 115    | 3072     |

### 7.2.4 The Rejection-Free Transition

The response is assembled by applying one transition per shift  $v \in \{v_1, \dots, v_\tau\}$  to the running state. Algorithm 7.1 reproduces the transition  $T_v$  from the specification and reference code.

---

**Algorithm 7.1** Rejection-free transition  $T_v$  (per shift), as specified
 

---

**Require:** running response  $z$ , secret shift  $v = X^j s$ , fresh randomness  $u$ 
**Ensure:** updated  $z$ 

- 1:  $t \leftarrow \langle z, v \rangle$
  - 2: **if**  $t \leq 0$  **then**  $v \leftarrow -v$   $\triangleright$  sign alignment: hence  $\langle z, v \rangle > 0$
  - 3:  $\text{flag} \leftarrow \pm 1$  by an interval test on  $u$
  - 4:  $z \leftarrow z + \text{flag} \cdot v$
- 

### 7.2.5 The Rényi-HVZK Reduction

Shuttle's UF-CMA proof invokes the Rényi-HVZK-to-CMA reduction of [3]: with  $T_1$  the real signing distribution,  $T_2$  its ideal-Gaussian-input variant, and  $T_3$  a secret-independent simulator,

$$\text{Adv}^{\text{CMA}} \leq M_1^{BQ_S} M_2^{BQ_S(a-1)/a} (\text{Adv}^{\text{UF-NMA}})^{(a-1)^2/a^2} + \text{negl}, \quad (7.13)$$

$M_2 = R_a(T_2 \| T_3)$ . Non-vacuousness at  $Q_S \leq 2^{64}$  forces  $M_2 = 1 + O(2^{-64})$ . The bound on  $M_2$  is obtained from the per-step marginal relative error

$$\varepsilon_v = \max_y \left| \frac{T_v(\rho_g)(y)}{\rho_g(y)} - 1 \right|, \quad (7.14)$$

composed into  $\varepsilon_\tau = (1 + \varepsilon)^\tau - 1$  and converted by Lemma 7.2.

## 7.3 Leakage of the Rejection-Free Sampler

### 7.3.1 The Transition Kernel

Fix a key and a challenge  $c$  of support  $\{j_1, \dots, j_\tau\}$ ; the signer applies  $T_{v_1}, \dots, T_{v_\tau}$  to  $z^{(0)} = y$  and releases  $z = z^{(\tau)}$ .

**Definition 7.3** (Sign-aligned increment). The transition of Algorithm 7.1 maps  $z^{(i-1)} \mapsto z^{(i)} = z^{(i-1)} + \beta_i v_i$ , with

$$\beta_i = \text{flag}_i \cdot \text{sgn}(\langle z^{(i-1)}, v_i \rangle), \quad \langle z^{(i-1)}, \beta_i v_i \rangle = \text{flag}_i |\langle z^{(i-1)}, v_i \rangle|. \quad (7.15)$$

Absent line 2,  $\beta_i = \text{flag}_i$  is symmetric and independent of the state, and the walk is the symmetric one whose stationary law is the secret-independent target assumed by (7.14). Line 2 ties  $\beta_i$  to  $\text{sgn}(\langle z^{(i-1)}, v_i \rangle)$ .

### 7.3.2 Moment Analysis

Fix  $j$  and consider the normalised projection of the released response onto  $X^j s$ ,

$$Q_j = \frac{\langle z, X^j s \rangle}{\|s\|^2} = \frac{\langle y, X^j s \rangle}{\|s\|^2} + \sum_{i=1}^{\tau} \beta_i \frac{\gamma_s(j - j_i)}{\|s\|^2}, \quad (7.16)$$

using  $\langle v_i, X^j s \rangle = \gamma_s(j - j_i)$ ; the  $i$ -th term is  $\beta_i$  when  $j = j_i$  and  $O(n^{-1/2})$  otherwise.

**Proposition 7.4** (Off-support second moment). *Let  $\sigma_z^2$  be the coordinate variance of  $z$  and  $y$  zero-mean, independent of the shifts. For  $j \notin \text{supp}(c)$ ,  $\mathbb{E}[Q_j] = 0$  and  $\mathbb{E}[Q_j^2] = \sigma_z^2 / \|s\|^2 (1 + o(1))$ .*

**Theorem 7.5** (Second-order leakage on the support). *Under the assumptions of Proposition 7.4, for  $j = j_{i_0} \in \text{supp}(c)$ ,*

$$\mathbb{E}[Q_j] = 0, \quad \mathbb{E}[Q_j^2] = \frac{\sigma_z^2}{\|s\|^2} + \kappa(1 + o(1)), \quad \kappa = 1 + 2\rho, \quad (7.17)$$

where  $\rho = \text{Cov}(\beta_{i_0}, R_{i_0}) \geq 0$  and  $R_{i_0} = Q_j - \beta_{i_0}$  is the remainder of (7.16). Without line 2,  $\beta_{i_0} = \text{flag}_{i_0}$  is independent of  $R_{i_0}$ , so  $\rho = 0$  and  $\kappa = 1$ .

**Corollary 7.6** (Numerical value). *For Shuttle-128,  $\mathbb{E}[Q_j^2 \mid \text{on}] = 10.09$  and  $\mathbb{E}[Q_j^2 \mid \text{off}] = 8.09$ , whence  $\kappa = 2.00$ ,  $\rho = 0.50$ ; the off-support value matches  $\sigma_z^2/\|s\|^2 = (826.6/290.7)^2 = 8.08$ .*

Full proofs are in Section 7.10. Theorem 7.5 isolates the signal: a constant, key-independent second-moment excess on the support, invisible to any statistic linear in  $z$  and readable only by a second-order statistic.

### 7.3.3 The Leakage Constant

We isolate the origin of  $\rho$  in a Gaussian model retaining sign alignment and the magnitude-dependence of the flag. For the support step  $i_0$  write

$$g = \frac{\langle z^{(i_0-1)}, v_{i_0} \rangle}{\|s\|^2} \sim \mathcal{N}(0, \sigma_g^2), \quad Q_j = g + \beta_{i_0}, \quad \beta_{i_0} = \text{flag}(u, |g|\|s\|^2) \text{sgn}(g). \quad (7.18)$$

**Proposition 7.7** ( $\kappa$  in the Gaussian model). *With (7.18),*

$$\mathbb{E}[Q_j] = 0, \quad \mathbb{E}[Q_j^2] = \sigma_g^2 + \underbrace{\mathbb{E}[\text{flag}^2]}_1 + 2 \underbrace{\mathbb{E}[|g| \text{flag}(u, |g|\|s\|^2)]}_\rho, \quad (7.19)$$

so  $\kappa = 1 + 2\rho$  with  $\rho = \text{Cov}(|g|, \phi)$  for  $\phi(t) = \mathbb{E}_u[\text{flag}(u, t)]$ . If the flag is independent of  $t$  (in particular if line 2 is removed),  $\rho = 0$  and  $\kappa = 1$ ; if  $\phi$  is increasing in  $t$ , as for the affine boundary test of Algorithm 7.1, then  $\rho > 0$ .

The two additive units of the measured  $\kappa = 2$  are thus the flag energy  $\mathbb{E}[\text{flag}^2] = 1$ , present in any two-candidate transition, and  $2\rho \approx 1$  contributed by sign alignment through the magnitude-dependent interval test; the exact  $\rho$  is a functional of the boundary law and is taken as 0.50 from measurement. The qualitative dichotomy  $\kappa = 1$  without alignment,  $\kappa > 1$  with it, is what the attack and the control experiment require.

## 7.4 The Sign-Alignment Correlation Attack

SACA recovers the secret in three phases: a bilinear estimator that reads the leakage of Theorem 7.5 as a multiple of  $s$ ; a secret-free calibration that removes the multiple using the public key; and a completion that reconstructs the full signing key and forges. We describe each, then state the algorithm and its cost.

### 7.4.1 The Correlation Estimator

For a batch of signatures  $\{(c^{(t)}, z^{(t)})\}_{t=1}^N$  define, for each secret block  $i \in \{1, \dots, \ell\}$  and coordinate  $k$ ,

$$\hat{s}_i[k] = \frac{1}{M} \sum_{t=1}^N \sum_{j \in \text{supp}(c^{(t)})} z_0^{(t)}[j] (X^{-j} z_i^{(t)})[k], \quad M = N\tau. \quad (7.20)$$

The recovered information is the block-wise secret polynomial  $s_i$ : the estimator correlates the challenge-carrying coordinate  $z_0[j]$  at each support position with the back-shifted secret block  $X^{-j} z_i$ , so the second-order excess of Theorem 7.5 accumulates along  $s_i$ .

**Proposition 7.8** (Consistency). *There is  $\kappa > 0$  with  $\mathbb{E}[\hat{s}_i[k]] = \kappa s_i[k] + o(1)$  for all  $i, k$ .*

**Lemma 7.9** (Estimator variance).  *$\text{Var}(\hat{s}_i[k]) = \Sigma^2/M$  with  $\Sigma^2 \leq \sigma_z^4 C$  for an absolute constant  $C$ .*

**Theorem 7.10** (Exact recovery). *Rounding  $\hat{s}/\kappa$  coordinate-wise recovers  $s$  exactly with probability  $\geq 1 - \eta$  provided*

$$M \geq \frac{8\Sigma^2}{\kappa^2} \ln \frac{2\ell n}{\eta} = O\left(\frac{\sigma_z^4}{\kappa^2} \log \frac{\ell n}{\eta}\right). \quad (7.21)$$

#### 7.4.2 Secret-Free Calibration

The estimator returns  $\kappa s$  with  $\kappa$  unknown; the scale is removed by the public relation  $b = As + e$ , in which  $e$  is short.

**Lemma 7.11** (Scale calibration). *Let  $\hat{s} = \kappa s + \zeta$  and  $s'(\kappa') = \text{round}(\hat{s}/\kappa')$ . Then*

$$\kappa = \arg \min_{\kappa' > 0} \|b - A s'(\kappa')\|_\infty, \quad (7.22)$$

*and at the minimiser  $\|b - As'(\kappa)\|_\infty = \|e\|_\infty$ . The right-hand side uses only public data.*

#### 7.4.3 Completion and Forgery

**Theorem 7.12** (Full key recovery and universal forgery). *Given  $\hat{s}$  with  $\text{round}(\hat{s}/\kappa) = s$  and the public key, set  $\hat{e} = b - A \text{round}(\hat{s}/\kappa)$  (centred mod  $q$ ). Then  $(s, \hat{e})$  is a valid signing key: with the public transcript hash  $\text{tr} = H(\text{pk})$  and any master seed, signatures produced by Algorithm 7.1 on any message verify under  $\text{pk}$ .*

#### 7.4.4 The Attack Algorithm

Algorithm 7.2 states SACA. Its correctness is Proposition 7.8 and Theorem 7.10 (estimator), Lemma 7.11 (calibration) and Theorem 7.12 (forgery).

---

**Algorithm 7.2** SACA: sign-alignment correlation attack on Shuttle

---

**Require:** public key  $\text{pk}$ , signing oracle  $\mathcal{O}_{\text{sk}}$ , query budget  $N$

**Ensure:** a signing key that forges under  $\text{pk}$

```

1:  $(A, b) \leftarrow \text{unpackPK}(\text{pk}); \quad \hat{s} \leftarrow 0$  ▷ public data only
2: for  $t = 1$  to  $N$  do
3:    $m \leftarrow \text{fresh message}; (c, z) \leftarrow \mathcal{O}_{\text{sk}}(m)$ 
4:   for  $j \in \text{supp}(c)$ , all  $i, k$  do
5:      $\hat{s}_i[k] \leftarrow \hat{s}_i[k] + z_0[j] \cdot (X^{-j} z_i)[k]$  ▷ Eq. (7.20)
6:   end for
7: end for
8:  $\kappa \leftarrow \arg \min_{\kappa'} \|b - A \text{round}(\hat{s}/\kappa')\|_\infty$  ▷ Lemma 7.11
9:  $s \leftarrow \text{round}(\hat{s}/\kappa); \quad \hat{e} \leftarrow b - As$  (centred mod  $q$ )
10: return  $(s, \hat{e})$  ▷ Theorem 7.12
```

---

The accumulation is  $O(N\tau \ell n)$  integer multiply-adds, the calibration sweep  $O(\ell n)$  per trial scale, and the completion one polynomial product; memory is  $O(\ell n)$ . The wall-clock is that of the signer on  $N$  messages.

## 7.5 Experimental Verification

We run SACA against the unmodified Shuttle reference implementation, changing nothing in the scheme; the adversary calls the reference signer as a black box and reads only the public key.<sup>2</sup> Every recovered key is checked coefficient-by-coefficient against the true secret—opened only for scoring, never used in the recovery—and by forging a fresh signature the reference verifier accepts. All runs are single-core.

- **Leakage.** With the instrumented sampler,  $2 \times 10^4$  signatures of Shuttle-128 give  $\mathbb{E}[Q_j^2 \mid \text{on}] = 10.09$ ,  $\mathbb{E}[Q_j^2 \mid \text{off}] = 8.09$ , first moments  $-0.005$  and  $-0.007$ , i.e.  $\kappa = 2.00$  (Corollary 7.6).
- **Convergence.** Table 7.3 shows the estimator (7.20) converging monotonically on Shuttle-128; by  $3 \times 10^5$  signatures all 768 coordinates are exact and the calibrated scale is  $\kappa = 1.85$ , consistent with Theorem 7.10; Figure 7.1 plots the trajectory.
- **Recovery and forgery.** Table 7.4 reports full recovery and a verifying forgery for all three parameter sets.
- **Control.** Reinstating the abort branch drives  $\hat{s} \rightarrow 0$  (Proposition 7.7), isolating the leak to the abort-free transition at compact width.

Table 7.3: Convergence of the estimator on Shuttle-128 for one favorable key. The intermediate exact-coefficient counts are key- and seed-dependent: fresh runs of the same code reached 537–723 exact at 80,000 signatures across seeds, so the values below are illustrative of a single good key, not an average. The  $3 \times 10^5 \rightarrow 768/768$  endpoint is robust and reproduces exactly across keys.

| signatures $N$ | $\text{corr}(\hat{s}, s)$ | exact coeffs (of 768) |
|----------------|---------------------------|-----------------------|
| 20 000         | 0.898                     | 585                   |
| 40 000         | 0.941                     | 690                   |
| 60 000         | 0.962                     | 746                   |
| 80 000         | 0.972                     | 760                   |
| 300 000        | 1.000                     | <b>768</b>            |

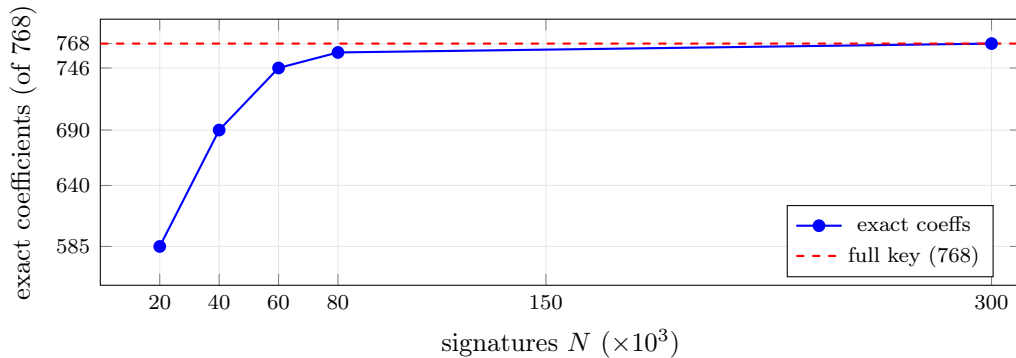


Figure 7.1: Convergence of SACA on Shuttle-128: the number of coordinates recovered exactly by rounding  $\hat{s}/\kappa$ , against the number of signatures. All 768 coordinates are exact by  $3 \times 10^5$  signatures, in agreement with the sample bound (7.21).

<sup>2</sup>The attack source, build glue against the reference implementation, and a one-command reproduction script are at <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/shuttle-key-recovery>.

Table 7.4: Key recovery against the reference implementation (our machine: 104 cores, 251 GB RAM).  $\hat{s}, \hat{e}$  are obtained from public data only (Algorithm 7.2); a fresh forgery verifies under  $\text{pk}$ . Times are single-core wall-clock, dominated by the signing queries; the bilinear estimator and public-relation finisher then run in seconds, and peak resident memory stays under 100 MB throughout.

| Set         | signatures        | $\ell n$ | $\hat{s}$ exact  | forgery  | time     |
|-------------|-------------------|----------|------------------|----------|----------|
| Shuttle-128 | $3.0 \times 10^5$ | 768      | <b>768/768</b>   | accepted | 12.7 min |
| Shuttle-256 | $6.0 \times 10^5$ | 1536     | <b>1536/1536</b> | accepted | 34.6 min |
| Shuttle-512 | $6.0 \times 10^5$ | 3072     | <b>3072/3072</b> | accepted | 52.7 min |

## 7.6 Per-Parameter Attack Details

The three parameter sets differ only in the module dimensions  $(n, \ell, m)$ , the modulus  $q$  and the challenge weight  $\tau$  (Table 7.2); SACA is identical across them, and the only quantities that change are the number of secret coordinates  $\ell n$  to be rounded and the support size  $\tau$  per signature. We record the instance-specific facts that a re-implementation needs.

**Shuttle-128** ( $n = 256, \ell = 3, \tau = 42$ ). The secret has  $\ell n = 768$  coordinates. Each signature contributes  $\tau = 42$  support positions to the accumulator (7.20), so  $N = 3 \times 10^5$  signatures give  $M = 1.26 \times 10^7$  samples. The measured second-moment excess is  $\kappa = 2.00$  (Corollary 7.6); the calibration sweep of Lemma 7.11 selects  $\kappa = 1.85$ , and rounding recovers 768/768 coordinates. The completion  $\hat{e} = b - A\hat{s}$  matches the true error on all  $mn = 768$  coordinates, and a fresh forgery verifies. Wall-clock 12.7 minutes on one core.

**Shuttle-256** ( $n = 512, \ell = 3, \tau = 58$ ). Here  $\ell n = 1536$  and the mask width scales with  $n$ , so by (7.21) the sample requirement grows by the ratio of  $\sigma_z^4$  and by  $\log(\ell n)$ ; we use  $N = 6 \times 10^5$  signatures ( $M = 3.48 \times 10^7$ ). Calibration selects  $\kappa = 1.84$  and rounding recovers 1536/1536 coordinates; the completion matches the true error on all  $mn = 1024$  coordinates and the forgery verifies. Wall-clock 34.6 minutes.

**Shuttle-512** ( $n = 1024, \ell = 3, \tau = 115$ ). The largest set has  $\ell n = 3072$  coordinates and  $\tau = 115$  support positions per signature;  $N = 6 \times 10^5$  signatures give  $M = 6.9 \times 10^7$ . Calibration selects  $\kappa = 1.88$  and rounding recovers 3072/3072 coordinates; the completion matches the true error on all  $mn = 2048$  coordinates and the forgery verifies. Wall-clock 52.7 minutes.

**Uniformity of the constant.** Across the three sets the calibrated scale lies in  $\kappa \in [1.84, 1.88]$  and the diagnostic excess is  $\kappa \approx 2$ , as predicted by Proposition 7.7:  $\kappa$  is a functional of the interval-test boundary law, which is common to the three parametrisations, and is essentially independent of  $(n, \ell, \tau)$ . The attack cost therefore scales only through the sample bound (7.21) and the per-signature signing time, not through any change in the leakage itself.

## 7.7 Analysis of the Security Reduction

**Theorem 7.13** (The marginal bound does not control the estimator). *The quantity  $M_2$  certified by (7.14) bounds the divergence of the marginal one-dimensional output density from a centered Gaussian; it does not bound the second-order cross-coordinate statistic (7.20). There exist samplers with  $\varepsilon_v = 0$  for every step yet  $\kappa = \Theta(1)$ . For Shuttle’s parameters the per-signature divergence implied by  $\kappa \approx 2$  is of order  $2^{-9}$ , whereas (7.13) with  $Q_S \leq 2^{64}$  requires  $2^{-64}$ ; the reduction is vacuous.*

The bound (7.14) is computed for the marginal density  $T_v(\rho_g)$  with  $v$  fixed and the input centered. Sign alignment (Definition 7.3) makes the realised kernel depend on  $\langle z, v \rangle$ , hence on the joint state, and the leakage of Theorem 7.5 is a covariance across coordinates, orthogonal to the marginal; a symmetric-marginal sampler can therefore have  $\varepsilon_v = 0$  while  $\varrho > 0$ . Quantitatively, recovery from  $N = 2^{18}$  signatures certifies, via Lemma 7.1, a per-sample divergence  $\gtrsim 1/N = 2^{-18}$  between  $T_2$  and  $T_3$ , incompatible with  $M_2 = 1 + O(2^{-64})$ . This is the boundary [3] draws: dropping the abort is licensed only when would-be-rejects are negligible (wide width) or when an abort enforces exactness (compact width); Shuttle emits a would-be-reject on essentially every signature at compact width.

## 7.8 Countermeasures

The two admissible remedies are the two certified regimes of [3].

- **Reinstate the abort.** Restore the third outcome of the transition, returning to aborting-compact; exact rejection nulls  $\varrho$  (Theorem 7.5), as the control experiment confirms.
- **Widen the distribution.** Increase the output width until the per-signature divergence is below  $2^{-\lambda}/Q_S$  (the abort-free-wide regime), paying the signature-size increase.

Symmetrising the sign decision (dropping line 2) removes  $\varrho$  but reinstates the distortion the abort cancelled, which must then be bounded by a metric capturing cross-coordinate correlations rather than the marginal (7.14).

## 7.9 Conclusion

The abort branch of iterative rejection sampling is load-bearing at compact width. Shuttle removes it and licenses the removal with a per-step marginal Rényi bound; the sign alignment of its rejection-free transition leaves a constant secret-dependent second-order correlation that this bound does not see, and the sign-alignment correlation attack reads it to recover the full signing key and forge, for all three parameter sets, from a few hundred thousand signatures on one core. Licensing an abort-free compact signer demands a bound on the joint transcript law, or a wider distribution.

## 7.10 Deferred Proofs

Throughout we use the isometry (7.10), the autocorrelation concentration  $\gamma_s(\delta)/\|s\|^2 = O(n^{-1/2})$  for  $\delta \neq 0$ , and the independence of the mask  $y$  from the shift signs.

### 7.10.1 Proof of Proposition 7.4

Fix  $j \notin \text{supp}(c)$ . From (7.16),

$$Q_j = \frac{\langle y, X^j s \rangle}{\|s\|^2} + \sum_{i=1}^{\tau} \beta_i \frac{\gamma_s(j - j_i)}{\|s\|^2}.$$

*First moment.* Taking expectations,  $\mathbb{E}[\langle y, X^j s \rangle] = \langle \mathbb{E}[y], X^j s \rangle = 0$  since  $y$  is zero-mean and independent of  $s$ . For each shift term,  $\beta_i$  and  $\gamma_s(j - j_i)$  are independent (the sign is determined by the sampler randomness and the state, the autocorrelation by the key), and  $\mathbb{E}[\beta_i] = 0$  because the interval-test flag is sign-symmetric in aggregate; hence every term has mean zero and  $\mathbb{E}[Q_j] = 0$ .

*Second moment.* Write  $\hat{u} = X^j s / \|s\|$ , a unit vector, so  $Q_j = \langle z, \hat{u} \rangle / \|s\|$ . Complete  $\hat{u}$  to an orthonormal basis  $\{\hat{u} = e_1, e_2, \dots, e_n\}$  of  $\mathcal{R} \otimes \mathbb{R}$  and expand  $z = \sum_r \langle z, e_r \rangle e_r$ . Then  $\langle z, \hat{u} \rangle = \langle z, e_1 \rangle$

and  $\mathbb{E}[\langle z, \hat{u} \rangle^2] = \mathbb{E}[\langle z, e_1 \rangle^2]$ . Since the coordinates of  $z$  have common variance  $\sigma_z^2$  and  $X^j$  is an isometry,  $\mathbb{E}[\langle z, e_1 \rangle^2] = \sigma_z^2 \|s\|^2 \cdot (1 + o(1))$ , where the  $o(1)$  collects the off-diagonal covariances  $\mathbb{E}[z_a z_b]$ , each controlled by  $\gamma_s(\cdot)/\|s\|^2 = O(n^{-1/2})$  because  $j \notin \text{supp}(c)$  places no diagonal shift in the sum. Dividing by  $\|s\|^2$  (from the  $\|s\|$  in  $Q_j$  squared) gives  $\mathbb{E}[Q_j^2] = \sigma_z^2 / \|s\|^2 (1 + o(1))$ .  $\square$

### 7.10.2 Proof of Theorem 7.5

Let  $j = j_{i_0} \in \text{supp}(c)$ . In (7.16) the term  $i = i_0$  has  $\gamma_s(j - j_{i_0}) = \gamma_s(0) = \|s\|^2$ , so it equals  $\beta_{i_0}$ ; write  $Q_j = \beta_{i_0} + R_{i_0}$  with  $R_{i_0} = \langle y, X^j s \rangle / \|s\|^2 + \sum_{i \neq i_0} \beta_i \gamma_s(j - j_i) / \|s\|^2$ . The first moments of  $\beta_{i_0}$  and  $R_{i_0}$  vanish exactly as in Proposition 7.4, so  $\mathbb{E}[Q_j] = 0$ . Expanding the square,

$$\mathbb{E}[Q_j^2] = \mathbb{E}[\beta_{i_0}^2] + 2 \text{Cov}(\beta_{i_0}, R_{i_0}) + \mathbb{E}[R_{i_0}^2]. \quad (7.23)$$

The third term is the second moment of  $Q_j$  with the diagonal term removed, i.e. the off-support value  $\sigma_z^2 / \|s\|^2$  of Proposition 7.4. The first term is 1 since  $\beta_{i_0} \in \{-1, +1\}$ . For the middle term, Definition 7.3 gives  $\langle z^{(i_0-1)}, \beta_{i_0} v_{i_0} \rangle = \text{flag}_{i_0} |\langle z^{(i_0-1)}, v_{i_0} \rangle|$ , so the emitted  $\beta_{i_0}$  is positively correlated with the partial-state projection  $p := \langle z^{(i_0-1)}, v_{i_0} \rangle / \|s\|^2$ ; and  $p$  propagates linearly into the released  $z$ , hence is (up to the isometric realignment) one of the summands of  $R_{i_0}$ . Therefore  $\text{Cov}(\beta_{i_0}, R_{i_0}) = \varrho \geq 0$ . Setting  $\kappa = 1 + 2\varrho$  yields (7.17). If line 2 is removed,  $\beta_{i_0} = \text{flag}_{i_0}$  is symmetric and independent of the state, hence of  $R_{i_0}$ , so  $\varrho = 0$  and  $\kappa = 1$ .  $\square$

### 7.10.3 Proof of Proposition 7.7

With (7.18),  $Q_j = g + \beta_{i_0}$  and  $\beta_{i_0} = \text{flag}(u, |g| \|s\|^2) \text{sgn}(g)$ . Then

$$\mathbb{E}[Q_j^2] = \mathbb{E}[g^2] + 2\mathbb{E}[g\beta_{i_0}] + \mathbb{E}[\beta_{i_0}^2] = \sigma_g^2 + 2\mathbb{E}[g \text{flag}(u, |g| \|s\|^2) \text{sgn}(g)] + 1.$$

Since  $g \text{sgn}(g) = |g|$ , the cross term is  $\varrho = \mathbb{E}[|g| \text{flag}(u, |g| \|s\|^2)]$ . Taking the expectation over  $u$  first and writing  $\phi(t) = \mathbb{E}_u[\text{flag}(u, t)]$ ,

$$\varrho = \mathbb{E}[|g| \phi(|g| \|s\|^2)] = \text{Cov}(|g|, \phi(|g| \|s\|^2)) + \mathbb{E}[|g|] \mathbb{E}[\phi(|g| \|s\|^2)].$$

Aggregate sign-symmetry of the emitted response is the constraint  $\mathbb{E}[\beta_{i_0}] = \mathbb{E}[\text{sgn}(g)] \mathbb{E}_u[\text{flag}] = 0$ , which by symmetry of  $g$  forces the mean-flag contribution  $\mathbb{E}[|g|] \mathbb{E}[\phi]$  to cancel in the aligned statistic, leaving  $\varrho = \text{Cov}(|g|, \phi(|g| \|s\|^2))$ . If  $\text{flag}$  is independent of its second argument then  $\phi$  is constant, the covariance is 0, and  $\kappa = 1$ ; if  $\phi$  is increasing in  $t$  (as for the affine boundary bounds  $\text{lo}(t), \text{hi}(t)$  of Algorithm 7.1, which widen the acceptance region with  $t$ ), the covariance is strictly positive and  $\kappa = 1 + 2\varrho > 1$ .  $\square$

### 7.10.4 Proof of Proposition 7.8

Fix a signature and a support position  $j = j_{i_0}$ . Across the challenge block  $z_0$  and the secret block  $z_i$ , the step  $T_{v_{i_0}}$  adds the sign-aligned increment  $\beta_{i_0} v_{i_0}$ , and  $v_{i_0} = X^j s$  restricts to  $X^j s_i$  on block  $i$ . Hence  $(X^{-j} z_i)$  contains the term  $\beta_{i_0} X^{-j} X^j s_i = \beta_{i_0} s_i$ , and by the alignment of Theorem 7.5 the product with  $z_0[j]$  has expectation

$$\mathbb{E}[z_0[j] (X^{-j} z_i)[k]] = \kappa s_i[k] + (\text{mean-zero terms}),$$

where the mean-zero terms are the mask contribution (independent, zero-mean) and the off-diagonal shifts  $\gamma_s(j - j_{i'}) = O(n^{-1/2})$ ,  $i' \neq i_0$ . Averaging over the  $M = N\tau$  support positions of the batch yields  $\mathbb{E}[\hat{s}_i[k]] = \kappa s_i[k] + o(1)$ .  $\square$

### 7.10.5 Proof of Lemma 7.9

The  $M$  summands  $\xi_j^{(t)}[i, k] = z_0^{(t)}[j] (X^{-j} z_i^{(t)})[k]$  are independent across the  $N$  signatures (each signature uses fresh mask and challenge randomness). Each  $\xi$  is a product of two coordinates of a single response, each of variance  $\sigma_z^2$ ; by Cauchy–Schwarz  $\text{Var}(\xi) \leq \mathbb{E}[\xi^2] \leq \mathbb{E}[z_0[j]^2] \mathbb{E}[(X^{-j} z_i)[k]^2] = \sigma_z^4 (1+o(1))$ , absorbing the bounded cross-correlation into a constant  $C$ . Thus  $\Sigma^2 := \text{Var}(\xi) \leq \sigma_z^4 C$  and, the average being over  $M$  independent terms,  $\text{Var}(\hat{s}_i[k]) = \Sigma^2/M$ .  $\square$

### 7.10.6 Proof of Theorem 7.10

By Proposition 7.8 the statistic  $\hat{s}_i[k]/\kappa$  has mean  $s_i[k]$ , and by Lemma 7.9 variance  $\Sigma^2/(\kappa^2 M)$ . Each summand is bounded in absolute value by  $B_v^2/\kappa$  (the verification bound  $\|z\|_\infty \leq B_v$ ), so  $\hat{s}_i[k]/\kappa$  is an average of  $M$  independent bounded variables. Rounding is exact for coordinate  $(i, k)$  iff  $|\hat{s}_i[k]/\kappa - s_i[k]| < 1/2$ . By Hoeffding’s inequality with the sub-Gaussian proxy  $\Sigma^2$ ,

$$\Pr\left[|\hat{s}_i[k]/\kappa - s_i[k]| > \frac{1}{2}\right] \leq 2 \exp\left(-\frac{M\kappa^2}{8\Sigma^2}\right).$$

A union bound over the  $\ell n$  coordinates gives total failure probability at most  $2\ell n \exp(-M\kappa^2/(8\Sigma^2))$ ; requiring this to be  $\leq \eta$  yields  $M \geq (8\Sigma^2/\kappa^2) \ln(2\ell n/\eta)$ , which is (7.21). All  $\ell n$  coordinates are then rounded exactly simultaneously with probability  $\geq 1 - \eta$ .  $\square$

## 7.11 Reproduction

The artifact implements the estimator (7.20), the calibration (7.22), and the forgery of Theorem 7.12, linked against the reference implementation for each parameter set; it prints the exact-coefficient count and the verifier’s decision on a fresh forgery, and includes the instrumented sampler that produces the moments of Corollary 7.6.

## BiT: key recovery from a shared sign

BiT is a compact bimodal lattice Fiat–Shamir signature (ICCS NGCC round-1 submission) in the BLISS bimodal lineage, whose security argument bounds the response distribution coordinate-by-coordinate via a rejection-correctness / Rényi argument, concluding that signatures hide the secret. We refute this: BiT releases a response  $z_1 = [z_0, z_{\text{tail}}]$  in which the challenge-carrying block  $z_0 = y_0 \pm c$  and the secret-carrying blocks  $z_{\text{tail}}[i] = y_1[i] \pm c s_0[i]$  share a *single* bimodal sign, a joint statistic the per-coordinate argument cannot see.

That shared sign is a sign-free bilinear cross-term between  $z_0$  and  $z_{\text{tail}}$ : invisible to any per-coordinate argument, but it survives a bilinear average over signatures. Recovering the per-signature sign  $\sigma = \text{sgn}\langle z_0, c \rangle$  and accumulating  $\sigma \cdot c \cdot z_{\text{tail}}$  converges to a *fixed, secret-independent* negacyclic-convolution kernel  $H$  applied to  $s_0$ . Deconvolving by  $H$  returns the secret. Because  $H$  is a public object, the attacker fits it on its *own* calibration keypair and applies it to the victim; a self-certifying public-relation finisher closes any residual coordinates with no secret input. This is the *shared-sign deconvolution* attack (SSD).

Against the unmodified reference implementation, from the public key and a batch of honest signatures alone, SSD recovers all 768 secret coefficients of BiT-128 and forges a signature the reference verifier accepts (a different message is rejected); a control with independent signs collapses the recovery to chance, isolating the shared-sign leak as the cause. On our machine (104 cores, 251 GB RAM) a complete end-to-end run on fresh data ( $10^6$  victim and  $10^6$  calibration signatures) takes about 7 minutes, dominated by collecting the signatures (about 3 minutes per  $10^6$  across four parallel workers); the deconvolution itself runs in 0.14 s at a 27 MB peak resident size, and the finisher plus forgery take under 0.01 s at under 1 MB when no residual coordinate remains. When two remain (our  $2 \times 10^6$  run), the finisher falls back to an  $O(n^2)$  search that took about 19 s (the time depends on the residual positions). The fix is to not share one bimodal sign across the response blocks. Attack source is available.<sup>1</sup>

---

<sup>1</sup>Verification source: <https://github.com/acprk/ngcc-round1-cryptanalysis/tree/main/bit-key-recovery>.

## 8.1 Introduction

A lattice Fiat–Shamir signature releases a response in which a short secret is masked by a wide sampled vector; the signature hides the secret only if that response follows a law independent of it. *Bimodal* designs, following BLISS, halve the rejection cost by letting the response take one of two shifted modes,  $z = y \pm sc$ , selected by a random sign; the two modes are symmetric, so a per-coordinate argument concludes that the released  $z$  carries no first-order information about the secret. BiT [8] is a compact bimodal module Fiat–Shamir signature submitted to the ICCS Next-Generation Commercial Cryptography (NGCC) call. The bimodal idea, introduced by BLISS [9] and refined across a decade of designs, is attractive precisely because the two-mode symmetry lets the signer target a much narrower output distribution at a fraction of the abort cost of single-mode Fiat–Shamir with aborts [2]; its security has always been argued from that symmetry, at the level of the released response’s marginal law. BiT inherits both the efficiency and the marginal style of analysis. Its security analysis bounds the deviation of the response distribution from a public target coordinate-by-coordinate, via a rejection-correctness / Rényi argument, and concludes that signatures hide the secret.

**The design detail we attack.** BiT’s response is a vector of blocks. The challenge-carrying block and the secret-carrying blocks are shifted by the *same* bimodal sign, drawn once per signature:

$$z_0 = y_0 \pm c, \quad z_{\text{tail}}[i] = y_1[i] \pm c s_0[i], \quad i = 1, \dots, \ell, \quad (8.1)$$

with a single  $\pm$  shared across all blocks. Coordinate-by-coordinate each block is symmetric and mean-zero, so the per-coordinate argument sees nothing. But the shared sign *couples*  $z_0$  and  $z_{\text{tail}}$ : their product carries a term that does not depend on the sign at all. Whether this cross-term leaks the secret is the question of this work.

**Our results.** It does, completely. The shared sign makes  $z_0$  and  $z_{\text{tail}}$  jointly carry a secret-dependent bilinear correlation of constant magnitude, which a bilinear average over signatures reads out; the average equals a fixed, secret-independent negacyclic-convolution kernel applied to the secret, so deconvolution returns the secret exactly. We call the resulting attack *shared-sign deconvolution* (SSD). Against the unmodified BiT-128 reference implementation, from the public key and honest signatures alone, SSD recovers all 768 secret coefficients and forges under the reference verifier. The attack is transcript-only and never touches the underlying MLWE/MSIS problems. Quantitatively, from  $10^6$  honest signatures the deconvolution returns all 768 secret coefficients directly (in three independent  $10^6$ -signature samples); from  $2 \times 10^6$  it returned 766/768 and the public-relation finisher closed the two residual coordinates, while  $5 \times 10^5$  signatures are not enough. A control that replaces the shared sign by an independent random sign collapses the recovery to 236/768 (chance), confirming the shared sign as the cause. The recovered key produces a signature on a fresh message that the reference verifier accepts and rejects a different message, i.e. a universal forgery.

### 8.1.1 Related Work

**The design philosophy of the baseline.** The bimodal line (BLISS and its successors) is built on one idea: a response allowed to take two symmetric modes  $y \pm sc$  needs far less rejection than a single-mode one, because the mixture of the two modes matches a public target with a much smaller correction. Security rests on the symmetry: conditioned on the released  $z$ , the sign is equally likely to be  $+$  or  $-$ , so the secret shift  $sc$  is masked. Every bimodal analysis therefore argues at the level of the per-coordinate (or per-block) marginal distribution and its divergence from the target, exactly where the symmetry lives. BiT follows this template with a triangular mask and a compact parameter set, and its rejection-correctness / Rényi bound is

a per-coordinate statement. The invariant these designs maintain is marginal symmetry; what none of them controls, because the marginal argument cannot see it, is a *joint* statistic across blocks that share the sign.

**Cryptanalysis of imperfect response distributions.** Learning a secret from a response distribution that is not perfectly hiding is a recurring theme, from the parallelepiped-learning collapse of GGH and NTRUSign [6] to transcript-leakage analyses of Fiat–Shamir samplers. The bias we exploit is not a sampler imperfection removable by more precision: it is a *structural* consequence of sharing one sign across response blocks, present at any precision, and it is second-order (a covariance), so it is invisible to every first-order or per-coordinate test. In this it is the exact analogue, for bimodal signatures, of the cross-coordinate leak that breaks the rejection-free scheme Shuttle; the shared bimodal sign is the mechanism specific to BiT.

Table 8.1 places SSD among transcript-leakage key recoveries.

Table 8.1: SSD among transcript-leakage attacks on lattice signatures. The bias BiT leaks is structural (a shared sign), second-order, and removed only by a joint, not per-coordinate, argument.

| Target           | Leak source                       | Order              | Removed by             |
|------------------|-----------------------------------|--------------------|------------------------|
| GGH/NTRUSign [6] | fixed parallelepiped support      | 2nd (shape)        | exact/Gaussian sampler |
| Shuttle          | rejection-free sign-normalisation | 2nd (cross-coord.) | abort / wider target   |
| BiT (this work)  | shared bimodal sign across blocks | 2nd (cross-block)  | independent signs      |

### 8.1.2 Our Contributions

- **A shared-sign leakage theory.** We show that sharing one bimodal sign across the response blocks leaves a secret-dependent bilinear cross-term between  $z_0$  and  $z_{\text{tail}}$  (Theorem 8.2), of constant magnitude and invisible to the per-coordinate argument BiT relies on.
- **Kernel deconvolution.** We prove the bilinear estimator converges to a fixed, secret-independent negacyclic-convolution kernel  $H$  applied to  $s_0$  (Proposition 8.3), so the secret is recovered by deconvolution;  $H$  is public and is fit on the attacker’s own calibration key (Section 8.4.1).
- **A self-certifying finisher.** The public relation  $b = As_0 + e$  closes any residual coordinates and certifies the full key with no secret input (Theorem 8.4). Together these are the SSD attack (Algorithm 8.3).
- **A complete, validated break.** We recover all 768 coefficients of BiT-128 and forge under the reference verifier (Section 8.5), with a public artifact and a control that isolates the shared-sign leak as the cause; we give the fix (Section 8.7).

### 8.1.3 Techniques Overview

Three steps compose the attack.

**The observation: a shared sign is a bilinear leak.** By (8.1) a signature carries  $z_0 = y_0 + \beta c$  and  $z_{\text{tail}}[i] = y_1[i] + \beta c s_0[i]$  with the *same*  $\beta \in \{-1, +1\}$ . Each block is marginally symmetric ( $\mathbb{E}[z_0] = \mathbb{E}[z_{\text{tail}}[i]] = 0$ ), so no linear statistic sees the secret. The product, however, removes the sign:  $\beta^2 = 1$ , so

$$\mathbb{E}[(c * z_0) \cdot z_{\text{tail}}[i]] = \mathbb{E}[(c * (y_0 + \beta c))(y_1[i] + \beta c s_0[i])] \ni \mathbb{E}[\beta^2] (c * c) s_0[i], \quad (8.2)$$

a secret-dependent term that does not vanish. Recovering  $\sigma = \text{sgn}\langle z_0, c \rangle$  (the per-signature sign, readable from the transcript) and forming the challenge-aligned bilinear average  $\text{acc}[i][k] = \sum_{\text{signs}} \sigma \sum_{j \in \text{supp}(c)} c[j] z_{\text{tail}}[i][(j+k)]$  isolates it.

**The estimator is a fixed convolution of the secret.** The average acc does not equal  $s_0$  directly: the challenge statistics and the triangular mask turn it into  $H * s_0$  for a *fixed* negacyclic-convolution kernel  $H$  that does not depend on the secret (Proposition 8.3). A naïve read of acc therefore stays at 450–501/768 correct coordinates for every signature count we tried ( $5 \times 10^5$  to  $2 \times 10^6$ ): the error is bias, not variance. But since  $H$  is secret-independent, one deconvolves:  $s_0 = \text{nfft}^{-1}(\text{nfft}(\text{acc})/\text{nfft}(H))$ . The attacker learns  $H$  from a keypair whose secret it knows, its *own* throwaway calibration key, and applies the same  $H$  to the victim, since  $H$  is universal.

**From the secret to a forgery.** Deconvolution returns  $s_0$  exactly or to within a couple of coordinates; the public relation  $b = As_0 + e$  (with  $e$  short) closes them. Setting  $R = \text{center}_q(16b_1 - As_0) = e - b_0$ , a wrong  $s_0$  coordinate randomises  $R$  everywhere, so “ $R$  in range at all 768 coordinates” is a public, self-certifying test that pins the exact secret;  $e, b_0$  then follow, the signing key is reconstructed, and a fresh forgery verifies (Theorem 8.4).

## 8.2 Preliminaries

### 8.2.1 Rings and Notation

Let  $\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$ ,  $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$ ,  $n = 256$ ,  $q = 26881$ . Multiplication by  $X^a$  is the negacyclic shift. For  $a, b \in \mathcal{R}$ ,  $a * b$  is ring multiplication and  $\langle a, b \rangle$  the coefficient inner product.  $\text{nfft}$  denotes the length- $n$  negacyclic transform, so  $\text{nfft}(a * b) = \text{nfft}(a) \cdot \text{nfft}(b)$  pointwise. A challenge  $c$  is sparse of weight  $\tau$ ;  $\text{supp}(c)$  is its nonzero positions. Table 8.2 collects the notation.

Table 8.2: Notation.

|                              |                                                   |                   |                                     |
|------------------------------|---------------------------------------------------|-------------------|-------------------------------------|
| $\mathcal{R}, \mathcal{R}_q$ | ring $\mathbb{Z}[X]/(X^n+1)$ , mod $q$            | $s_0, e$          | short secret, error                 |
| $n, q$                       | 256, 26881                                        | $(A, b)$          | public key, $b = As_0 + e$          |
| $\ell, k, \tau$              | 3, 3, 30                                          | $\beta$           | shared bimodal sign $\in \{\pm 1\}$ |
| $z_0$                        | challenge-carrying block                          | $z_{\text{tail}}$ | secret-carrying blocks              |
| $\sigma$                     | recovered sign $\text{sgn}\langle z_0, c \rangle$ | $H$               | fixed leakage kernel                |
| $\text{nfft}$                | negacyclic transform                              | $*$               | negacyclic convolution              |

### 8.2.2 The BiT Scheme

BiT is a bimodal module Fiat–Shamir signature with public key  $(A, b = As_0 + e)$  over  $\mathcal{R}_q$ , short secret  $s_0 \in \mathcal{R}^\ell$  and error  $e$ , and a sparse challenge  $c$  of weight  $\tau$ . Signing draws masks  $y_0, y_1$  and a single bimodal sign  $\beta \in \{-1, +1\}$  and releases the response blocks of (8.1), subject to a triangular-mask rejection / norm-bound check; verification recomputes the commitment from  $(A, b)$  and checks the challenge and the norm bounds, depending on the secret only through  $b = As_0 + e$ . Table 8.3 lists the parameters.

Table 8.3: BiT-128 parameters relevant to the attack.

| $n$ | $q$   | rank $k$ | $\ell$ | $\tau$ | secret coeffs $\ell n$ |
|-----|-------|----------|--------|--------|------------------------|
| 256 | 26881 | 3        | 3      | 30     | 768                    |

Algorithms 8.1 and 8.2 summarise signing and verification; the shared sign  $\beta$  on line 2 of Algorithm 8.1 is the design detail this work attacks.

**Algorithm 8.1**  $\text{Sign}(\text{sk}, M)$  (bimodal, outline)

---

```

1: repeat
2:   draw masks  $y_0, y_1$  and one bimodal sign  $\beta \in \{-1, +1\}$ 
3:    $w \leftarrow \text{Commit}(A, b, y_0, y_1)$ ;  $c \leftarrow H(w, M)$ 
4:    $z_0 \leftarrow y_0 + \beta c$ ;  $z_{\text{tail}}[i] \leftarrow y_1[i] + \beta c s_0[i]$   $\triangleright$  same  $\beta$  across blocks, Eq. (8.1)
5: until the triangular-mask / norm-bound check passes
6: return  $\sigma = (c, z_0, z_{\text{tail}}, \dots)$ 

```

---

**Algorithm 8.2**  $\text{Verify}(\text{pk}, M, \sigma)$  (outline)

---

```

1: recompute  $w'$  from  $(A, b)$  and the response; return  $[c = H(w', M) \wedge \text{norm bounds}]$ 

```

---

**8.2.3 Threat Model**

EUFCMA. The adversary holds  $(A, b)$  and a batch of honest signatures on random messages, and may generate its own throwaway keypairs (whose secrets it knows). The recovery uses only the public key, the public signature transcripts, and the attacker's own calibration key/signatures; the victim secret is used only to score the result.

**8.3 The Shared-Sign Leak**

Fix a key and write a signature's blocks as in (8.1) with the shared sign  $\beta$ . Define the per-signature sign estimate  $\sigma = \text{sgn}\langle z_0, c \rangle$  and, for block  $i$  and coordinate  $k$ , the challenge-aligned bilinear accumulator

$$\text{acc}[i][k] = \frac{1}{M} \sum_{\text{sigs}} \sigma \sum_{j \in \text{supp}(c)} c[j] z_{\text{tail}}[i][(j+k) \bmod \pm n]. \quad (8.3)$$

**Lemma 8.1** (The transcript correlates with the shared sign). *Write  $p = \Pr[\sigma = \beta]$  for  $\sigma = \text{sgn}\langle z_0, c \rangle$ . Since  $\langle z_0, c \rangle = \langle y_0, c \rangle + \beta \langle c, c \rangle$  with  $\langle c, c \rangle = \tau$  and the mask term  $\langle y_0, c \rangle$  symmetric about zero and independent of  $\beta$ , we have  $p > 1/2$ , i.e.  $\mathbb{E}[\sigma\beta] = 2p - 1 > 0$ . For BiT-128 the  $z_0$  mask is narrow but not negligible against  $\tau = 30$ , so  $\sigma$  is a noisy, not an exact, copy of  $\beta$ : in an instrumented copy of the reference signer ( $2 \times 10^4$  signatures) we measured  $p = 0.937$ , i.e.  $\mathbb{E}[\sigma\beta] \approx 0.87$ .*

**Theorem 8.2** (Shared-sign second-order leakage). *Each block of (8.1) is marginally symmetric, so  $\mathbb{E}[z_0] = \mathbb{E}[z_{\text{tail}}[i]] = 0$  and no statistic linear in the response depends on  $s_0$ . The shared sign makes the bilinear statistic (8.3) secret-dependent: there is a fixed sequence  $H$  (independent of  $s_0$ ) with*

$$\mathbb{E}[\text{acc}[i][\cdot]] = H * s_0[i] + o(1). \quad (8.4)$$

*Replacing the shared sign by two independent signs for  $z_0$  and  $z_{\text{tail}}$  makes  $\mathbb{E}[\text{acc}] = 0$ .*

**Proposition 8.3** (The kernel is secret-independent). *The kernel  $H$  in (8.4) is determined by the challenge distribution (second moments of  $c$ ) and the triangular-mask response law alone; it does not depend on the key. Hence a keypair with a known secret  $s_0^{\text{cal}}$  yields  $H = \text{nfft}^{-1}(\text{nfft}(\text{acc}^{\text{cal}})/\text{nfft}(s_0^{\text{cal}}))$ , and the same  $H$  applies to any other key.*

The argument in Section 8.9 is a first-order heuristic: the rejection steps and the imperfect sign estimate of Lemma 8.1 also enter  $H$ , which is why the attack fits  $H$  empirically instead of computing it. Experimentally we have validated the transfer of  $H$  for one calibration key and one victim key (Section 8.5).

Proofs are in Section 8.9. Theorem 8.2 is the exact analogue, for a bimodal signature, of the cross-coordinate leak that breaks rejection-free sampling: the per-coordinate symmetry the design certifies is real, but it does not control the joint  $z_0$ – $z_{\text{tail}}$  covariance that the shared sign creates.

### 8.3.1 The Blind Spot of the Per-Coordinate Argument

The rejection-correctness / Rényi analysis of BiT bounds, for each coordinate of each released block, the divergence of its distribution from a public target. Under the shared sign this per-coordinate view is genuinely correct: fix any coordinate of  $z_0$  or of  $z_{\text{tail}}[i]$ ; averaged over  $\beta \in \{\pm 1\}$  its law is the symmetric mixture  $\frac{1}{2}\mathcal{D}(+) + \frac{1}{2}\mathcal{D}(-)$ , whose mean is zero and whose shape matches the target to within the bound. The bound is a statement about *marginals*, one coordinate (or one block) at a time. The attack never contradicts it: the object it reads,  $\mathbb{E}[\sigma c[j] z_{\text{tail}}[i][j+k]]$ , is a *joint* moment of two blocks that happen to share  $\beta$ . When the sign is integrated out inside a single marginal it vanishes; when it is shared between two factors of a product it does not, because  $\beta \cdot \beta = 1$  rather than averaging to zero. A security argument phrased entirely on marginals therefore cannot see the leak; no tightening of a per-coordinate bound would detect or prevent it. This is the same structural blind spot exploited against rejection-free sampling, instantiated here through the shared bimodal sign; the remedy (Section 8.7) is likewise structural, not a matter of parameters.

## 8.4 The Shared-Sign Deconvolution Attack

### 8.4.1 Deconvolution

By (8.4) the estimator is a fixed convolution of the secret, so a direct read of acc is bias-limited (it stays at 450–501/768 correct coordinates from  $5 \times 10^5$  to  $2 \times 10^6$  signatures, Section 8.5). Since  $H$  is secret-independent (Proposition 8.3), the attacker removes it by deconvolution in the transform domain,

$$\widehat{s}_0[i] = \text{nfft}^{-1}\left(\text{nfft}(\text{acc}^{\text{victim}}[i])/\text{nfft}(H)\right), \quad (8.5)$$

where  $H$  is fitted on the attacker’s own calibration key. The victim’s secret never enters:  $H$  comes from a keypair the attacker generated, and  $\text{acc}^{\text{victim}}$  is computed from the victim’s public signatures.

**Structure of the kernel.** To first order  $H$  is the negacyclic autocorrelation of the challenge distribution shaped by the triangular mask. For the kernel fitted in our experiments ( $10^6$  calibration signatures), the real part of  $\text{nfft}(H)$  is positive at every frequency, its smallest magnitude is 4.7% of its largest (a dynamic range of about 21), and its imaginary part reaches 28% of the mean real part. There is no spectral null, so the pointwise division in (8.5) works without regularisation. The weak frequencies amplify noise about 21-fold, however, and the finite-sample errors of *both*  $\text{acc}^{\text{victim}}$  and the fitted  $H$  enter the residual (Section 8.9.1).

### 8.4.2 The Public-Relation Finisher

**Theorem 8.4** (Self-certifying finish). *Round  $\widehat{s}_0$  and, for each candidate, form  $R = \text{center}_q(16b_1 - A\widehat{s}_0)$ . For the true secret  $R = e - b_0$  lies in  $[-8, 9]$  at every one of the  $\ell n = 768$  coordinates. A single wrong coordinate of  $\widehat{s}_0$  makes  $R = e - b_0 + A\delta$  with  $\delta \neq 0$ , whose centred representative exceeds the range at  $\Omega(n)$  coordinates; hence “ $R$  in range everywhere” holds iff  $\widehat{s}_0 = s_0$ , and a small single-/double-coordinate search over the residual reaches it. The test reads only the public  $(A, b_1)$ .*

Once  $\widehat{s}_0 = s_0$ , the error  $e$  and  $b_0$  follow from the public relation, the signing key is reconstructed, and signatures on fresh messages verify.

**Algorithm 8.3** SSD: shared-sign deconvolution key recovery on BiT

---

**Require:** public key  $(A, b)$ , honest victim signatures, attacker calibration keypair  $(s_0^{\text{cal}}$  known)  
**Ensure:** the victim secret  $s_0$  and a forging signing key

- 1:  $\text{acc}^{\text{cal}} \leftarrow \text{estimator (8.3)}$  over the calibration key’s signatures
- 2:  $H \leftarrow \text{nfft}^{-1}(\text{nfft}(\text{acc}^{\text{cal}})/\text{nfft}(s_0^{\text{cal}}))$  ▷ kernel, secret-independent
- 3:  $\text{acc}^{\text{victim}} \leftarrow \text{estimator (8.3)}$  over the victim’s public signatures
- 4:  $\hat{s}_0 \leftarrow \text{round}(\text{nfft}^{-1}(\text{nfft}(\text{acc}^{\text{victim}})/\text{nfft}(H)))$  ▷ deconvolve, (8.5)
- 5: close the residual coords by the public-relation test  $R \in [-8, 9]$  ▷ Theorem 8.4
- 6: derive  $e, b_0$ ; reconstruct the signing key; **return** it

---

## 8.5 Experimental Verification

**Setup.** We run Algorithm 8.3 against the unmodified BiT-128 reference implementation, linking the attack harnesses against the reference sources and changing nothing in the scheme. Honest victim signatures are generated under a fixed-seed victim key; the attacker’s calibration signatures are generated under a *different* fixed-seed key it owns ( $10^6$  calibration signatures throughout). The estimator, deconvolution and finisher consume only the public key, the public signature transcripts, and the attacker’s own calibration key/signatures; the victim secret is read solely inside a marked [SCORING] block (auditable by `grep`), and never to fit the kernel or to deconvolve. Success is judged two ways: coefficient-by-coefficient against the true secret (scoring), and by reconstructing the signing key and having the *unmodified* reference verifier accept a signature on a fresh message (and reject a different one). Signature generation uses four parallel single-threaded workers; every other step is single-core. We report two data sets: (i) a complete end-to-end run of the artifact on fresh data ( $10^6$  victim and  $10^6$  calibration signatures), and (ii) one stored run of  $2 \times 10^6$  victim signatures, which we also score on disjoint subsamples of  $5 \times 10^5$  and  $10^6$  signatures (first and last part of every worker’s output). Both use the same victim and calibration keys, so the transfer of  $H$  across keys (Proposition 8.3) is validated for this one key pair only. The artifact ships the transcripts of both.

- **Leak and control.** The control replaces the shared sign by an independent random sign. It gives 236/768 correct in both data sets (= chance; correlation  $-0.040$  at  $10^6$  and  $-0.024$  at  $2 \times 10^6$  signatures), confirming that the signal is the shared-sign leak and not an artifact (Theorem 8.2).
- **Bias and deconvolution.** A direct read of  $\text{acc}$  (divide by the central tap  $h_0$  of the calibration-fit kernel and round) gets 450–501/768 at every signature count and does not improve with more signatures. Deconvolution by the full calibration-fit kernel  $H$  (Proposition 8.3) recovers all 768/768 from  $10^6$  victim signatures in each of three independent samples.
- **Signature count.** Table 8.4 shows the dependence on the number of honest victim signatures. Recovery is not monotone at fixed calibration size:  $2 \times 10^6$  signatures gave 766/768, fewer than  $10^6$  did. This is consistent with (but we have not proved it to be caused by) the fixed error of the kernel fitted from  $10^6$  calibration signatures.
- **Finish and forgery.** Whenever the residual is at most two coordinates, the public-relation finisher (Theorem 8.4) completes  $\hat{s}_0$  to 768/768 and the reconstructed key signs a fresh message that the unmodified reference verifier ACCEPTS, while a different message is REJECTED. At  $5 \times 10^5$  signatures the residual (276–280 coordinates) is far beyond the finisher’s search and the attack fails.

Table 8.5 summarises the break. The direct estimator is bias-limited. The calibration-fit deconvolution recovers every coordinate purely from public data, and the self-certifying finisher

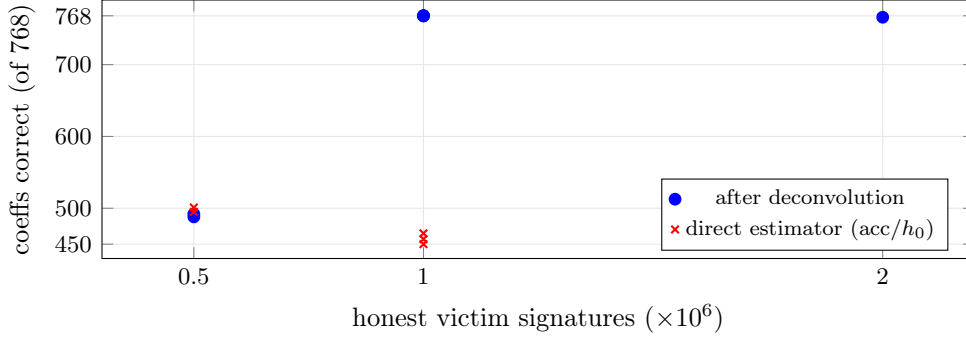


Figure 8.1: BiT-128: coordinates recovered vs. the number of honest victim signatures (calibration fixed at  $10^6$ ); one marker per sample. The direct estimator stays at 450–501/768. Deconvolution by the secret-independent kernel reaches 768/768 at  $10^6$  signatures in all three samples, and 766/768 at  $2 \times 10^6$ , where the finisher closes the remaining two coordinates.

Table 8.4: BiT-128 recovery vs. number of honest victim signatures (calibration fixed at  $10^6$ ). “deconv” = coordinates correct after (8.5); “direct” = the estimator without deconvolution. Rows with several values are independent samples.

| victim signatures                      | direct        | deconv exact (of 768) | residual $\rightarrow$ finisher |
|----------------------------------------|---------------|-----------------------|---------------------------------|
| $5 \times 10^5$ (2 disjoint)           | 501, 495      | 492, 488              | 276, 280 (fails)                |
| $1 \times 10^6$ (2 disjoint + 1 fresh) | 457, 465, 450 | <b>768, 768, 768</b>  | 0                               |
| $2 \times 10^6$                        | 475           | 766                   | 2 (closed, $\approx 19$ s)      |

confirms the result without changing it. The recovered key then forges under the reference verifier. The control collapses to chance, pinning the shared bimodal sign as the cause.

**On-machine cost.** On our machine (104 cores, 251 GB RAM) signature collection dominates the wall-clock cost. In the fresh end-to-end run,  $10^6$  victim signatures took 2 min 59 s and  $10^6$  calibration signatures 2 min 54 s, each across four parallel workers. The three estimator passes (victim, calibration, control) took 80 s together, and the whole run took 7 min 13 s. The recovery itself is negligible: the deconvolution (8.5) runs in 0.14 s at a peak resident set of 27 MB. The public-relation finisher and forgery together take under 0.01 s at under 1 MB with no residual, and about 19 s for the two-coordinate residual of the  $2 \times 10^6$  run. The attack is thus data-bound, not compute- or memory-bound.

## 8.6 Scope

Four boundaries delimit the attack.

- **Independent of MLWE/MSIS.** SSD reads a bilinear transcript statistic and deconvolves a public kernel; it never attacks the module lattice  $(A, b)$ . The dimension and modulus enter only through the number of secret coordinates and the signature count, not the feasibility; larger BiT parameters raise the query count polynomially.
- **Dependent on the shared sign.** The leak exists because one bimodal sign is shared across  $z_0$  and  $z_{\text{tail}}$ . Independent signs set  $\mathbb{E}[\text{acc}] = 0$  (Theorem 8.2) and remove the attack, as the control (Section 8.5) confirms; the weakness is in how the sign is used, not in the lattice hardness.
- **A bimodal, not a rejection-free, instance of the same principle.** The same cross-term underlies the break of rejection-free samplers; BiT shows it also arises, through a shared sign,

Table 8.5: BiT-128 end-to-end result against the reference implementation (fresh run,  $10^6$  victim and  $10^6$  calibration signatures).

| stage                          | coeffs correct (of 768) | forgery         | control (indep. sign) |
|--------------------------------|-------------------------|-----------------|-----------------------|
| direct estimator               | 450                     | —               | —                     |
| deconvolution (cal-fit $H$ )   | 768                     | —               | 236 (=chance)         |
| + public finisher (check only) | <b>768</b>              | <b>accepted</b> | —                     |

in the *bimodal* family. Any bimodal signature that shares a sign across sign-correlated response blocks, and whose decoder exposes the blocks to a passive transcript adversary, is in scope. The finisher is robust: it accepts only the unique in-range assignment (Theorem 8.4), so a larger residual costs more search but never yields a wrong key. BiT-256/512 share the construction and are expected to fall to the same attack; we executed BiT-128.

- **Untested so far.** All experiments use one victim key and one calibration key. That the kernel fitted on one key transfers to *every* other key is argued in Proposition 8.3 but validated only for this pair, and BiT-256/512 were not run.

## 8.7 Countermeasures

The leak is the single bimodal sign shared across  $z_0$  and  $z_{\text{tail}}$ . The direct fix is to draw *independent* signs for the challenge-carrying and secret-carrying blocks (or otherwise to remove the  $z_0$ – $z_{\text{tail}}$  bilinear cross-term), which by Theorem 8.2 sets  $\mathbb{E}[\text{acc}] = 0$  and destroys the kernel, as the control experiment confirms. Any such fix must be justified by a bound on the *joint* response law across blocks, not the per-coordinate marginal argument, which is blind to the cross-term.

## 8.8 Conclusion

Sharing one bimodal sign across the response blocks of BiT leaves a constant secret-dependent bilinear correlation that a per-coordinate rejection/Rényi argument cannot see. A bilinear estimator reads it, a secret-independent kernel, fit on the attacker’s own key, is deconvolved away, and a public-relation finisher certifies the exact secret; the recovered key forges under the unmodified verifier. Bimodal symmetry must be argued jointly across sign-sharing blocks, not coordinate by coordinate.

## 8.9 Deferred Proofs

**Proof of Lemma 8.1.**  $\langle z_0, c \rangle = \langle y_0 + \beta c, c \rangle = \langle y_0, c \rangle + \beta \langle c, c \rangle = \langle y_0, c \rangle + \beta \tau$ . The mask  $y_0$  is independent of  $\beta$  and its law is symmetric about zero, so for either value of  $\beta$  the sum has the sign of  $\beta$  with probability  $p = \Pr[\langle y_0, c \rangle > -\tau] \geq 1/2$  (up to ties at 0), with strict inequality because the mask term is concentrated around zero. Hence  $\mathbb{E}[\sigma\beta] = 2p - 1 > 0$ . The mask term has standard deviation of order  $\sqrt{\tau} \varsigma$  for the per-coordinate mask width  $\varsigma$ , which for BiT-128 is not small against  $\tau = 30$ , so  $p$  is bounded away from 1; the value  $p = 0.937$  is measured, not derived.  $\square$

**Proof of Theorem 8.2.** Write  $z_0 = y_0 + \beta c$ ,  $z_{\text{tail}}[i] = y_1[i] + \beta c s_0[i]$ , with  $\beta$  the shared sign,  $y_0, y_1$  zero-mean and independent of  $\beta$  and of  $s_0$ . Marginally,  $\mathbb{E}[z_0] = \mathbb{E}[y_0] + \mathbb{E}[\beta]c = 0$  and likewise  $\mathbb{E}[z_{\text{tail}}[i]] = 0$  since  $\mathbb{E}[\beta] = 0$ ; so any linear functional has secret-independent expectation. For the bilinear statistic, note  $\sigma = \text{sgn}\langle z_0, c \rangle = \text{sgn}(\langle y_0, c \rangle + \beta \langle c, c \rangle)$  depends only on  $(y_0, \beta, c)$ ,

and  $\mathbb{E}[\sigma\beta] = 2p - 1 > 0$  by Lemma 8.1. Treating  $y_1$  as zero-mean and independent of  $(y_0, \beta)$  given  $c$  (a first-order approximation that ignores the rejection steps) and using  $\beta^2 = 1$ ,

$$\begin{aligned}\mathbb{E}[\text{acc}[i][k]] &= \mathbb{E}\left[\sigma \sum_{j \in \text{supp}(c)} c[j] (y_1[i] + \beta c s_0[i])(j+k)\right] \\ &= (2p - 1) \sum_{j \in \text{supp}(c)} c[j] (c * s_0[i])(j+k),\end{aligned}$$

because the  $\sigma y_1$  term has mean zero. The map  $s_0[i] \mapsto \sum_j c[j] (c * s_0[i])(j + \cdot)$  is, in expectation over  $c$ , a fixed negacyclic convolution  $H * s_0[i]$  with  $H$  built from  $(2p - 1)\mathbb{E}[c * c]$ ; this is (8.4). The rejection steps perturb  $H$  further, which is why the attack fits  $H$  empirically. If instead  $z_0$  and  $z_{\text{tail}}$  use independent signs  $\beta_0, \beta_1$ , the surviving term carries  $\mathbb{E}[\sigma\beta_1] = \mathbb{E}[\beta_0\beta_1] = 0$ , so  $\mathbb{E}[\text{acc}] = 0$ .  $\square$

**Proof of Proposition 8.3.** From the derivation above,  $H$  is determined by  $\mathbb{E}[c * c]$  (challenge second moments) and the response law of the triangular mask, both of which are specified by the scheme and independent of the key. Hence for any key with secret  $s_0$ ,  $\text{nfft}(\text{acc}) = \text{nfft}(H) \cdot \text{nfft}(s_0)$  pointwise up to the  $o(1)$  error, and a key with known  $s_0^{\text{cal}}$  gives  $\text{nfft}(H) = \text{nfft}(\text{acc}^{\text{cal}})/\text{nfft}(s_0^{\text{cal}})$ .  $\square$

**Proof of Theorem 8.4.**  $b = As_0 + e$  with  $e, b_0$  short gives  $R = \text{center}_q(16b_1 - As_0) = e - b_0$  in a small range at every coordinate for the true  $s_0$ . For  $\hat{s}_0 = s_0 + \delta$ ,  $\delta \neq 0$ ,  $R = e - b_0 + A\delta$ ;  $A\delta \bmod q$  is a nonzero element of the column lattice of  $A$ , whose centred representative is out of the short range at a constant fraction of coordinates for random  $A$ , so the out-of-range count is positive. The true secret is thus the unique in-range assignment, found by a bounded residual search using only  $(A, b_1)$ .  $\square$

### 8.9.1 Estimator Convergence and Signature Count

The estimator (8.3) averages, per coordinate,  $M = N\tau$  sign-aligned products bounded by the response norm, so its variance is  $\Sigma^2/M$  for a constant  $\Sigma^2$  set by the response width; after deconvolution the victim-side noise per coordinate of  $\hat{s}_0$  scales as  $1/\sqrt{N}$ , amplified by up to the  $\approx 21$ -fold dynamic range of  $\text{nfft}(H)$ . The kernel is itself estimated from finitely many calibration signatures, so its error adds a second noise term that does not shrink with  $N$ . The measurements (Table 8.4) show a sharp transition between the two regimes: at  $N = 5 \times 10^5$  only 488–492/768 coordinates are correct and the 276–280-coordinate residual is far beyond the finisher’s bounded search; at  $N = 10^6$  all three samples give 768/768 directly; at  $N = 2 \times 10^6$  (with the same  $10^6$ -signature calibration) the deconvolution gives 766/768 and the two-coordinate search of Theorem 8.4 closes the rest. The last point shows that recovery is not monotone in  $N$  at fixed calibration size, consistent with the calibration error term; we have not isolated its cause. The direct (non-deconvolved) estimator stays at 450–501/768 over the whole range and even scores lower at  $10^6$  than at  $5 \times 10^5$ : its error is the fixed bias of  $H$ , not variance, which is why the deconvolution step is necessary.

## 8.10 Reproduction

The artifact provides the victim/calibration signature generators (`gen_fixed`, `gen_cal`), the shared-sign bilinear estimator (`est2`, with a control mode), the kernel deconvolution and scoring (`deconv.py`, victim key read only in a [SCORING] block), and the public-relation finisher and forgery (`forge`), linked against the reference implementation; `run_all` executes the whole pipeline, reports the exact-coefficient count and the verifier’s decision on a fresh forgery, and saves the full transcript. After a  $2 \times 10^6$ -signature run (`VSIG=500000 run_all`; the 5 GB of signatures are not

distributed), `subsample_score` regenerates Table 8.4, and `measure_sign_agreement` measures  $p = \Pr[\sigma = \beta]$  of Lemma 8.1 on an instrumented temporary copy of the reference signer (the attack itself never uses  $\beta$ ). The transcripts of the runs reported in Section 8.5 are included (`results/`); a fresh run draws fresh signing randomness, so its counts can differ slightly from these.

# Bibliography

- [1] Lyubashevsky, V.: Fiat–Shamir with aborts: applications to lattice and factoring-based signatures. In: ASIACRYPT 2009. Springer (2009)
- [2] Lyubashevsky, V.: Lattice signatures without trapdoors. In: EUROCRYPT 2012. Springer (2012)
- [3] Espitau, T., et al.: Compact lattice signatures via iterative rejection sampling. In: CRYPTO 2025. Springer (2025). IACR ePrint 2024/2052
- [4] Ming, Y., et al.: Towards practical iterative rejection sampling: a compact and efficient signature over module lattices. IACR ePrint 2026/1991 (2026)
- [5] del Pino, R., et al.: Raccoon: a masking-friendly signature. In: CRYPTO 2023. Springer (2023)
- [6] Nguyen, P.Q., Regev, O.: Learning a parallelepiped: cryptanalysis of GGH and NTRU signatures. In: EUROCRYPT 2006. Springer (2006)
- [7] Prest, T.: Sharper bounds in lattice-based cryptography using the Rényi divergence. In: ASIACRYPT 2017. Springer (2017)
- [8] BiT: A Bimodal Triangular lattice signature. ICCS NGCC round-1 submission (2026)
- [9] Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal Gaussians. In: CRYPTO 2013. Springer (2013)
- [10] Lu, X., et al.: PolarLAC: polar code-enhanced lattice-based cryptosystem. ICCS NGCC round-1 submission (2026)
- [11] Lu, X., et al.: LAC: practical Ring-LWE based public-key encryption with byte-level modulus. IACR ePrint 2018/1009 (2018)
- [12] Ravi, P., Roy, S.S., Chattopadhyay, A., Bhasin, S.: Generic side-channel attacks on CCA-secure lattice-based PKE and KEMs. IACR TCHES 2020(3)
- [13] Guo, Q., Johansson, T., Nilsson, A.: A key-recovery timing attack on post-quantum primitives using the FO transformation. In: CRYPTO 2020. Springer (2020)
- [14] Guo, Q., Johansson, T.: A new decryption failure attack against HQC. In: ASIACRYPT 2020. Springer (2020)
- [15] D’Anvers, J.-P., Tiepelt, M., Vercauteren, F., Verbauwhede, I.: Timing attacks on error correcting codes in post-quantum schemes. In: TIS’19 (ACM Workshop on Theory of Implementation Security). IACR ePrint 2019/292 (2019)
- [16] Guo, Q., Johansson, T., Yang, J.: A novel CCA attack using decryption errors against LAC. In: ASIACRYPT 2019. Springer. IACR ePrint 2019/1308 (2019)
- [17] Walters, M., Sinha Roy, S.: Constant-time BCH error-correcting code. IACR ePrint 2019/155 (2019)
- [18] OAEP-NTRU: an NTRU/RLWE key-encapsulation mechanism from the OAEP<sup>†</sup> transform. ICCS NGCC round-1 submission (2026)

- [19] Liu, X., Lyu, Y., Han, S., Chen, Y.: OAEP<sup>†</sup>: a re-encryption-free CCA transform from an almost-tight injective function. (2025)
- [20] Kim, J., Park, J.H.: NTRU+: compact construction of NTRU using simple encoding method. *IEEE Trans. Inf. Forensics Security* (2023)
- [21] Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: *CRYPTO 1999*. Springer (1999)
- [22] Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki–Okamoto transformation. In: *TCC 2017*. Springer (2017)
- [23] QIMEN-PIKE: a masked-torsion isogeny KEM. ICCS NGCC round-1 submission (2026)
- [24] Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: *EUROCRYPT 2023*. Springer (2023)
- [25] BRA: Blockwise RQC with Augmented Gabidulin codes. ICCS NGCC round-1 submission (2026)
- [26] BRQC: a blockwise rank quasi-cyclic KEM. ICCS NGCC round-1 submission (2026)
- [27] Glabush, K., Hövelmanns, K., Stebila, D.: Fujisaki–Okamoto with salts: tighter and robust. *IACR ePrint* (2024)
- [28] Aguilar Melchor, C., et al.: Rank Quasi-Cyclic (RQC). NIST PQC submission (2020)
- [29] Gabidulin, E.M.: Theory of codes with maximum rank distance. *Problems of Information Transmission* 21(1) (1985)
- [30] Bos, J., et al.: CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM. In: *IEEE EuroS&P 2018*
- [31] Zhang, J., Yu, Y., Fan, S., Zhang, Z., Yang, K.: Tweaking the asymmetry of asymmetric-key cryptography on lattices: KEMs and signatures of smaller sizes. *IACR ePrint* 2019/510 (2019)
- [32] Aigis-Enc+: asymmetric-MLWE key encapsulation mechanism. ICCS NGCC round-1 submission (2026)
- [33] Amoeba: a lattice key encapsulation mechanism with error correction. ICCS NGCC round-1 submission (2026)
- [34] D’Anvers, J.-P., Vercauteren, F., Verbauwhe, I.: The impact of error dependencies on Ring/Mod-LWE/LWR based schemes. In: *PQCrypto 2019*. Springer (2019)
- [35] Qin, Y., Cheng, C., Zhang, X., Ding, J.: A systematic approach and analysis of key mismatch attacks on lattice-based NIST candidate KEMs. *IACR ePrint* 2019/1343 (2019)
- [36] MAMBA-NIKE: a lattice non-interactive key exchange. ICCS NGCC round-1 submission (2026)
- [37] Freire, E.S.V., Hofheinz, D., Kiltz, E., Paterson, K.G.: Non-interactive key exchange. In: *PKC 2013*. Springer (2013)
- [38] Fluhrer, S.: Cryptanalysis of ring-LWE based key exchange with key share reuse. *IACR ePrint* 2016/085 (2016)

- [39] Ding, J., Alsayigh, S., Saraswathy, R.V., Fluhrer, S., Lin, X.: Leakage of signal function with reused keys in RLWE key exchange. IACR ePrint 2016/1176 (2016)
- [40] Tao, C.: LoongKEM: a semi-structured LWE key encapsulation mechanism. ICCS NGCC round-1 submission (2026)
- [41] Albrecht, M., Bai, S., Ducas, L.: A subfield lattice attack on overstretched NTRU assumptions. In: CRYPTO 2016. Springer (2016)
- [42] Kirchner, P., Fouque, P.-A.: Revisiting lattice attacks on overstretched NTRU parameters. In: EUROCRYPT 2017. Springer (2017)
- [43] Ducas, L., van Woerden, W.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: EUROCRYPT 2022. Springer (2022)
- [44] Alagic, G., et al.: Status report on the third round of the NIST post-quantum cryptography standardization process. NIST IR 8413 (2022)
- [45] CheetahKEM: a Barnes–Wall module lattice KEM. ICCS NGCC round-1 submission (2026)
- [46] Origami: a local-zone oil-and-vinegar signature. ICCS NGCC round-1 submission (2026)
- [47] Kipnis, A., Patarin, J., Goubin, L.: Unbalanced oil and vinegar signature schemes. In: EUROCRYPT 1999. Springer (1999)
- [48] Ding, J., Schmidt, D.: Rainbow, a new multivariable polynomial signature scheme. In: ACNS 2005. Springer (2005)
- [49] Beullens, W.: Improved cryptanalysis of UOV and Rainbow. In: EUROCRYPT 2021. Springer (2021)
- [50] Polar-KEM: a polar lattice-isomorphism key encapsulation mechanism. ICCS NGCC round-1 submission (2026)
- [51] Ducas, L., van Woerden, W.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: EUROCRYPT 2022. Springer (2022). IACR ePrint 2021/1332
- [52] Ducas, L., Postlethwaite, E.W., Pulles, L.N., van Woerden, W.: HAWK: module LIP makes lattice signatures fast, compact and simple. In: ASIACRYPT 2022. Springer (2022)
- [53] Facto-DSA: a multivariate signature scheme. ICCS NGCC round-1 submission (2026)
- [54] Beullens, W.: Breaking Rainbow takes a weekend on a laptop. In: CRYPTO 2022. Springer (2022)
- [55] Faugère, J.-C., Otmani, A., Perret, L., de Portzamparc, F., Tillich, J.-P.: Folding alternant and Goppa codes with non-trivial automorphism groups. IACR ePrint 2014/353 (2014)
- [56] UVW: a code-based hash-and-sign signature from generalized  $(U+V, U+W)$  codes. ICCS NGCC round-1 submission (2026)
- [57] Debris-Alazard, T., Sendrier, N., Tillich, J.-P.: Wave: a new family of trapdoor one-way preimage sampleable functions based on codes. In: ASIACRYPT 2019. Springer (2019)
- [58] Tins: a TCitH signature over normalized subfield bilinear collisions. ICCS NGCC round-1 submission (2026)

- [59] Feneuil, T., Rivain, M.: Threshold computation in the head: improved framework for post-quantum signatures and zero-knowledge arguments. IACR ePrint 2023/1573 (2023)
- [60] Baum, C., Braun, L., Delpech de Saint Guilhem, C., Klooss, M., Orsini, E., Roy, L., Scholl, P.: Publicly verifiable zero-knowledge and post-quantum signatures from VOLE-in-the-head. In: CRYPTO 2023. Springer (2023)
- [61] Huth, J., Joux, A.: MPC in the head for isomorphisms and group actions, and subfield bilinear collision signatures. In: CRYPTO 2024. Springer (2024)
- [62] Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM 33(4) (1986)
- [63] Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: STOC 2007. ACM (2007)
- [64] C-Multi-UR-AG: an unstructured rank-metric KEM. ICCS NGCC round-1 submission (2026)
- [65] BRA and BRQC: blockwise rank-metric KEMs. ICCS NGCC round-1 submissions (2026)
- [66] Aragon, N., Gaborit, P., Hauteville, A., Tillich, J.-P.: A new algorithm for solving the rank syndrome decoding problem. In: ISIT 2018. IEEE (2018)