CRYPTHARNESS: A HARNESS FOR AGENT-ASSISTED CRYPTANALYTIC RESEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Cryptanalysis typically requires both a mathematical model and code, making it a natural testbed for research agents. We present *CryptHarness*, a command-line agent for cryptanalytic research, built from reusable domain skills. Given only a researcher's intuition for how a cryptographic scheme might be attacked, CryptHarness develops the attack and produces a paper-style report. The report comes with three artifacts. First, a mathematical model of the underlying problem to support a theoretical analysis of the attack. Second, a proof of concept implementation designed for readability by abstracting away nonessential details. Third, an implementation of the scheme exactly according to the specification, and of the attack itself, allowing rigorous evaluation across experiments. As a case study, we apply CryptHarness to public-key decompression for ML-DSA, the new primary post-quantum signature standard. Given only the initial idea and minimal guidance, our agent reduced the required number of signatures of the state-of-the-art decompression method by a factor of three and provided a mathematical analysis. In a separate verification layer, we formally verify a central finite-size first-moment theorem in Lean 4, providing an (almost) matching lower bound on the number of signatures required to decompress an ML-DSA-44 public key. In a second case study, we apply CryptHarness to secret-key recovery from randomness leakage in Falcon, a secondary post-quantum signature standard. The agent identified three improvements to the state-of-the-art attack, reducing the number of signatures required for key recovery by factors of 2 and 3.7 for the two existing parameter sets, respectively.

1 INTRODUCTION

Cryptographic schemes are designed in a way that any successful attack requires solving well-defined mathematical problems. The task of cryptanalysis is to assess the security of these schemes by developing and evaluating methods for solving the underlying problems. Overestimating their hardness can lead to insecure cryptography, whereas underestimating it can result in unnecessarily inefficient protocols.

The resulting cryptanalytic attacks often rely on heuristic assumptions that must be tested experimentally through implementations. However, cryptanalysis is not merely a matter of implementing attacks. Any attack must be modeled precisely in mathematical terms for a thorough analysis of the correctness and runtime.

The recent CryptanalysisBench evaluates general-purpose coding agents on known attacks against cryptographic schemes (Fluri et al., 2026). However, using agents to assist in research, i.e., to find new attacks or optimize existing attacks, has been widely unexplored. Existing work on AI-assisted research ranges from practical guidance for mathematics and machine learning (Zimmer et al., 2026) to domain-specific systems for first-order optimization, machine-learning theory, and computational mathematics (Kim et al., 2026; Zhang et al., 2026; Chen et al., 2026). These developments raise the question of whether *a domain-specific harness can support both the computational and mathematical sides of cryptanalysis*.

Contributions. Our main contribution is CryptHarness, a skill-based command-line agent for cryptanalytic research. Given a rough written description of an attack on a cryptographic scheme, the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

agent evaluates its feasibility and produces a detailed report, including three artifacts: (1) a precise mathematical model of the attack; (2) a rigorous implementation, optimized for efficient experiments and reliable measurements; and (3) a readable, standalone proof of concept that allows the attack to be readily understood and executed. Reusable skills guide the agent in formalizing the task, developing and testing the attack, translating between the three artifacts, and determining what each artifact establishes.

As a case study and an independent cryptanalytic contribution, we present an improved public-key decompression method for the post-quantum signature standard ML-DSA. The method, developed primarily by CryptHarness, improves upon the state of the art and reduces the required number of signatures by a factor of three. We further provide a machine-checked Lean formalization of a central finite-size expectation theorem in the abstract mathematical model. Since public-key decompression is the bottleneck of several attacks on ML-DSA (Damm et al., 2026; Schubert et al., 2026; Bashiri et al., 2026; Wang et al., 2024; Paskarbeit et al., 2026), the improved decompression implies that these attacks can be mounted with fewer resources than previously known.

As a second case study and cryptanalytic contribution, we improve secret-key recovery from randomness leakage in Falcon. Building on the halfspace-learning attack of Brinkmann et al. (2026), CryptHarness developed three refinements: uncertainty-guided correction of rounded key coefficients, posterior-ranked candidate search, and a scale-anchored meet-in-the-middle search using Falcon's public-key relation. Under exact, noiseless leakage, the final attack recovered 12 of 12 Falcon-512 keys from 50 signatures and 12 of 12 Falcon-1024 keys from 30 signatures, improving on the state-of-the-art attack described in Brinkmann et al. (2026) by factors of 2 and 3.7, respectively.

2 Related work

AI agents for scientific research. The AI Scientist generates research ideas, implements and evaluates them, and produces manuscripts with automated reviews (Lu et al., 2024). Its successor introduces agentic tree search for experimentation and removes the dependence on human-authored code templates (Yamada et al., 2025). Agent Laboratory organizes literature review, experimentation, and report writing around a researcher-provided idea, with opportunities for human feedback throughout (Schmidgall et al., 2025). The Agentic Researcher provides methodological instructions for command-line coding agents working under researcher supervision (Zimmer et al., 2026). Several systems focus on mathematical research. AutoOPT combines numerical design of optimization methods, symbolic discovery, and Lean verification (Kim et al., 2026); VALG tracks theorem specifications and proof dependencies in machine-learning theory (Zhang et al., 2026); and Iteris develops numerical evidence, constructions, and proof drafts for computational mathematics, followed by expert review (Chen et al., 2026). Trace2Skill studies how lessons from execution trajectories can be distilled into reusable instructions (Ni et al., 2026). CryptHarness uses skills to guide attack development and evaluation, with checks on the attacker's permitted information and the implementation's conformance to the cryptographic scheme.

Large language models in cryptography. CryptoQA provides a large collection of questions from academic sources and evaluates factual knowledge, mathematical reasoning, and consistency (Elfares et al., 2025). CipherBank studies decryption tasks involving classical and custom ciphers (Li et al., 2025), while Maskey et al. (2026) evaluate decryption across ciphertext datasets under several prompting settings. AICrypto broadens evaluation to multiple-choice questions, capture-the-flag challenges, and proof problems, with expert baselines and proof-scoring rubrics (Wang et al., 2026). Muzsai et al. (2025) use procedurally generated cryptographic challenges to improve a Python-equipped agent through reinforcement learning. ShannonProver addresses formal proof engineering: given a cryptographer-supplied security model and a decomposition into lemma-level obligations, it constructs EasyCrypt proof scripts (Ma et al., 2026). CryptanalysisBench evaluates coding agents through executable security games and reports both known attacks and previously unknown findings (Fluri et al., 2026). Nasr & Carlini (2026) report an LLM-discovered improvement to an attack on seven-round AES that exploits the algebraic structure of its S-box, supported by mathematical analysis and computational checks. Our two case studies examine how an agent improves existing attacks under researcher supervision, and how the resulting code and mathematical claims are checked.

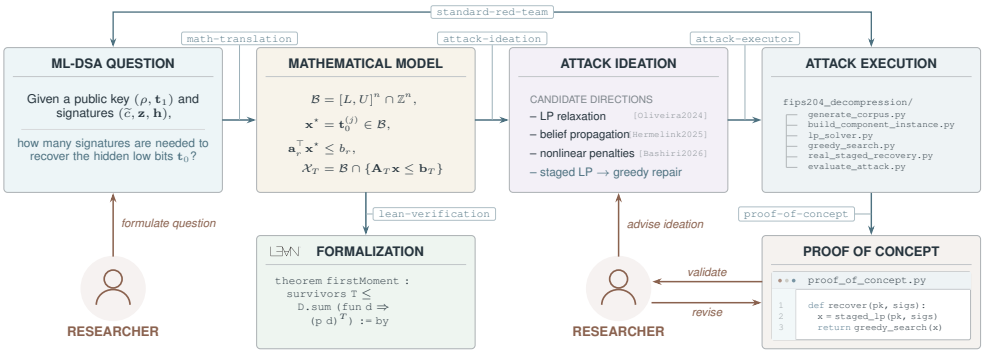

Figure 1: The two-stage workflow in the ML-DSA run. Mathematical formulation maps the cryptographic question to a constrained integer problem and enables Lean formalization of selected claims. Attack development proceeds from ideation to a standards-faithful implementation, which is then distilled into a readable proof of concept. Researchers supply the problem, advise the process, and validate the returned artifacts.

Earlier machine learning for cryptanalysis. Gohr (2019) developed neural distinguishers for round-reduced Speck and integrated them into key recovery. Chen et al. (2020) proposed a neural-aided statistical attack framework with complexity estimates, while Benamira et al. (2021) analyzed the information captured by Gohr's distinguisher and constructed an interpretable cryptanalytic counterpart with similar performance. For lattice problems, SALSA combines transformers and statistical techniques to recover sparse binary secrets in small-to-medium Learning with Errors instances (Wenger et al., 2023). SALSA VERDE extends this approach through improved preprocessing and recovery techniques to sparse binary, ternary, and narrow Gaussian secrets (Li et al., 2023). These methods use learned models during key recovery. CryptHarness uses a language model to develop the attack; the resulting algorithm runs without the agent. The cost of agent-assisted research is therefore separate from the signature and computational costs of running the attack. The specific predecessors of our ML-DSA and Falcon attacks are discussed in the corresponding case studies.

3 CryptHarness

CryptHarness is a framework for end-to-end, agent-assisted cryptanalysis under researcher supervision, spanning mathematical formulation, attack ideation, implementation, and analysis. Its skills and artifact requirements are harness- and model-agnostic in principle. Our implementation runs on Pi, a minimal terminal coding harness (Zechner et al., 2026), and gives the agent access to project files, a shell and Python. For our case study we used GPT-5.6 Sol at its high-reasoning setting and DeepSeek v4 Flash.

CryptHarness builds on *skills*: versioned instructions, optionally bundled with scripts and references, that are loaded into the agent's context on demand. Following Ni et al. (2026), we derive these skills from research traces and refine them with cryptanalysis researchers. Each skill addresses a specific research task. For example, attack ideation produces a candidate method and its assumptions, execution produces code and measurements, and conformance checking examines whether those measurements concern the intended scheme and recovery objective. The same candidate can pass through these tasks several times as it is revised. This division gives researchers intermediate results to inspect and a way to identify which part of an unsuccessful attempt needs further work.

Input. A run starts from a cryptanalytic question, its threat model, the scheme specification and implementation, and relevant literature. Researchers may also supply initial attack ideas and problem-specific software, such as SageMath for symbolic computation (The Sage Developers, 2022) or SCIP for linear and mixed-integer programming (Hojny et al., 2025). The question specifies the unknown quantity, the available observations, and the criterion for recovery. For ML-DSA, the target is the complete omitted public-key vector and the observations are public keys and signatures. For Falcon, the attack additionally receives the assumed randomness-sign leakage. Hidden key material is available to the evaluator for checking candidates, but is excluded from the attack inputs. This

separation determines what information an implementation may use during estimation, candidate selection, and validation.

Workflow. CryptHarness separates the research into mathematical formulation and attack development. The two stages can be run independently or iterated together. The mathematical model guides attack design, while implementations and experiments may reveal that the model must be revised.

Stage 1: Mathematical formulation. The `math-translation` skill identifies the unknowns and constraints and records how they correspond to objects and operations in the cryptographic scheme. This lets the agent use mathematical tools such as SageMath (The Sage Developers, 2022) and SCIP (Hojny et al., 2025). Selected claims can then be formalized through `lean-verification`, while experimental findings may lead to revisions of the model. The formulation exposes structure that an implementation can exploit. In ML-DSA, the unknowns are bounded integers and each signature supplies sparse linear inequalities. In Falcon, sign observations constrain short integer coefficients that also satisfy a public modular relation. Both formulations suggest using numerical methods to locate a candidate and discrete search to finish recovery. The mathematical description records the coefficient domains, arithmetic conventions, and observations so that a proposed optimization can be checked against the original cryptographic task.

Stage 2: Attack development. The `attack-ideation` skill develops candidate attacks with explicit assumptions, permitted information, and success conditions. The `attack-executor` implements selected candidates, reproduces the baseline, and evaluates both under comparable conditions. Separating ideation from execution helps prevent implementation failure from being mistaken for attack infeasibility, a failure mode observed by Fluri et al. (2026). The `standard-red-team` skill checks standards conformance and separates public attack inputs from secret evaluation data. Because faithful experimental code often spans several optimized files, `proof-of-concept` distills the attack into a standalone program for inspection rather than performance measurement. Reproducing the earlier method provides a common reference for comparison. Experimental feedback then distinguishes errors in the evaluation from shortcomings of the candidate method. In the ML-DSA investigation, a check found that the initial implementation recovered one component of the target vector. The evaluation was repeated for the complete vector before the improvement was reported. In the Falcon investigation, a rounded numerical estimate could fail while remaining close enough to the key for sparse coefficient corrections to succeed. These outcomes call for different next steps: correcting the recovery test in the first case and extending the recovery algorithm in the second.

Recovery checks. An optimization objective measures progress within an attack, while the recovery test determines whether the cryptanalytic task has been completed. A small residual or a stable rounded estimate alone does not establish success. For ML-DSA, the evaluation requires recovery of every component of the omitted vector and checks candidates against fresh signatures withheld from reconstruction. For Falcon, the evaluator compares the returned candidate with the source key; the recovered polynomials must also satisfy the public-key and NTRU relations and produce a fresh valid signature. These checks connect numerical output to the stated recovery objective. We report data requirements in signatures, preserving the distinction between one observed signature and the multiple constraints extracted from it.

Representations and validation. The workflow produces a mathematical model for analysis, a standards-faithful implementation for empirical claims, and a proof of concept for inspection. Executable checks test basic implementation properties, Lean kernel-checks selected formalized claims, and the proof of concept makes the attack easier to inspect. We stress that these checks do not replace human review of the threat model, the correspondence between the scheme and mathematical model, the code, and the resulting claims. The final report links each claim to its supporting artifact and records the relevant assumptions, checks, and limitations. The standalone ML-DSA demonstration illustrates this division. It retains signature generation, inequality extraction, staged localization, and integer repair, while omitting byte encodings, standard XOF samplers, and the NTT. A reader can follow the recovery procedure in one program; the reported signature counts come from the full experimental implementation. The two programs therefore serve different reproducibility needs: understanding the algorithm and measuring its behavior on the specified scheme.

Researcher guidance. Researchers contribute both the initial direction and judgments during development. In the ML-DSA case, they suggested exploiting the discreteness of the omitted coefficients; the agent developed the coordinate-wise tabu repair that made this suggestion effective. The skills specify research tasks without prescribing this particular algorithm. The case studies describe the supplied hypotheses, the methods developed during the runs, and the subsequent checks, so that the researchers' guidance and the agent's contributions can be assessed together.

4 Case Study: ML-DSA public-key decompression

The secret signing key of the post-quantum signature standard ML-DSA (National Institute of Standards and Technology, 2024) consists of the vectors $\mathbf{s}_1 \in \mathbb{Z}^{\ell n}$ and $\mathbf{s}_2 \in \mathbb{Z}^{kn}$. The corresponding uncompressed public key is given by

$$\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2 \in \mathbb{Z}_q^{kn}, \qquad \mathbf{A} \in \mathbb{Z}_q^{kn \times \ell n},$$

where arithmetic is performed modulo q. To reduce the public key size, ML-DSA publishes only a compressed version of $\mathbf{t}$. In particular, $\mathbf{t} = 2^d\mathbf{t}_1 + \mathbf{t}_0$, where $\mathbf{t}_1$ is published as part of the public key, while the low-order part $\mathbf{t}_0 = \left(\mathbf{t}_0^{(1)}, \ldots, \mathbf{t}_0^{(k)}\right)$ is omitted. Each unknown component $\mathbf{t}_0^{(i)}$, $1 \le i \le k$ consists of $n = 256$ integer coordinates in the range $\mathbf{t}_0^{(i)} \in \{-2^{d-1}+1, \ldots, 2^{d-1}\}^n$.

The compressed public key part suffices for signature verification. However, several side-channel attacks require the *decompressed* public-key part $\mathbf{t}_0$ to recover the secret key (Damm et al., 2025; 2026; Schubert et al., 2026; Bashiri et al., 2026; Wang et al., 2024; Paskarbeit et al., 2026). Once $\mathbf{t}_0$ is known, some of these attacks require fewer than 10 000 signatures for key recovery (Schubert et al., 2026; Bashiri et al., 2026). Recovering $\mathbf{t}_0$, however, is expensive. The state of the art method by Oliveira et al. (2024) requires roughly $180\,000$ signatures. Public-key decompression is therefore the main bottleneck in these attacks.

ML-DSA public-key decompression problem. Given the compressed public key $(\mathbf{A}, \mathbf{t}_1)$ and m signatures generated from the same secret key, recover the omitted vector $\mathbf{t}_0$. Recovery succeeds when the output $\widehat{\mathbf{t}}_0$ equals the key's true $\mathbf{t}_0$ in every coordinate. The uncompressed public vector is then $\mathbf{t} = 2^d\mathbf{t}_1 + \widehat{\mathbf{t}}_0 \pmod q$. We measure the cost of decompression by the number m of signatures needed for exact recovery.

Contributions of the case study. CryptHarness derived a first-moment bound with a threshold of roughly $57\,000$ signatures in an abstract reconstruction model, and improved the decompression algorithm to require approximately $60\,000$ signatures for ML-DSA-44. The small gap motivates studying whether the improved method is close to optimal. The bound concerns the expected number of alternative candidates; it does not establish non-uniqueness with high probability for actual signatures. The empirical improvement reduces the signature cost of the decompression step needed by the aforementioned side-channel attacks.

Previous work. Oliveira et al. (2024) observed that every valid signature provides public linear inequalities satisfied by the components of $\mathbf{t}_0$. Recovering $\mathbf{t}_0$ can therefore be formulated as an integer linear program. Their algorithm progressively narrows the search space. In each step, a small linear program is solved to obtain an estimate $\tilde{\mathbf{t}}_0$ that lies in a smaller box than in the previous iteration. Inequalities that do not cut this box are discarded. Once the estimate is sufficiently accurate, the final solution is rounded to an integer candidate for $\mathbf{t}_0$.

Expert hints as starting point. We provided CryptHarness with the observation that the existing decompression method models the signature leakage as linear inequalities, but does not fully exploit that $\mathbf{t}_0$ is discrete. We suggested that incorporating this discreteness could reduce the required number of signatures. As a possible direction, we suggested investigating belief propagation and nonlinear techniques that had performed well in similar settings (Hermelink et al., 2025; Bashiri et al., 2026). However, we did not specify how to adapt these techniques to public-key decompression or how to perform the final recovery.

Stage 1. Mathematical formulation. The agent invoked `math-translation` to record public-key decompression as bounded integer recovery from inequalities leaked by signatures. It also

translated the relevant arguments in the supplied papers into the same notation. We then prompted the agent to analyze the reconstruction threshold in this model. It derived dense first-moment bounds of 57,184 signatures for ML-DSA-44, 140,331 for ML-DSA-65, and 126,746 for ML-DSA-87, which appear in the lower-bound column below. The Lean formalization took approximately six hours and produced 2,718 lines across 14 Lean modules. The agent first produced a theorem blueprint and dependency graph, then formalized the definitions, intermediate lemmas, arithmetic certificates, and final theorem in Lean. This run used $162.53 of API credits, covered by the ChatGPT Pro subscription used for the experiment. The theorem is stated in the abstract model.

Stage 2: Attack development. The agent then spent approximately five hours exploring attack ideas and implementing candidates. Its final evaluation suggested a threefold reduction in the required number of signatures compared to the state of the art, but checks from `standard-red-team` showed that the implementation recovered only one of the four components of $\mathbf{t}_0$ for ML-DSA-44. After the implementation and evaluation were corrected to require all components of $\mathbf{t}_0$, the signature count remained nearly unchanged.

The newly developed method extends the algorithm of Oliveira et al. (2024) with a coordinate-wise tabu repair step (Glover, 1989). Starting from an integer estimate that may violate some inequalities, the algorithm selects the most violated inequality and considers the coordinates that occur in it. For each such coordinate, it tests a change of exactly $+1$ or -1 in the direction that reduces the violation. It applies the move that most reduces the total positive violation across all inequalities. A short tabu tenure prevents the same coordinate from being changed back immediately. The process continues until every retained inequality is satisfied. The repair step can start from a less accurate estimate of $\mathbf{t}_0$ than the rounding technique of Oliveira et al. (2024), reducing the number of inequalities and signatures needed.

Empirical result. Table 1 compares the signature counts for full recovery of $\mathbf{t}_0$. These numbers refer to recovery of the *entire* $\mathbf{t}_0$, not to individual components. The measurements use standards-conformant signatures, and recovered candidates are additionally tested for correctness against fresh signatures that were not used during reconstruction.

Table 1: Required number of signatures for public-key decompression. Previous signature counts are from Oliveira et al. (2024).

Parameters	Success rate	Mean signatures		Novel lower bound
		Previous	Ours	
ML-DSA-44	10/10	179 354	**60 112**	**57 184**
ML-DSA-65	10/10	523 534	**158 091**	**140 331**
ML-DSA-87	9/10	513 106	**137 465**	**126 746**

Verification of the results. The individual mathematical lemmas are formally verified in Lean 4. In addition, the proof and the implementations were reviewed by cryptanalysts, who identified no errors. As an additional validation step, the agent reimplemented the method of Oliveira et al. and successfully reproduced their results.

5 Case Study: Key Recovery in Falcon from Randomness Leakage

Falcon's secret key consists of four short polynomials $f, g, F, G \in R = \mathbb{Z}[X]/(X^n + 1)$, where $n \in \{512, 1024\}$ and $q = 12289$. They satisfy the NTRU equation $fG - gF = q$, and the public key is $h = gf^{-1} \pmod{(q, X^n + 1)}$. We identify each polynomial $a = \sum_{i=0}^{n-1} a_i X^i \in R$ with its coefficient vector $\mathbf{a} = (a_0, \ldots, a_{n-1})$. The secret key defines a basis

$$\mathbf{B} = \begin{pmatrix} C(g) & -C(f) \\ C(G) & -C(F) \end{pmatrix} \in \mathbb{Z}^{2n \times 2n}, \qquad C(a) = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ -a_{n-1} & a_0 & \cdots & a_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ -a_1 & -a_2 & \cdots & a_0 \end{pmatrix}.$$

To sign, Falcon hashes a message to a point $\boldsymbol{c} \in \mathbb{Z}_q^n$, samples an integer randomness vector $\boldsymbol{z} \in \mathbb{Z}^{2n}$, and uses the secret basis to produce a short signature vector $\boldsymbol{s} = (\boldsymbol{c}, \mathbf{0}^n) - \boldsymbol{z}\mathbf{B} \in \mathbb{Z}^{2n}$. Solving for the randomness gives

$$\mathbf{z} = ((\mathbf{c}, \mathbf{0}^n) - \mathbf{s})\,\mathbf{B}^{-1}, \qquad \text{where} \qquad \mathbf{B}^{-1} = \frac{1}{q}\begin{pmatrix} -C(F) & C(f) \\ -C(G) & C(g) \end{pmatrix}.$$

We consider the setting where a side channel reveals the signs of the n coordinates z_{n+i} for $0 \le i < n$. This setting arises from Hamming weight leakage during the signing as observed in Lin et al. (2025). For a signature-hash pair $(\mathbf{s}, \mathbf{c})$, set

$$\mathbf{w} = (\mathbf{c}, \mathbf{0}^n) - \mathbf{s} = (\mathbf{w}^{(1)}, \mathbf{w}^{(2)}).$$

Then $\mathbf{z} = \mathbf{w}\mathbf{B}^{-1}$. Let $\mathbf{e}_i$ be the i-th standard basis vector of $\mathbb{R}^n$. The known vectors

$$\boldsymbol{a}_i = \frac{1}{q} C(\mathbf{w}^{(1)})\mathbf{e}_i, \qquad \boldsymbol{b}_i = \frac{1}{q} C(\mathbf{w}^{(2)})\mathbf{e}_i \tag{1}$$

give, for $0 \le i < n$,

$$y_i = \operatorname{sign}(z_{n+i}) = \operatorname{sign}(\langle \boldsymbol{a}_i, \mathbf{f} \rangle + \langle \boldsymbol{b}_i, \mathbf{g} \rangle)\,.$$

We set $\operatorname{sign}(u) = +1$ for $u \ge 0$ and -1 otherwise. Thus, each sign leak y_i defines a linear inequality on the secret key parts f, g. Each signature supplies n such inequalities.

Falcon key recovery from sign leakage problem. Given m signature-hash pairs $(\mathbf{s}, \mathbf{c})$ generated under one public key h, together with the sign leaks z_{n+i} for $0 < i < n$, the task is to recover the complete private key (f, g, F, G). We measure key recovery cost by the required number m of signatures.

Previous work. The *Falcon key recovery from sign leakage problem* was first solved by Zhang et al. (2023), who used a method based on the spectral decomposition of the covariance of filtered signatures. Their attack was later improved by Lin et al. (2025), and recently Brinkmann et al. (2026) reduced the problem to the halfspace learning problem (Blum et al., 1998), which significantly reduced the required number of signatures.

In particular, Brinkmann et al. (2026) did not solve the problem for both $\mathbf{f}$ and $\mathbf{g}$, which is a learning problem in $2n$ dimensions. Instead, their attack reduces it to n dimensions by using the sign leakage to exclusively learn $\boldsymbol{f}$ while omitting the $\boldsymbol{g}$ contribution. This is effective because the coefficients of the hash point are typically much larger than those of the short signature vector. The omitted term nevertheless changes about 1% of the predicted signs, even when the side-channel labels themselves contain no errors. Brinkmann et al. (2026) therefore use logistic regression to tolerate these sign errors and estimate the direction of $\boldsymbol{f}$. After recovering the direction, they recover the scale using the known secret key distribution of Falcon and round to an integer candidate. They then derive $g = hf \pmod{(q, X^n + 1)}$ from the public key and solve the NTRU equation to obtain F and G. Their reported experiments achieved key recovery on average at 100 signatures for Falcon-512 and 110 for Falcon-1024.

Contributions of the case study. CryptHarness found three improvements to the state-of-the-art attack from Brinkmann et al. (2026): (1) it fits the exact sign equations in both $\boldsymbol{f}$ and $\boldsymbol{g}$, (2) uses uncertainty in the resulting estimate to select sparse integer corrections, and (3) filters those corrections through Falcon's public-key relation using a meet-in-the-middle search. The resulting attack recovers Falcon-512 keys on average from 50 signatures and Falcon-1024 keys from 30 signatures, reducing the required number of signatures by a factor of 2, respectively a factor of 3.7.

Expert hints as starting point. We provided the agent with the work of Brinkmann et al. (2026) and the fact that the Falcon secret key consists of short integers subject to public algebraic constraints, which was not exploited by Brinkmann et al. (2026). This suggested that exact key recovery might be possible even when the real-valued halfspace estimate did not round correctly. However, we did not specify the anchored full-sign fit, how to rank integer corrections, or how to join them efficiently. The agent developed those steps during the attack investigation.

Stage 1: Mathematical formulation. The unknowns are the short integer coefficient vectors of f and g. Each leaked sign gives a linear inequality in these $2n$ variables. The public relation $g = hf \pmod{(q, X^n + 1)}$ adds modular constraints. This formulation supports a real-valued estimate followed by discrete corrections checked against the public key.

Stage 2: Attack development. The agent spent approximately 12 hours exploring attack ideas and implementing candidates using the `attack-ideation` skill. Researchers guided the exploration and helped select directions to pursue. Tested candidates included alternative direction estimators, scale search, nonlinear integrality penalties, beam search, and sparse rounding repairs. The exploration led to the anchored joint fit and modular meet-in-the-middle search described below.

The agent first used the dimension-halved logistic estimate to obtain a scaled, rounded candidate $\boldsymbol{f}_0$. It then fitted the exact signs jointly in $\boldsymbol{f}$ and $\boldsymbol{g}$, penalizing distance from this candidate:

$$(\widetilde{\boldsymbol{f}}, \widetilde{\boldsymbol{g}}) = \underset{\boldsymbol{v},\boldsymbol{w}\in\mathbb{R}^n}{\arg\min} \frac{1}{mn} \sum_{j=1}^{m} \sum_{i=0}^{n-1} \log\big(1 + \exp\big[-y_{j,i}\big(\langle \boldsymbol{a}_{j,i}, \boldsymbol{v}\rangle + \langle \boldsymbol{b}_{j,i}, \boldsymbol{w}\rangle\big)\big]\big) + \frac{\lambda_f}{2}\|\boldsymbol{v} - \boldsymbol{f}_0\|_2^2 + \frac{\lambda_g}{2}\|\boldsymbol{w}\|_2^2.$$

The $\boldsymbol{g}$ term accounts for sign information lost in the earlier approximation. The anchor $\boldsymbol{f}_0$ keeps the fit near an integer scale, which signs alone cannot determine.

Second, the agent rounded $\widetilde{\boldsymbol{f}}$ to $\bar{\boldsymbol{f}}$ and ranked its coefficients by $|\widetilde{f}_i - \bar{f}_i|$. Large fractional distances mark coordinates where rounding is least certain. It searched sparse candidates

$$f_S = \bar{\boldsymbol{f}} + \sum_{i\in S} \operatorname{sign}(\widetilde{f}_i - \bar{f}_i) X^i,$$

so each selected coordinate moves by one integer step in the direction suggested by the fit.

Third, the agent used the public key to test these corrections before attempting full key reconstruction. Each candidate f_S determines a polynomial g_S such that

$$g_S = h f_S \pmod{(q,\, X^n + 1)},$$

with each coefficient of g_S chosen as its centered representative modulo q. The agent split the candidate corrections into two groups and matched their modular contributions using a small subset of coefficients of g_S. Only combinations that might yield a short g_S received a full public-constraint check. If the initial search found no key, the agent tried other balanced splits of the corrections and included less highly ranked uncertain coordinates that the initial search had omitted. For a surviving f_S, the agent completed (f_S, g_S, F, G) and required the resulting key to produce a fresh signature that verifies.

Attack description. For each $j \in \{1, \ldots, m\}$, the signature $(\boldsymbol{s}_j, \boldsymbol{c}_j)$ and the leaked sign vector $\operatorname{sign}(\boldsymbol{z}_j)$ yield n structurally dependent samples $(\boldsymbol{a}_{j,i}, \boldsymbol{b}_{j,i}, y_{j,i})$, for $i \in \{1, \ldots, n\}$. The vectors $\boldsymbol{a}_{j,i}$ and $\boldsymbol{b}_{j,i}$ are computed as in Equation (1), and

$$y_{j,i} = \operatorname{sign}(z_{j,n+i}) = \operatorname{sign}(\langle \boldsymbol{a}_{j,i}, \boldsymbol{f}\rangle + \langle \boldsymbol{b}_{j,i}, \boldsymbol{g}\rangle)\,.$$

We first omit the $\boldsymbol{g}$ term to estimate the direction of $\boldsymbol{f}$, then select an integer scale and round to obtain an anchor $\boldsymbol{f}_0$. An anchored fit of the full sign model gives a refined real-valued estimate $(\widetilde{\boldsymbol{f}}, \widetilde{\boldsymbol{g}})$. If rounding $\widetilde{\boldsymbol{f}}$ does not yield a valid key, we try sparse one-unit corrections, prioritizing coordinates nearest a rounding boundary. Each prescribed search phase splits its selected coordinates into two balanced sets. A meet-in-the-middle join uses a few coefficients of the public modular relation to discard unlikely pairs before full key reconstruction.

We define

$$\ell(\xi) := \log(1 + e^{-\xi}),$$

$$L_{\text{half}}(\boldsymbol{v}) := \frac{1}{mn} \sum_{j=1}^{m} \sum_{i=0}^{n-1} \ell\big(y_{j,i}\langle \boldsymbol{a}_{j,i}, \boldsymbol{v}\rangle\big)\,,$$

$$L_{\text{full}}(\boldsymbol{v}, \boldsymbol{w}; \boldsymbol{f}_0) := \frac{1}{mn} \sum_{j=1}^{m} \sum_{i=0}^{n-1} \ell\big(y_{j,i}\big(\langle \boldsymbol{a}_{j,i}, \boldsymbol{v}\rangle + \langle \boldsymbol{b}_{j,i}, \boldsymbol{w}\rangle\big)\big) + \frac{\lambda_f}{2}\|\boldsymbol{v} - \boldsymbol{f}_0\|_2^2 + \frac{\lambda_g}{2}\|\boldsymbol{w}\|_2^2.$$

We use $\lambda_f = 10^{-4}$ and $\lambda_g = 10^{-5}$ for both Falcon-512 and Falcon-1024. The complete attack algorithm is given in Algorithm 10 in Appendix E.

Empirical result. The agent evaluated the attack on 12 secret keys per Falcon parameter set. It increased the number of signatures m in steps of 10 until the key was successfully recovered. The sign leakage was noise-free, i.e., no artificial noise was added. For Falcon-512, we achieved key recovery from 50 signatures in 100% of cases, and in Falcon-1024 from 30 signatures in 100% of cases. Our results for Falcon-512 are shown in Table 2, compared to prior work.

Table 2: Key recovery from sign leakage for Falcon-512 (128-bit security, noise-free setting). We re-implemented the approach from Brinkmann et al. (2026) to compare its runtime with ours on the same hardware.

Authors	Signatures	Samples	Time	Success Rate
Zhang et al. (2023)	170k	170k	30 min	25 %
Lin et al. (2025)	25k	100k	–	25 %
Brinkmann et al. (2026)	100	51k	1 sec	100 %
our work	**50**	**26k**	2 sec	100 %

Verification of the results. The agent reproduced the earlier logistic recovery method on three Falcon-512 known-answer-test keys using 100 accepted signatures per key. For the new attack, an independent evaluator checked every returned candidate against the source key. Every recovered secret key was verified to produce fresh valid signatures.

Cryptanalysts independently reviewed the attack, its mathematical analysis, and the code.

6 Discussion and conclusion

The case studies show how a coding agent can support cryptanalytic research by helping produce an improved algorithm, a mathematical model, a readable proof of concept, and a Lean-checked first-moment lower bound theorem. It also shows the limits of this setup: the Lean development checks selected mathematical claims, not the full attack or implementation, and human review remains necessary.

We observed a safety issue relevant to research agents: direct cryptanalysis prompts triggered safety guardrails in GPT-5.6 Sol, while mathematically equivalent formulations often did not. This possible mismatch between cyber-safety guardrails and mathematically phrased security research suggests that safety evaluations should test both direct cyber prompts and equivalent mathematical reformulations.

7 Limitations

Our evaluation covers two lattice-signature problems. Applying CryptHarness to other cryptographic families may require different mathematical tools and scheme-specific checks, even when the overall workflow remains the same. The case studies also start from researcher-supplied questions, prior work, and initial attack ideas; choosing a useful research direction and assessing its significance remain expert responsibilities. Research trajectories can vary with the model version, prompts, and computational budget, so repeating a run need not produce the same sequence of ideas. The resulting attack code can be evaluated independently of that discovery process. Our experiments measure signature requirements under the stated observation models. In particular, the Falcon evaluation uses exact randomness-sign leakage; applying the method to physical traces would additionally require extracting those signs and accounting for measurement errors. Signature counts also capture only one part of attack cost: numerical optimization, candidate search, and data acquisition must be considered when assessing a concrete deployment.

AI USE STATEMENT

Generative AI was used to develop and refine attack ideas, formulate mathematical models and claims, assist with proof development and Lean formalization, implement and debug methods, design and run experiments, and interpret results. The research models and workflow are described in the CryptHarness section and the case studies. AI tools were also used to search and summarize literature, draft and revise manuscript text, and prepare the LaTeX source. Researchers supplied the research questions, prior work, and expert guidance, and reviewed the mathematical arguments and implementations as described in the case studies. The authors are responsible for the final text, claims, references, code, and experimental results, including contributions produced with AI assistance.

ETHICS STATEMENT

Cryptanalytic improvements can support security assessment and also reduce the cost of attacks. The reported results concern ML-DSA public-key decompression and Falcon key recovery under an explicit randomness-leakage model. Public-key decompression alone does not recover the ML-DSA signing key, and the Falcon results assume access to exact leaked signs. These conditions are essential to interpreting the security implications. Our observation that equivalent mathematical and cryptographic prompts can receive different safety responses also motivates evaluating research agents under both formulations.

REPRODUCIBILITY STATEMENT

The case studies describe the attack assumptions, parameter sets, recovery criteria, baseline comparisons, and validation procedures. Appendix B provides the detailed ML-DSA recovery algorithm, and Appendix C states the mathematical model and derivations. The Falcon case study gives the fitting objectives, regularization parameters, candidate-search procedure, and key-validation checks. The CryptHarness section identifies the research models and skills used. Reproducing the resulting attack experiments is distinct from reproducing the agent's sequence of research decisions.

REFERENCES

Kaveh Bashiri, Jan Geuenich, and Johannes Mittmann. Exploiting noisy single-bit leakage in ml-dsa. *Cryptology ePrint Archive*, 2026.

Adrien Benamira, David Gerault, Thomas Peyrin, and Quan Quan Tan. A deeper look at machine learning-based cryptanalysis. Cryptology ePrint Archive, Paper 2021/287, 2021.

Avrim Blum, Alan Frieze, Ravi Kannan, and Santosh Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22(1):35–52, 1998.

Marcus Brinkmann, Nicolai Kraus, and Alexander May. Halfspace learning for lattice signature key recovery from signs. In *Annual International Cryptology Conference*, pp. 359–390. Springer, 2026.

Mark Cesar and Ryan Rogers. Bounding, concentrating, and truncating: Unifying privacy loss composition for data analytics. In *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pp. 421–457, 2021. URL `https://proceedings.mlr.press/v132/cesar21a.html`.

Leheng Chen, Zihao Liu, Wanyi He, and Bin Dong. Iteris. Agentic Research Loops for Computational Mathematics. *arXiv preprint 2606.02484*, 2026.

Yi Chen, Yantian Shen, Hongbo Yu, and Sitong Yuan. Neural aided statistical attack for cryptanalysis. Cryptology ePrint Archive, Paper 2020/1620, 2020.

Simon Damm, Nicolai Kraus, Alexander May, Julian Nowakowski, and Jonas Thietke. One bit to rule them all–imperfect randomness harms lattice signatures. In *IACR International Conference on Public-Key Cryptography*, pp. 284–316. Springer, 2025.

Simon Damm, Nicolai Kraus, Alexander May, Julian Nowakowski, and Jonas Thietke. One (noisy) bit to rule them all. key recovery from randomness leakage in ml-dsa. *Journal of Cryptology*, 39 (4), 2026.

Mayar Elfares, Pascal Reisert, Tilman Dietz, Manpa Barman, Ahmed Zaki, Ralf Küsters, and Andreas Bulling. CryptoQA: A Large-scale Question-answering Dataset for AI-assisted Cryptography, December 2025.

Lukas Fluri, Avital Shafran, Nicholas Carlini, Matthew Jagielski, Milad Nasr, Orr Dunkelman, Eyal Ronen, and Florian Tramèr. CryptanalysisBench. can LLMs do cryptanalysis? *arXiv preprint 2607.18538*, 2026.

Pierre-Alain Fouque, Phillip Gajland, Hubert de Groote, Jonas Janneck, and Eike Kiltz. A closer look at Falcon. Cryptology ePrint Archive, Paper 2024/1769, 2024. URL https://eprint.iacr.org/2024/1769.

Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pp. 197–206, 2008. doi: 10.1145/1374376.1374407.

Fred Glover. Tabu search—part i. *ORSA Journal on computing*, 1(3):190–206, 1989.

Aron Gohr. Improving attacks on round-reduced speck32/64 using deep learning. Cryptology ePrint Archive, Paper 2019/037, 2019.

Julius Hermelink, Silvan Streit, Erik Mårtensson, and Richard Petri. A generic framework for side-channel attacks against lwe-based cryptosystems. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 3–32. Springer, 2025.

Christopher Hojny, Mathieu Besançon, Ksenia Bestuzheva, Sander Borst, João Dionísio, Johannes Ehls, Leon Eifler, Mohammed Ghannam, Ambros Gleixner, Adrian Göß, Alexander Hoen, Jacob von Holly-Ponientzietz, Rolf van der Hulst, Dominik Kamp, Thorsten Koch, Kevin Kofler, Jurgen Lentz, Marco Lübbecke, Stephen J. Maher, Paul Matti Meinhold, Gioni Mexi, Til Mohr, Erik Mühmer, Krunal Kishor Patel, Marc E. Pfetsch, Sebastian Pokutta, Chantal Reinartz Groba, Felipe Serrano, Yuji Shinano, Mark Turner, Stefan Vigerske, Matthias Walter, Dieter Weninger, and Liding Xu. The scip optimization suite 10.0, 2025. URL https://arxiv.org/abs/2511.18580.

Huiwen Jia, Shiduo Zhang, Yang Yu, and Chunming Tang. Revisiting the concrete security of Falcon-type signatures. Cryptology ePrint Archive, Paper 2026/096, 2026. URL https://eprint.iacr.org/2026/096.

Heechang Kim, Ernest Ryu, and Shuvomoy Das Gupta. A domain-specific harness for end-to-end automation of optimization research. *arXiv preprint 2608.07407*, 2026.

Cathy Yuanchen Li, Emily Wenger, Zeyuan Allen-Zhu, Francois Charton, and Kristin Lauter. SALSA VERDE: A machine learning attack on Learning With Errors with sparse small secrets, October 2023.

Yu Li, Qizhi Pei, Mengyuan Sun, Honglin Lin, Chenlin Ming, Xin Gao, Jiang Wu, Conghui He, and Lijun Wu. CipherBank: Exploring the Boundary of LLM Reasoning Capabilities through Cryptography Challenges, April 2025.

Xiuhan Lin, Shiduo Zhang, Yang Yu, Weijia Wang, Qidi You, Ximing Xu, and Xiaoyun Wang. Thorough power analysis on falcon gaussian samplers and practical countermeasure. In *IACR International Conference on Public-Key Cryptography*, pp. 229–258. Springer, 2025.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery, September 2024.

Yiping Ma, Yu-Lin Tsai, Mayank Rathee, Deevashwer Rathee, François Dupressoir, Pierre-Yves Strub, and Raluca Ada Popa. Shannonprover: Towards automating formal cryptographic proofs, 2026. URL https://arxiv.org/abs/2607.02847.

Utsav Maskey, Chencheng Zhu, and Usman Naseem. Benchmarking Large Language Models for Cryptanalysis and Side-Channel Vulnerabilities, May 2026.

Lajos Muzsai, David Imolai, and András Lukács. Improving LLM Agents with Reinforcement Learning on Cryptographic CTF Challenges, August 2025.

Milad Nasr and Nicholas Carlini. Cryptanalysis of 7-round AES via the algebraic structure of its S-box. Technical Report, Anthropic, July 2026.

National Institute of Standards and Technology. NIST Releases First 3 Finalized Post-Quantum Encryption Standards. `www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards`, August 2024. Accessed 2026-08-19.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Trace2skill: Distill trajectory-local lessons into transferable agent skills, 2026. URL `https://arxiv.org/abs/2603.25158`.

NIST Digital Library of Mathematical Functions. Chapter 20, section 20.5: Infinite products and related results. `https://dlmf.nist.gov/20.5`. Jacobi theta-product identities; accessed 2026-09-09.

Paco Azevedo Oliveira, Andersson Calle Viera, Benoît Cogliati, and Louis Goubin. Uncompressing dilithium's public key. *Cryptology ePrint Archive*, 2024.

Niklas Paskarbeit, Carsten Schubert, Marian Margraf, and Jean-Pierre Seifert. Statistical inference from noisy randomness leakage for ml-dsa attacks. *Cryptology ePrint Archive*, 2026.

Thomas Prest. Sharper bounds in lattice-based cryptography using the Rényi divergence. In *Advances in Cryptology—ASIACRYPT 2017, Part I*, pp. 347–374, 2017. Cryptology ePrint Archive, Paper 2017/480.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. Agent Laboratory: Using LLM Agents as Research Assistants. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 5977–6043, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.320.

Carsten Schubert, Niklas Julius Müller, Jean-Pierre Seifert, and Marian Margraf. Descent into broken trust. uncovering ml-dsa subkeys with scarce leakage and local optimization. *Cryptology ePrint Archive*, 2026.

The Sage Developers. *SageMath, the Sage Mathematics Software System*, 2022. URL `https://www.sagemath.org`. DOI 10.5281/zenodo.6259615.

Ruize Wang, Kalle Ngo, Joel Gärtner, and Elena Dubrova. Unpacking needs protection. *IACR Communications in Cryptology*, 1(3), 2024.

Yu Wang, Yijian Liu, Liheng Ji, Han Luo, Wenjie Li, Xiaofei Zhou, Chiyun Feng, Puji Wang, Yuhan Cao, Geyuan Zhang, Xiaojian Li, Rongwu Xu, Yilei Chen, and Tianxing He. AICrypto: Evaluating Cryptography Capabilities of Large Language Models, May 2026.

Emily Wenger, Mingjie Chen, François Charton, and Kristin Lauter. SALSA: Attacking Lattice Cryptography with Transformers, April 2023.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist-v2: Workshop-Level Automated Scientific Discovery via Agentic Tree Search, April 2025.

Mario Zechner et al. Pi: A minimal terminal coding harness. `https://github.com/earendil-works/pi`, 2026. GitHub repository.

Dechen Zhang, Xuan Tang, Xinxiang Yin, Xingwu Chen, Jian Qian, and Difan Zou. VALG. an agentic system for ML theory research. *arXiv preprint 2608.13060*, 2026.

Shiduo Zhang, Xiuhan Lin, Yang Yu, and Weijia Wang. Improved power analysis attacks on falcon. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 565–595. Springer, 2023.

Shiduo Zhang, Huiwen Jia, Delong Ran, Yang Yu, Yu Yu, and Xiaoyun Wang. GPV preimage sampling with weak smoothness and its applications to lattice signatures. In *Advances in Cryptology—ASIACRYPT 2025*, 2025.

Max Zimmer, Nico Pelleriti, Christophe Roux, and Sebastian Pokutta. The agentic researcher: A practical guide to AI-assisted research in mathematics and machine learning. In *ICML 2026 Workshop: AI as a Tool for Mathematics, Computer Science, and Machine Learning*, 2026. URL https://openreview.net/forum?id=vpcw03stJR.

A ML-DSA

The following algorithms present simplified versions of ML-DSA key generation, signing and verification. The routines $\mathrm{ExpandA}$, $\mathrm{Power2Round}$, $\mathrm{LowBits}$, $\mathrm{SampleInBall}$, $\mathrm{MakeHint}$, and $\mathrm{UseHint}$, are used exactly as specified in ML-DSA. We omit implementation details such as byte encoding, domain separation, seed expansion, context handling, and NTT representations, while retaining all operations relevant to our public-key reconstruction.

We denote vectors using bold lowercase letters and matrices using bold uppercase letters. ML-DSA uses the rings

$$R = \mathbb{Z}[X]/(X^n + 1) \qquad \text{and} \qquad R_q = \mathbb{Z}_q[X]/(X^n + 1), \qquad n = 256.$$

Elements of the rings $\mathbb{Z}$, $\mathbb{Z}_q$, R, and R_q are denoted by italicized lowercase letters. For a polynomial vector $\mathbf{x} = (x^{(1)}, \ldots, x^{(m)})$, multiplication by $c \in R$ is applied componentwise:

$$c\mathbf{x} = \left(cx^{(1)}, \ldots, cx^{(m)}\right).$$

When an element of R occurs in an operation over R_q, its coefficients are implicitly reduced modulo q. The coefficients of a polynomial $x \in R$ or $x \in R_q$ are denoted by x_j, so that

$$x = x_0 + x_1 X + \cdots + x_{n-1} X^{n-1}.$$

Thus, a single coefficient x_j is an element of $\mathbb{Z}$ if $x \in R$ and an element of $\mathbb{Z}_q$ if $x \in R_q$.

Algorithm 1: Simplified ML-DSA key generation

Input: ML-DSA parameters
Output: Public key pk and secret key sk
Sample ρ, $\mathbf{s}_1 \in R^\ell$, and $\mathbf{s}_2 \in R^k$
$\mathbf{A} \leftarrow \mathrm{ExpandA}(\rho) \in R_q^{k \times \ell}$
$\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2 \in R_q^k$
$(\mathbf{t}_1, \mathbf{t}_0) \leftarrow \mathrm{Power2Round}(\mathbf{t}, d) \in R^k \times R^k$
$\mathsf{pk} \leftarrow (\rho, \mathbf{t}_1)$
$\mathsf{sk} \leftarrow (\rho, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$
return $(\mathsf{pk}, \mathsf{sk})$

Algorithm 2: Simplified ML-DSA signing

Input: Secret key $(\rho, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$ and message M
Output: Signature $(\widetilde{c}, \mathbf{z}, \mathbf{h})$
$\mathbf{A} \leftarrow \mathrm{ExpandA}(\rho) \in R_q^{k \times \ell}$
repeat
  Sample masking vector $\mathbf{y} \in R^\ell$
  $\mathbf{w} \leftarrow \mathbf{A}\mathbf{y} \in R_q^k$
  $\mathbf{w}_1 \leftarrow \mathrm{HighBits}(\mathbf{w}, 2\gamma_2) \in R^k$
  $\widetilde{c} \leftarrow H(M \,\|\, \mathbf{w}_1)$
  $c \leftarrow \mathrm{SampleInBall}(\widetilde{c}) \in R$
  $\mathbf{z} \leftarrow \mathbf{y} + c\mathbf{s}_1 \in R^\ell$
  $\mathbf{r} \leftarrow \mathbf{w} - c\mathbf{s}_2 \in R_q^k$
  if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\,\mathrm{LowBits}(\mathbf{r}, 2\gamma_2)\|_\infty \geq \gamma_2 - \beta$ **then**
    restart
  if $\|c\mathbf{t}_0\|_\infty \geq \gamma_2$ **then**
    restart
  $\mathbf{h} \leftarrow \mathrm{MakeHint}(-c\mathbf{t}_0, \mathbf{r} + c\mathbf{t}_0, 2\gamma_2) \in R^k$
until $\mathrm{wt}(\mathbf{h}) \leq \omega$
return $(\widetilde{c}, \mathbf{z}, \mathbf{h})$

Algorithm 3: Simplified ML-DSA verification

Input: Public key $(\rho, \mathbf{t}_1)$, message M, and signature $(\widetilde{c}, \mathbf{z}, \mathbf{h})$.
Output: true or false
if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\mathrm{wt}(\mathbf{h}) > \omega$ **then**
  return false
$\mathbf{A} \leftarrow \mathrm{ExpandA}(\rho) \in R_q^{k \times \ell}$
$c \leftarrow \mathrm{SampleInBall}(\widetilde{c}) \in R$
$\mathbf{v} \leftarrow \mathbf{A}\mathbf{z} - 2^d c\mathbf{t}_1 \in R_q^k$
$\mathbf{w}_1' \leftarrow \mathrm{UseHint}(\mathbf{h}, \mathbf{v}, 2\gamma_2) \in R^k$
$\widetilde{c}' \leftarrow H(M \,\|\, \mathbf{w}_1')$
return $\widetilde{c}' = \widetilde{c}$

A.1 ML-DSA PARAMETER SETS

All three ML-DSA parameter sets use the ring degree $n = 256$, modulus $q = 8\,380\,417$, and rounding parameter $d = 13$. The remaining parameters are listed in Table 3. Here, (k, ℓ) determines the dimensions of $\mathbf{A} \in R_q^{k \times \ell}$, η bounds the secret-key coefficients, and τ is the Hamming weight of the challenge c. Furthermore, $\beta = \tau\eta$, while γ_1, γ_2, and ω determine the rejection bounds, decomposition intervals, and maximum hint weight, respectively.

Table 3: Parameters of the three ML-DSA parameter sets.

Parameter	**ML-DSA-44**	**ML-DSA-65**	**ML-DSA-87**
NIST security category	2	3	5
(k, ℓ)	$(4, 4)$	$(6, 5)$	$(8, 7)$
η	2	4	2
τ	39	49	60
$\beta = \tau\eta$	78	196	120
γ_1	2^{17}	2^{19}	2^{19}
γ_2	$\frac{q-1}{88} = 95\,232$	$\frac{q-1}{32} = 261\,888$	$\frac{q-1}{32} = 261\,888$
ω	80	55	75

B Detailed Algorithm for ML-DSA Public-Key Decompression

Each signature yields linear inequalities on the coefficients of every component $\mathbf{t}_0^{(j)}$. Processing all collected inequalities simultaneously would produce an unnecessarily large optimization problem. We therefore recover the k components independently and in stages. Each stage retains only inequalities that constrain a box around the current estimate, computes a new approximate feasible point, and then reduces the box radius. A final local search repairs any remaining violations over the integer domain.

The routines $\mathrm{ExpandA}$, $\mathrm{SampleInBall}$, $\mathrm{Decompose}$, and $\mathrm{UseHint}$ are used exactly as specified in ML-DSA. The attack-specific routines are given below.

Inequalities obtained from a signature. Let $\alpha = 2\gamma_2$ and $B = \gamma_2 - \beta$. For one component of the verifier computation, define

$$\mathbf{r} = \mathbf{A}\mathbf{z} - 2^d c\mathbf{t}_1 = \mathbf{a} + c\mathbf{t}_0 \pmod q,$$

where $\mathbf{a}$ denotes the value whose low bits were checked during signing. For each coefficient i, the hint determines $a_{1,i} = \mathrm{UseHint}(h_i, r_i)$. Therefore, $\delta_i = \mathrm{center}_q(r_i - \alpha a_{1,i}) = (c\mathbf{t}_0)_i + a_{0,i}$, where $a_{0,i}$ is the corresponding low-order part of $\mathbf{a}$. Here, center_q returns the centered representative modulo q. Since a valid signature satisfies $|a_{0,i}| < B$, we obtain

$$\delta_i - B + 1 \le (c\mathbf{t}_0)_i \le \delta_i + B - 1.$$

Thus, every signature provides two linear inequalities per coefficient and component of $\mathbf{t}_0$.

Selection of relevant inequalities. An inequality is relevant to a box if it is not satisfied throughout the entire box. For $\mathcal{B} = [\boldsymbol{\ell}, \mathbf{u}]$, the maximum of $\mathbf{a}^\top \mathbf{x}$ is obtained by selecting the upper or lower endpoint of each coordinate according to the sign of the corresponding coefficient.

Approximate feasible point. For a system $\mathbf{M}\mathbf{x} \le \mathbf{b}$, define the squared positive violation

$$F(\mathbf{x}) = \frac{1}{2} \left\| \max(\mathbf{M}\mathbf{x} - \mathbf{b}, \mathbf{0}) \right\|_2^2 .$$

The following routine approximately minimizes this objective using gradient steps with a line search over the currently violated inequalities. It is not a linear-programming solver; its purpose is only to produce a sufficiently accurate center for the next stage.

Integer repair. Rounding the approximate solution likely reintroduces a small number of violated inequalities. The final repair uses these violated inequalities to determine the correct $\mathbf{t}_0$. At each iteration, it selects the most violated row and considers unit changes to the incident coordinates. A tabu tenure prevents immediate reversal of recent moves.

Algorithm 4: Recovery of $\mathbf{t}_0$

Input: Public key $(\rho, \mathbf{t}_1)$; signatures; coefficient bounds L, U; radii $R_1 > \cdots > R_s$; row caps $M_1, \ldots, M_s$; relaxation limit K; tabu tenure T; and repair limit $I_{\max}$
Output: Candidate $\widehat{\mathbf{t}}_0 = (\widehat{\mathbf{t}}_0^{(1)}, \ldots, \widehat{\mathbf{t}}_0^{(k)})$ and a Boolean indicating feasibility
Expand $\mathbf{A}$ from ρ
for $j \leftarrow 1$ **to** k **do**
  $\widehat{\mathbf{t}}_0^{(j)} \leftarrow \mathbf{0}$
  $\mathcal{C}_j \leftarrow \varnothing$
for $r \leftarrow 1$ **to** s **do**
  for $j \leftarrow 1$ **to** k **do**
    $\mathcal{B}_j \leftarrow [\widehat{\mathbf{t}}_0^{(j)} - R_r, \widehat{\mathbf{t}}_0^{(j)} + R_r] \cap [L, U]^n$
    Remove from $\mathcal{C}_j$ every $(\mathbf{a}, b)$ for which $\mathrm{RelevantToBox}(\mathbf{a}, b, \mathcal{B}_j) = \mathsf{false}$
  while $|\mathcal{C}_j| < M_r$ *for some* j **do**
    Obtain a signature $(\widetilde{c}, \mathbf{z}, \mathbf{h})$
    $c \leftarrow \mathrm{SampleInBall}(\widetilde{c})$
    $\mathbf{r}' \leftarrow \mathbf{A}\mathbf{z} - 2^d c\mathbf{t}_1$
    for $j \leftarrow 1$ **to** k **do**
      if $|\mathcal{C}_j| < M_r$ **then**
        $\mathcal{I} \leftarrow \mathrm{ExtractInequalities}(c, \mathbf{r}'^{(j)}, \mathbf{h}^{(j)})$
        foreach $(\mathbf{a}, b) \in \mathcal{I}$ **do**
          if $\mathrm{RelevantToBox}(\mathbf{a}, b, \mathcal{B}_j) = \mathsf{true}$ **then**
            Add $(\mathbf{a}, b)$ to $\mathcal{C}_j$
  for $j \leftarrow 1$ **to** k **do**
    $\widehat{\mathbf{t}}_0^{(j)} \leftarrow \mathrm{ApproximateFeasiblePoint}(\mathcal{C}_j, \widehat{\mathbf{t}}_0^{(j)}, L, U, K)$
$\mathsf{ok} \leftarrow \mathsf{true}$
for $j \leftarrow 1$ **to** k **do**
  $(\widehat{\mathbf{t}}_0^{(j)}, \mathsf{ok}_j) \leftarrow \mathrm{IntegerRepair}(\mathcal{C}_j, \widehat{\mathbf{t}}_0^{(j)}, L, U, T, I_{\max})$
  $\mathsf{ok} \leftarrow \mathsf{ok} \wedge \mathsf{ok}_j$
return $(\widehat{\mathbf{t}}_0, \mathsf{ok})$

Algorithm 5: ChallengeMatrix

Input: Challenge polynomial $c = \sum_{i=0}^{n-1} c_i X^i$
Output: Negacyclic multiplication matrix $\mathbf{C} \in \mathbb{Z}^{n \times n}$
for $i \leftarrow 0$ **to** $n-1$ **do**
  for $j \leftarrow 0$ **to** $n-1$ **do**
    if $i \geq j$ **then**
      $C_{i,j} \leftarrow c_{i-j}$
    else
      $C_{i,j} \leftarrow -c_{n+i-j}$
return $\mathbf{C}$

Algorithm 6: ExtractInequalities

Input: Challenge c, verifier value $\mathbf{r} \in \mathbb{Z}_q^n$, and hint $\mathbf{h} \in 0, 1^n$
Output: Set $\mathcal{I}$ of inequalities $\mathbf{a}^\top \mathbf{x} \leq b$
$\alpha \leftarrow 2\gamma_2$
$B \leftarrow \gamma_2 - \beta$
$\mathbf{C} \leftarrow \text{ChallengeMatrix}(c)$
$\mathcal{I} \leftarrow \varnothing$
for $i \leftarrow 0$ **to** $n-1$ **do**
  $a_{1,i} \leftarrow \text{UseHint}(h_i, r_i)$
  $\delta_i \leftarrow \text{center}_q(r_i - \alpha a_{1,i})$
  $\mathbf{a} \leftarrow \mathbf{C}_{i,*}$
  Add $(\mathbf{a}, \delta_i + B - 1)$ to $\mathcal{I}$
  Add $(-\mathbf{a}, -\delta_i + B - 1)$ to $\mathcal{I}$
return $\mathcal{I}$

Algorithm 7: RelevantToBox

Input: Inequality $\mathbf{a}^\top \mathbf{x} \leq b$ and box $\mathcal{B} = [\boldsymbol{\ell}, \mathbf{u}]$
Output: Whether the inequality constrains $\mathcal{B}$
$m \leftarrow 0$
for $i \leftarrow 0$ **to** $n-1$ **do**
  if $a_i \geq 0$ **then**
    $m \leftarrow m + a_i u_i$
  else
    $m \leftarrow m + a_i \ell_i$
return $[m > b]$

Algorithm 8: ApproximateFeasiblePoint

Input: Inequalities $\mathcal{C}$, initial point $\mathbf{x}$, bounds L, U, and iteration limit K
Output: An integer candidate $\mathbf{x}$
Construct $\mathbf{M}$ and $\mathbf{b}$ from $\mathcal{C} = (\mathbf{a}_r, b_r)_r$; Convert $\mathbf{M}$, $\mathbf{b}$, and $\mathbf{x}$ to floating point
for $i \leftarrow 0$ **to** $K-1$ **do**
  $\mathbf{v} \leftarrow \max(\mathbf{M}\mathbf{x} - \mathbf{b}, \mathbf{0})$;
  if $\mathbf{v} = \mathbf{0}$ **then**
    break
  $\mathbf{g} \leftarrow \mathbf{M}^\top \mathbf{v}$
  $\mathcal{V} \leftarrow \{r : v_r > 0\}$
  $\mathbf{d} \leftarrow \mathbf{M}\mathbf{g}$
  $D \leftarrow \sum_{r \in \mathcal{V}} d_r^2$
  if $D = 0$ **then**
    break
  $\lambda \leftarrow \|\mathbf{g}\|_2^2 / D$
  $\mathbf{x} \leftarrow \mathbf{x} - \lambda \mathbf{g}$
$\mathbf{x} \leftarrow \text{clip}(\text{round}(\mathbf{x}), L, U)$
return $\mathbf{x}$

Algorithm 9: IntegerRepair

Input: Inequalities $\mathcal{C}$, initial integer candidate $\mathbf{x}$, bounds L, U, tabu tenure T, iteration limit $I_{\max}$
Output: A candidate $\mathbf{x}$ and a Boolean indicating feasibility

Construct $\mathbf{M}$ and $\mathbf{b}$ from $\mathcal{C} = (\mathbf{a}_r, b_r)_r$; $\mathbf{v} \leftarrow \mathbf{M}\mathbf{x} - \mathbf{b}$; $\mathrm{lastMoved}[j] \leftarrow -\infty$ for all j
for $i \leftarrow 0$ **to** $I_{\max} - 1$ **do**
  $\mathcal{V} \leftarrow r : v_r > 0$
  if $\mathcal{V} = \varnothing$ **then**
    return $(\mathbf{x}, \mathsf{true})$
  $r^\star \leftarrow \arg\max_{r \in \mathcal{V}} v_r$
  $\mathcal{S} \leftarrow \{j : M_{r^\star, j} \neq 0\}$
  $P^\star \leftarrow \sum_r \max(v_r, 0)$
  $(j^\star, \delta^\star) \leftarrow \bot$
  foreach $j \in \mathcal{S}$ **do**
    $\delta \leftarrow -\operatorname{sign}(M_{r^\star, j})$
    if $L \le x_j + \delta \le U$ *and* $\mathrm{lastMoved}[j] + T < i$ **then**
      $P \leftarrow \sum_r \max\big(v_r + \delta M_{r,j}, 0\big)$
      if $P < P^\star$ **then**
        $(j^\star, \delta^\star) \leftarrow (j, \delta)$
        $P^\star \leftarrow P$
  if $(j^\star, \delta^\star) = \bot$ **then**
    Choose $j^\star \in \mathcal{S}$ uniformly at random
    $\delta^\star \leftarrow \mathrm{clip}\big(x_{j^\star} - \operatorname{sign}(M_{r^\star, j^\star}), L, U\big) - x_{j^\star}$
  if $\delta^\star \neq 0$ **then**
    $x_{j^\star} \leftarrow x_{j^\star} + \delta^\star$
    $\mathbf{v} \leftarrow \mathbf{v} + \delta^\star \mathbf{M}_{*, j^\star}$
    $\mathrm{lastMoved}[j^\star] \leftarrow i$
return $(\mathbf{x}, \mathsf{false})$

C FINITE DENSE FIRST-MOMENT BOUNDS FOR ML-DSA

We prove a conditional first-moment bound for one low-part polynomial under the interval model.

C.1 OBSERVATION MODEL

Use the ML-DSA notation and parameters in Table 3. For a residual coefficient e, the strict rejection bound is $|e| < \gamma_2 - \beta$. Consequently its integer interval and cardinality are

$$I = \{-\gamma_2 + \beta + 1, \dots, \gamma_2 - \beta - 1\}, \qquad Q = |I| = 2(\gamma_2 - \beta) - 1 = 2(\gamma_2 - \tau\eta) - 1.$$

Assume $1 \le \tau \le n$ and $\gamma_2 > \beta$.

For a polynomial $f \in R$, let $\|f\|_1 = \sum_{j=0}^{n-1} |f_j|$. Let $x^\star \in \mathbb{Z}^n$ be the fixed hidden vector and $\mathcal{B} \subset \mathbb{Z}^n$ a finite candidate set containing it. For one ML-DSA low-part polynomial, the coefficient box is J_d^n, where $J_d = \{-2^{d-1} + 1, \dots, 2^{d-1}\}$.

Write $c = (c_0, \dots, c_{n-1}) \in \{0, \pm 1\}^n$ for one challenge. For a candidate $x \in \mathcal{B}$, let $\delta = x - x^\star$ be its coefficient difference from the true vector. Let $\mathsf{C}(c) \in \mathbb{Z}^{n \times n}$ be the matrix of multiplication by c in $\mathbb{Z}[X]/(X^n + 1)$. Thus, with indices reduced modulo n,

$$(\mathsf{C}(c)\delta)_i = \sum_{s \in \mathrm{supp}(c)} c_s \sigma_{i,s} \delta_{i-s}, \qquad \sigma_{i,s} \in \{\pm 1\},$$

Here i labels the output coefficient, s a challenge position, and $i - s \pmod n$ the corresponding input position, with representatives in $\{0, \dots, n-1\}$. The sign $\sigma_{i,s}$ is $+1$ when $s \le i$ and -1 when $s > i$, accounting for reduction by $X^n = -1$. The set $\mathrm{supp}(c) = \{s : c_s \ne 0\}$ is the challenge support. All products in this observation model are over the integers, without reduction modulo q.

A block consists of the challenge c and its residual vector $(e_0, \dots, e_{n-1})$. Conditional on the fixed $x^\star$, assume:

(H1) Each challenge has a uniformly chosen τ-element support in $\mathbb{Z}/n\mathbb{Z}$, with independent uniform nonzero signs.

(H2) Within each block, the residual coefficients $e_0, \dots, e_{n-1}$ are independent and uniform on I, and the residual vector is independent of the challenge.

(H3) The blocks are independent and identically distributed.

These are the distributional hypotheses of the model. Their application to accepted ML-DSA signatures is heuristic. No distribution on $x^\star$ is assumed. Unless a subscript specifies otherwise, $\Pr$ and $\mathbb{E}$ take probability and expectation over the sampled blocks, with $x^\star$ and $\mathcal{B}$ fixed. The notation $\mathbb{E}_c$ averages over the challenge alone.

A difference $\delta \in \mathbb{Z}^n$ survives one block when

$$e_i + (\mathsf{C}(c)\delta)_i \in I \qquad (0 \le i < n).$$

Let $T \in \mathbb{N} = \{0, 1, 2, \dots\}$ be the number of observed blocks, and define

$$\mathcal{X}_T = \{x \in \mathcal{B} : x - x^\star \text{ survives all } T \text{ blocks}\}.$$

This is the set of all candidates satisfying the T interval observations; $|\mathcal{X}_T| - 1$ counts its alternatives to $x^\star$.

Lemma C.1.1 (First-moment restriction)**.** *Suppose $x^\star \in \mathcal{B}$ and let*

$$\Delta = \{\delta \in \mathbb{Z}^n \setminus \{0\} \mid x^\star + \delta \in \mathcal{B}\}.$$

Thus Δ is the set of all nonzero admissible differences. Every family $\mathcal{D} \subseteq \Delta$ selected from these differences satisfies

$$\mathbb{E}[|\mathcal{X}_T| - 1] \ge \sum_{\delta \in \mathcal{D}} \Pr[\delta \textit{ survives } T \textit{ blocks}].$$

Proof. For $\delta \in \Delta$, let I_δ equal one if δ survives all T blocks and zero otherwise. Since the zero difference always survives,

$$|\mathcal{X}_T| - 1 = \sum_{\delta \in \Delta} I_\delta.$$

Taking expectations and using $\mathcal{D} \subseteq \Delta$ gives

$$\mathbb{E}[|\mathcal{X}_T| - 1] = \sum_{\delta \in \Delta} \Pr[\delta \text{ survives } T \text{ blocks}] \geq \sum_{\delta \in \mathcal{D}} \Pr[\delta \text{ survives } T \text{ blocks}].$$

□

The left-hand side is the expected number of surviving alternatives in $\mathcal{B}$. The restriction to $\mathcal{D}$ discards nonnegative contributions from other candidates, so evaluating one admissible family suffices for a lower bound on the full expectation.

Restricted family. We consider differences whose nonzero coefficients have magnitude one or two. For integers $u, v \geq 0$ with $1 \leq u + v \leq n$, let

$$\mathcal{D}_{u,v} = \{\delta \in \{0, \pm 1, \pm 2\}^n \mid \#\{i \mid |\delta_i| = 1\} = u, \quad \#\{i \mid |\delta_i| = 2\} = v\}.$$

Thus u and v count coefficients of magnitudes one and two, and $n - u - v$ counts zero coefficients. The profile cardinality is

$$N_{u,v} = |\mathcal{D}_{u,v}| = \frac{n!}{(n-u-v)!\, u!\, v!}\, 2^{u+v}.$$

By Lemma C.3.1,

$$L_{u,v} = \mathbb{E}_c \|\mathsf{C}(c)\delta\|_1$$

is independent of the choice of fixed $\delta \in \mathcal{D}_{u,v}$. It is the expected total absolute coefficient displacement caused by one random challenge. Moreover, $L_{u,v} \leq 2n\tau$ by the triangle inequality. Assume $2n\tau < Q$, as holds for all three ML-DSA parameter sets. If $\mathcal{D}_{u,v} \subseteq \Delta$, then Lemmas C.1.1 and C.2.1 give

$$\mathbb{E}[|\mathcal{X}_T| - 1] \geq N_{u,v} \left(1 - \frac{L_{u,v}}{Q}\right)^T.$$

Let $T^\star(u, v)$ be the real exponent at which this lower bound equals one. Solving for that exponent gives

$$T^\star(u, v) = \frac{\log N_{u,v}}{-\log(1 - L_{u,v}/Q)}.$$

All logarithms are natural logarithms.

Parameter evaluation. Substituting the parameters in Table 3 gives $Q = 190\,307$, $523\,383$, and $523\,535$ for ML-DSA-44, ML-DSA-65, and ML-DSA-87, respectively. The bound concerns one polynomial component, so the public-matrix dimensions k, ℓ do not enter its count. We first evaluate the bound for ML-DSA-44. The other two evaluations follow in the final subsection.

Lemma C.1.2 (Finite profile optimization)**.** *For the ML-DSA-44 parameters fixed above,*

$$\max_{1 \leq u+v \leq 256} T^\star(u, v) = T^\star(96, 2) \approx 57\,184.93.$$

Proof. By Lemma C.3.1, hypergeometric sampling of the zero, magnitude-one and magnitude-two coordinates, followed by the finite average over the independent signs, gives an exact rational value for $L_{u,v}$. The certified exhaustive computation evaluates all $33\,152$ pairs with $1 \leq u + v \leq 256$ and encloses every value of $T^\star(u, v)$ by directed logarithm intervals. It certifies $T^\star(96, 2) > 57\,184.93$, while every other profile has crossing below $57\,184.31$. The largest competing profile is $(u, v) = (95, 2)$, with crossing approximately $57\,184.30$. This proves the stated maximum. □

By Lemma C.1.2, we use

$$\mathcal{D} = \mathcal{D}_{96,2}.$$

Theorem C.1 (Dense first-moment bound). *Suppose $x^\star \in \mathcal{B}$ and*

$$x^\star + \delta \in \mathcal{B} \qquad \text{for every } \delta \in \mathcal{D}.$$

Then, for every natural number $T < 57\,185$,

$$\mathbb{E}\big[|\mathcal{X}_T| - 1\big] \geq 1.$$

Equivalently, the conclusion holds for every $T \leq 57\,184$.

Remark C.1.3 (Choice of $\mathcal{D}$). *The following family comparisons assume that each compared family is admissible in the candidate set. In this paragraph, e_j denotes the jth standard basis vector, with entry one at position j and zero elsewhere, for $0 \leq j < n$. For the $2n$ differences $\{\pm e_j\}$, the exact crossing is*

$$\frac{\log(2n)}{\tau[-\log(1 - 1/Q)]} \approx 30\,440.86.$$

Restricting the optimization above to $v = 0$ gives

$$\max_{1 \leq u \leq 256} T^\star(u, 0) = T^\star(93, 0) \approx 56\,903.30.$$

Thus most of the improvement comes from differences with many coordinates of magnitude one. Permitting magnitude two raises the optimum by a further 0.5%. For a fixed support of size 98,

$$N_{96,2} = \binom{98}{2} N_{98,0},$$

which records the additional choices for the locations of the two entries of magnitude two.

The choice of magnitude two rather than a larger secondary magnitude follows from monotonicity. Replacing it by an integer magnitude $m \geq 3$ leaves the cardinality unchanged for fixed class sizes. Conditional on selecting the coefficient being varied, let Y be the signed sum of the other selected coordinates. Averaging over the two signs of that coefficient gives

$$\frac{|Y + m| + |Y - m|}{2} = \max(|Y|, m).$$

The average convolution norm cannot decrease, so $m = 2$ gives the largest crossing among choices of one secondary magnitude with fixed class sizes.

C.2 Survival of one difference

Lemma C.2.1 (Survival of one difference). *For $\delta \in \mathbb{Z}^n$, set*

$$L(\delta) = \mathbb{E}_c \|\mathsf{C}(c)\delta\|_1$$

Thus $L(\delta)$ is the expected sum of absolute displacements over all output coefficients. Let $\theta(\delta)$ be the one-block survival probability of δ, averaging over both challenge and residuals. Then

$$\theta(\delta) \geq 1 - \frac{L(\delta)}{Q}, \tag{2}$$

and

$$\Pr[\delta \text{ survives all } T \text{ blocks}] = \theta(\delta)^T. \tag{3}$$

In particular, if $L(\delta) < Q$, then

$$\Pr[\delta \text{ survives all } T \text{ blocks}] \geq \left(1 - \frac{L(\delta)}{Q}\right)^T.$$

Proof. Let e be one uniform residual coefficient on I, and let $z \in \mathbb{Z}$ be a fixed displacement of that coefficient. Then

$$\Pr[e + z \in I] = \frac{|I \cap (I - z)|}{Q} = \left(1 - \frac{|z|}{Q}\right)_+. \tag{4}$$

Here $(x)_+ = \max\{x, 0\}$. The intersection of the two integer intervals has $(Q - |z|)_+$ elements. Let $g_c(\delta)$ denote the one-block survival probability conditional on the challenge c. Independence of the residual coordinates gives

$$g_c(\delta) = \prod_{i=0}^{n-1} \left(1 - \frac{|(\mathsf{C}(c)\delta)_i|}{Q}\right)_+ .$$

The Weierstrass inequality

$$\prod_i (1 - u_i)_+ \geq 1 - \sum_i u_i$$

holds for any finite collection of nonnegative numbers u_i. Apply it with $u_i = |(\mathsf{C}(c)\delta)_i|/Q$, the absolute displacement at coordinate i divided by the interval cardinality. This gives

$$\theta(\delta) = \mathbb{E}_c g_c(\delta) \geq 1 - \frac{\mathbb{E}_c \|\mathsf{C}(c)\delta\|_1}{Q} = 1 - \frac{L(\delta)}{Q}.$$

Finally, independence of the T blocks gives $\Pr[\delta \text{ survives all } T \text{ blocks}] = \theta(\delta)^T$. □

C.3 Exact norm for the dense type

Lemma C.3.1 (Type reduction). *For every $\delta \in \mathbb{Z}^n$,*

$$L(\delta) = n\, \mathbb{E}_{J,\varepsilon} \left| \sum_{j \in J} \varepsilon_j \delta_j \right|, \tag{5}$$

where J is the random set of selected input positions, uniform among the τ-subsets of $\mathbb{Z}/n\mathbb{Z}$. The variables $\varepsilon_j \in \{-1, +1\}$ are independent uniform signs, independent of J. The expectation $\mathbb{E}_{J,\varepsilon}$ averages over both this set and these signs. Hence $L(\delta)$ depends only on the multiset $\{|\delta_j|\}$.

Proof. Fix an output coordinate i. The map $s \mapsto i - s$ permutes $\mathbb{Z}/n\mathbb{Z}$, so the image of the random challenge support is a uniform τ-subset J. The deterministic wrap factors $\sigma_{i,s}$ and the fixed signs of the entries of δ can be absorbed into the independent signs of the challenge. Thus

$$\mathbb{E}_c |(\mathsf{C}(c)\delta)_i| = \mathbb{E}_{J,\varepsilon} \left| \sum_{j \in J} \varepsilon_j \delta_j \right|.$$

The right-hand side is independent of i. Summing over the n output coordinates proves (5). Independent uniform signs also absorb the signs of the entries of δ, leaving only their magnitudes. □

Lemma C.3.2 (Exact norm for the selected family). *Let $L = L_{96,2}$ be the common challenge-averaged norm for the selected family $\mathcal{D}$. For every $\delta \in \mathcal{D}$,*

$$L(\delta) = L = \frac{24227471878587181954841374467148959010306251336898 9}{29927869870279568002949180131655172172126617600 0}. \tag{6}$$

Proof. Suppose a sampled support contains r_1 coordinates of magnitude one and r_2 coordinates of magnitude two. The number of selected zero coordinates is $r_0 = 39 - r_1 - r_2$. Let $M(r_1, r_2)$ be the expected absolute sum of r_1 independent signed ones and r_2 independent signed twos. Counting their signs gives

$$M(r_1, r_2) = \frac{1}{2^{r_1 + r_2}} \sum_{a=0}^{r_1} \sum_{b=0}^{r_2} \binom{r_1}{a} \binom{r_2}{b} \left| (r_1 - 2a) + 2(r_2 - 2b) \right|.$$

Here a and b are local summation indices: they count negative signs among the selected magnitude-one and magnitude-two coordinates, respectively. The signed sums in the two groups are $r_1 - 2a$ and $2(r_2 - 2b)$, and the binomial factors count the corresponding sign assignments. The factor $2^{-(r_1+r_2)}$ averages over all assignments. The signs attached to selected zero coordinates do not affect the sum. The hypergeometric choice of support and Lemma C.3.1 give

$$L = \frac{256}{\binom{256}{39}} \sum_{r_2=0}^{2} \sum_{r_1=0}^{39-r_2} \binom{158}{39 - r_1 - r_2} \binom{96}{r_1} \binom{2}{r_2} M(r_1, r_2). \tag{7}$$

There are $40 + 39 + 38 = 117$ rational terms. Reducing their sum gives (6). □

In particular $0 < L < Q$. Write b for the common one-block survival lower bound of the selected family:

$$b = 1 - \frac{L}{Q} = \frac{43625043534805435122941601914213476088850722802048 47}{43811408703102259599517304794730006542776155504640 00}. \tag{8}$$

By Lemma C.2.1, every $\delta \in \mathcal{D}$ satisfies

$$\Pr[\delta \text{ survives } T \text{ blocks}] \geq b^T. \tag{9}$$

C.4 Counting and the endpoint

Write $N = N_{96,2}$ for the cardinality of the selected family. The positions of the magnitude-1 and magnitude-2 entries and their 98 signs can be chosen in

$$N = |\mathcal{D}| = \frac{256!}{158!\,96!\,2!}\,2^{98} \tag{10}$$

ways. Distinct differences give distinct translates of $x^\star$. By the admissibility hypothesis, $\mathcal{D} \subseteq \Delta$. Combining Lemma C.1.1 with (9) gives

$$\mathbb{E}[|\mathcal{X}_T| - 1] \geq N b^T. \tag{11}$$

Lemma C.4.1 (Endpoint arithmetic certificate)**.** *For N and b defined in Equations* (8) *and* (10)*,*

$$N b^{57184} > 1. \tag{12}$$

Proof. The certificate uses integer arithmetic in place of floating-point logarithms or a direct expansion of the rational power. Let

$$S = 2^{512}, \qquad B = \lfloor Sb \rfloor, \qquad R = 18200691348764106995433411431651036675079627507 72.$$

The integer S is the fixed-point scale: each stored integer numerator represents its value divided by S. The integer B is the numerator of the initial downward-rounded approximation B/S to b. In the recurrence below, a/S is the current power approximation, r/S is the accumulated product, and e is the remaining nonnegative integer exponent. These three variables are local to this computation. The initial value $r = S$ represents an accumulated product of one; R is the terminal value of the accumulator numerator r. Starting from $(a, r, e) = (B, S, 57\,184)$, repeatedly apply the following updates while $e > 0$, stopping when $e = 0$:

$$\begin{aligned} &e \text{ even}: \quad a \leftarrow \lfloor a^2/S \rfloor, \quad e \leftarrow e/2, \\ &e \text{ odd}: \quad r \leftarrow \lfloor ra/S \rfloor, \quad e \leftarrow e - 1. \end{aligned} \tag{13}$$

The successive exponents are

$$\begin{aligned} &57184, 28592, 14296, 7148, 3574, 1787, 1786, 893, 892, 446, 223, 222, \\ &111, 110, 55, 54, 27, 26, 13, 12, 6, 3, 2, 1, 0. \end{aligned}$$

Exact integer evaluation of the recurrence ends with $r = R$. Each rounded multiplication is a lower bound, and $B/S \leq b$. Induction over the recurrence gives

$$\frac{R}{S} \leq \left(\frac{B}{S}\right)^{57184} \leq b^{57184}.$$

Substitution of (10) gives the exact integer inequality

$$10^6 N R > 1\,003\,979\,S.$$

Therefore

$$N b^{57184} \geq \frac{NR}{S} > \frac{1\,003\,979}{10^6} > 1.$$

□

The Lean development checks the 117 summands in (7), every rounded multiplication in (13), and the final integer comparison.

Proof of Theorem C.1. Let $T < 57\,185$. Since T is a natural number, $T \leq 57\,184$. Because $0 < b < 1$, we have $b^T \geq b^{57184}$. Combining (11) with Lemma C.4.1 gives

$$\mathbb{E}[|\mathcal{X}_T| - 1] \geq N b^T \geq N b^{57184} > 1.$$

This proves the stated weak inequality. □

C.5 Other ML-DSA parameter sets

The same argument applies with the ML-DSA-65 and ML-DSA-87 values of τ and Q given above, keeping $n = 256$. Write $L_{u,v}^{(\tau)}$ for the average convolution norm of a fixed member of $\mathcal{D}_{u,v}$ when the challenge weight is τ. Its exact value is

$$L_{u,v}^{(\tau)} = \frac{256}{\binom{256}{\tau}} \sum_{r_2=0}^{\min(v,\tau)} \sum_{r_1=0}^{\min(u,\tau-r_2)} \binom{256-u-v}{\tau-r_1-r_2} \binom{u}{r_1} \binom{v}{r_2} M(r_1, r_2), \tag{14}$$

Here r_1, r_2 again count selected coordinates of magnitudes one and two; the remaining $\tau - r_1 - r_2$ selected coordinates have magnitude zero. The signed-sum expectation $M(r_1, r_2)$ is defined in the proof of Lemma C.3.2, and invalid binomial coefficients are zero. The family size is

$$N_{u,v} = \frac{256!}{(256-u-v)!\, u!\, v!}\, 2^{u+v}.$$

Let $b_{u,v}^{(\tau,Q)}$ be the one-block survival lower bound for this profile and these parameters, and let $T_{\tau,Q}^{\star}(u,v)$ be the real exponent at which its first-moment lower bound equals one:

$$b_{u,v}^{(\tau,Q)} = 1 - \frac{L_{u,v}^{(\tau)}}{Q}, \qquad T_{\tau,Q}^{\star}(u,v) = \frac{\log N_{u,v}}{-\log b_{u,v}^{(\tau,Q)}},$$

and, for a fixed profile (u, v), let

$$T_{\mathrm{FM}} = \max\left\{ T \in \mathbb{N} :\ N_{u,v} \big(b_{u,v}^{(\tau,Q)}\big)^T \geq 1 \right\}. \tag{15}$$

Thus T_{FM} is the last nonnegative integer exponent for which this particular profile's lower bound is at least one. The crossings in Table 4 are nonintegral, so $T_{\mathrm{FM}} = \lfloor T_{\tau,Q}^{\star}(u,v) \rfloor$.

Table 4: Best signed profile among all 33 152 families $\mathcal{D}_{u,v} \subset \{0, \pm 1, \pm 2\}^{256}$. The values of L and $T^{\star}$ are rounded; T_{FM} is the last integer for which the first-moment lower bound is at least one.

level	τ	Q	(u,v)	$L_{u,v}^{(\tau)}$	$T^{\star}$	T_{FM}
ML-DSA-44	39	190 307	$(96, 2)$	809.5287765	57 184.9317	57 184
ML-DSA-65	49	523 383	$(96, 2)$	908.3896769	140 331.3633	140 331
ML-DSA-87	60	523 535	$(98, 3)$	1 034.7680602	126 746.2914	126 746

For every profile, the computation evaluates $N_{u,v}$ and $L_{u,v}^{(\tau)}$ exactly. Directed intervals for the logarithms certify each maximizer. Applying the fixed-point recurrence (13) with downward rounding at T_{FM} and upward rounding at $T_{\mathrm{FM}} + 1$ gives

	$Nb^{T_{\mathrm{FM}}}$	$Nb^{T_{\mathrm{FM}}+1}$
ML-DSA-44	≥ 1.0039797	≤ 0.9997090
ML-DSA-65	≥ 1.0006312	≤ 0.9988946
ML-DSA-87	≥ 1.0005767	≤ 0.9985991

For each row, N abbreviates $N_{u,v}$ and b abbreviates $b_{u,v}^{(\tau,Q)}$ for that row's parameters and selected profile; the table heading $T^{\star}$ likewise abbreviates $T_{\tau,Q}^{\star}(u,v)$. Upward rounding uses ceilings in the initialization and in every multiplication of the recurrence.

Corollary C.2 (All-level first-moment endpoints)**.** *Choose a row of Table 4 and use its τ, Q, and profile in the observation model. If $x^{\star} \in \mathcal{B}$ and $x^{\star} + \delta \in \mathcal{B}$ for every $\delta \in \mathcal{D}_{u,v}$, then*

$$\mathbb{E}[|\mathcal{X}_T| - 1] \geq 1 \qquad \textit{for every natural number } T \leq T_{\mathrm{FM}}.$$

Proof. Lemmas C.1.1 and C.2.1 and (14) give $\mathbb{E}[|\mathcal{X}_T| - 1] \geq Nb^T$. The endpoint certificate and $0 < b < 1$ give $Nb^T \geq Nb^{T_{\mathrm{FM}}} > 1$ for $T \leq T_{\mathrm{FM}}$. □

The ML-DSA-44 theorem and endpoint are checked in Lean. The three profile maxima and the other two endpoints are certified by exact-arithmetic computation.

Remark C.5.1. *The conclusion is an annealed first-moment statement. It does not imply that a second candidate exists with high probability below the endpoint; that would require control of correlations among surviving differences.*

D SHARPER NORMALIZER BOUNDS FOR A FALCON+-TYPE CORE

Scope and overview. This appendix is mathematically self-contained and independent of the leaked-sign key-recovery experiments. It analyzes exact Klein–GPV sampling in the Falcon+ hash-and-sign framework through the variation of its Gaussian normalizers. A single basis-dependent quantity W controls three objects: the sampler's Rényi divergence from an ideal coset Gaussian, the divergence of a joint syndrome–preimage simulation, and the expected overhead of exact rejection correction. The bounds are $\log R_a \le aW^2/8$ and $\mathbb{E}[T] \le e^W$. Applied to the normalizer intervals of Jia et al. (2026), this gives $R_a \le \exp(a \log^2(1+\delta_{B,s})/8)$, improving the leading coefficient in their Corollary 1 by a factor of four. A direct joint comparison yields a single-transition UF-CMA reduction for an explicitly defined, exact-arithmetic Falcon+-type core. We also derive a sharper fixed-vector discrete-Gaussian distance bound, improving the probabilities reported in their Remark 5 from 0.2627 and 0.867 to approximately 0.956259 and 0.998087. The scope is the classical random oracle model and idealized sampling; no new implementation-level or unconditional strong-unforgeability security level for deployed Falcon is asserted.

D.1 PRIOR WORK AND THE PRECISE IMPROVEMENT

The GPV framework Gentry et al. (2008) implements lattice hash-and-sign signatures by sampling a short preimage of a hash value. Falcon instantiates this framework over NTRU lattices. Its concrete analysis uses Rényi divergence to quantify the distributional error of Gaussian sampling, following the tools of Prest Prest (2017). Fouque, Gajland, de Groote, Janneck, and Kiltz Fouque et al. (2024) analyze Falcon-type signatures concretely; the Falcon+ variant discussed by Jia et al. refreshes the salt on every raw sampling attempt, including retries.

Jia, Zhang, Yu, and Tang Jia et al. (2026) sharpen this analysis by using the actual Gram–Schmidt profile of the trapdoor basis rather than a worst-case bound for every direction. Their Theorem 2 defines

$$\delta_{B,s} = \prod_{i=1}^{m} \frac{1+\varepsilon_i}{1-\varepsilon_i} - 1, \tag{16}$$

and their Corollary 1 gives the displayed, second-order estimate

$$R_a(K_{B,C} \| G_C) \lesssim 1 + \frac{a}{2}\delta_{B,s}^2. \tag{17}$$

Here $K_{B,C}$ is the idealized Klein–GPV residual distribution in coset C, and G_C is the exact discrete Gaussian on that coset. Their Theorem 1 separates the sampler transition from the random-oracle syndrome transition, introducing two Rényi factors and two probability-preservation exponents.

Our change is to retain the range of the log likelihood ratio. The normalizer intervals already used in their proof imply a range of width

$$W_{B,s} = \sum_i \log \frac{1+\varepsilon_i}{1-\varepsilon_i} = \log(1+\delta_{B,s}).$$

The bounded-range implication from concentrated-divergence theory Cesar & Rogers (2021) then gives the nonasymptotic bound

$$\boxed{R_a(K_{B,C} \| G_C) \le \exp\Big(\frac{a}{8}\log^2(1+\delta_{B,s})\Big).} \tag{18}$$

For fixed a and $\delta_{B,s} \to 0$, its right side is $1 + a\delta_{B,s}^2/8 + O_a(\delta_{B,s}^3)$. Thus the factor-four improvement is in the leading coefficient of the displayed sampler bound, not a factor-four increase in bit security. Unlike (17), (18) has no unspecified asymptotic remainder and holds for every $a > 1$.

The second improvement is structural: if a fresh syndrome and its preimage are compared *jointly*, the coset normalizers cancel from the likelihood ratio. The same W bounds the entire joint operation. Section D.6 turns this observation into a complete UF-CMA reduction for the core model defined there, with one Rényi exponent. We do not obtain that reduction by simply deleting terms from an existing theorem.

The rejection correction itself is prior work, used in the weak-smoothness sampler Zhang et al. (2025) and reproduced in Figure 2 of Jia et al. (2026). Our contribution here is a common certificate for its overhead and the uncorrected sampler's divergence. Similarly, the general bounded-range inequality, Poisson summation, and theta-product identities are established tools, not new general theorems. The claimed improvements are over the displayed Falcon-analysis bounds identified above; no comprehensive publication-priority claim is made.

Version convention. Every theorem number, page number, and numerical comparison to Jia et al. (2026) refers to the 32-page version of ePrint 2026/096. Statements about Fouque et al. (2024) are compared in the form reproduced in that version, not presented as an audit of every later revision.

D.2 Lattices, cosets, and the exact sampler

Fix $m \geq 1$ and a full-rank integer basis $B = [b_1, \dots, b_m] \in \mathbb{Z}^{m\times m}$. Write

$$\Lambda = B\mathbb{Z}^m, \qquad g_i = \left\|\widetilde{b}_i\right\| > 0, \qquad s_i = s/g_i, \qquad s > 0,$$

where $\widetilde{b}_i$ are the Gram–Schmidt vectors. The quotient $\mathcal{C} = \mathbb{Z}^m/\Lambda$ has size $M = |\det B|$. Choose one representative $t_C \in C$ for each coset.

For a countable set $S \subset \mathbb{R}^m$, put

$$\rho_s(x) = e^{-\pi\|x\|^2/s^2}, \qquad \rho_s(S) = \sum_{x\in S} \rho_s(x).$$

All Gaussian lattice sums used below are absolutely convergent and positive. Define

$$G_C(x) = \frac{\rho_s(x)}{\rho_s(C)} \quad (x \in C), \qquad D_0(x) = \frac{\rho_s(x)}{\rho_s(\mathbb{Z}^m)} \quad (x \in \mathbb{Z}^m).$$

Thus $D_0 = D_{\mathbb{Z}^m,s}$, equivalently a product of m independent integer Gaussians of width s. Width and standard deviation are related by $s = \sqrt{2\pi}\,\sigma$.

For $u > 0$ and $d \in \mathbb{R}$, let

$$Z_u(d) = \sum_{k\in\mathbb{Z}} e^{-\pi(k-d)^2/u^2}.$$

The exact Klein–GPV residual sampler for coset C begins with target t_C and samples $z_m, z_{m-1}, \dots, z_1$. At step i, set

$$d_i = \frac{\left\langle t_C - \sum_{j>i} z_j b_j, \widetilde{b}_i \right\rangle}{g_i^2}, \qquad \mathbb{P}[z_i = k \mid z_{i+1}, \dots, z_m] = \frac{e^{-\pi(k-d_i)^2/s_i^2}}{Z_{s_i}(d_i)}.$$

Return $x = t_C - \sum_i z_i b_i \in C$. Denote its law by $K_{B,C}$. This is an exact-arithmetic distribution, not an assertion about a particular floating-point implementation.

Lemma D.2.1 (Exact product formula)**.** *For each $x \in C$, let $d_i(x, C)$ be the centers on its unique sampling path, and put*

$$Z_B(x, C) = \prod_{i=1}^{m} Z_{s_i}(d_i(x, C)).$$

Then

$$K_{B,C}(x) = \frac{\rho_s(x)}{Z_B(x, C)}. \tag{19}$$

In particular, $K_{B,C}$ and G_C have the same support C.

Proof. The map $z \mapsto t_C - Bz$ is a bijection from $\mathbb{Z}^m$ to C. Orthogonal decomposition in the Gram–Schmidt vectors gives

$$\|x\|^2 = \sum_i g_i^2 (z_i - d_i)^2.$$

Multiplying the conditional sampling probabilities yields (19). Each conditional integer Gaussian has positive mass at every integer, so every $x \in C$ has positive probability. □

Definition D.1 (Normalizer certificate)**.** Suppose positive constants ℓ_i, u_i satisfy

$$0 < \ell_i \le Z_{s_i}(d) \le u_i \quad \text{for every } d \in \mathbb{R}. \tag{20}$$

Define

$$L_B = \prod_i \ell_i, \qquad U_B = \prod_i u_i, \qquad W = \log(U_B/L_B) = \sum_i \log(u_i/\ell_i) \ge 0.$$

The bounds are uniform over all cosets and all internal sampling centers.

No independence assumption is made about those centers.

D.3 The probability inequality behind the improvement

For distributions P, Q on a countable set with common support, use the multiplicative Rényi convention of Jia et al. (2026):

$$R_a(P\|Q) = \left(\sum_x P(x)^a Q(x)^{1-a} \right)^{1/(a-1)}, \qquad a > 1.$$

Thus $\log R_a$ is logarithmic Rényi divergence.

Lemma D.3.1 (Bounded log likelihood; cf. Cesar & Rogers (2021))**.** *Suppose P and Q have the same support and $\log(P(x)/Q(x))$ lies in an interval of length W. Then, for every $a > 1$,*

$$\log R_a(P\|Q) \le \frac{a}{8} W^2. \tag{21}$$

The same bound holds with P and Q interchanged.

Proof. Write $L = \log(P/Q)$ as a random variable under P. Common support gives $\mathbb{E}_P[e^{-L}] = 1$. Hoeffding's bounded-variable inequality is

$$\log \mathbb{E}[e^{t(X - \mathbb{E}X)}] \le t^2 W^2/8 \quad (t \in \mathbb{R})$$

when X lies in an interval of length W. For completeness, if $h(t) = \log \mathbb{E}[e^{tX}]$, then $h''(t)$ is the variance under the exponentially tilted law. That law has the same range, and its variance is at most $W^2/4$, by measuring squared distance from the interval midpoint. Integrating this second-derivative bound proves the inequality; boundedness justifies differentiation.

Apply it with $X = L$ and $t = -1$ to obtain $0 \le -\mathbb{E}_P L + W^2/8$, hence $\mathbb{E}_P L \le W^2/8$. A second application gives

$$\log \mathbb{E}_P[e^{(a-1)L}] \le (a-1)\frac{W^2}{8} + (a-1)^2 \frac{W^2}{8} = a(a-1)\frac{W^2}{8}.$$

Divide by $a - 1$. Reversing the distributions negates the pointwise log ratio and preserves its range width. □

Remark D.3.2 (Why support matters)**.** *Support inclusion alone is insufficient. If $P = (1, 0)$ and $Q = (1/2, 1/2)$, the log ratio is constant on $\mathrm{Supp}(P)$, but $\log R_a(P\|Q) = \log 2$. The normalization $\mathbb{E}_P e^{-L} = 1$ fails. The exact distributions in Section D.2 do have common support.*

D.4 One certificate for sampling, simulation, and correction

Define two joint laws on pairs (C, x) with $x \in C$:

$$P_B : C \leftarrow \mathrm{Unif}(\mathcal{C}), \quad x \leftarrow K_{B,C},$$
$$Q : x \leftarrow D_0, \quad C = x + \Lambda.$$

Theorem D.2 (Single-certificate theorem)**.** *Under* (20)*, the following statements hold for every* $a > 1$.

(i) *For every coset* C*,*

$$\log R_a(K_{B,C} \| G_C) \le aW^2/8. \tag{22}$$

(ii) *For the joint syndrome–preimage laws,*

$$\log R_a(P_B \| Q) \le aW^2/8. \tag{23}$$

Both inequalities also hold in the reverse direction.

(iii) *If a proposal* $x \leftarrow K_{B,C}$ *is accepted with probability* $Z_B(x, C)/U_B$*, the conditional accepted law is exactly* G_C*. Its acceptance probability and expected number of independent proposals are*

$$p_C = \frac{\rho_s(C)}{U_B} \ge e^{-W}, \qquad \mathbb{E}[T_C] = \frac{U_B}{\rho_s(C)} \le e^{W}. \tag{24}$$

Proof. For (i), Lemma D.2.1 gives

$$\frac{K_{B,C}(x)}{G_C(x)} = \frac{\rho_s(C)}{Z_B(x, C)}.$$

Since $Z_B(x, C) \in [L_B, U_B]$, the logarithm has range at most W. Lemma D.3.1 applies.

For (ii), the exact joint masses are

$$P_B(C, x) = \frac{1}{M} \frac{\rho_s(x)}{Z_B(x, C)}, \qquad Q(C, x) = \frac{\rho_s(x)}{\rho_s(\mathbb{Z}^m)}.$$

Consequently,

$$\boxed{\frac{P_B(C, x)}{Q(C, x)} = \frac{\rho_s(\mathbb{Z}^m)}{MZ_B(x, C)}.} \tag{25}$$

The coset Gaussian normalizers are absent. The two joint laws have the same support, and the log ratio again has range at most W.

For (iii), the unnormalized mass of an accepted proposal is

$$K_{B,C}(x) \frac{Z_B(x, C)}{U_B} = \frac{\rho_s(x)}{U_B}.$$

Summing proves the exact expression for p_C and the conditional Gaussian law. Moreover,

$$\rho_s(C) = \sum_{x \in C} K_{B,C}(x) Z_B(x, C) \in [L_B, U_B].$$

Thus $p_C \ge L_B/U_B = e^{-W}$. Independent repetitions give a geometric waiting time with mean $1/p_C$. □

The syndrome bias has not vanished: it is included in the joint likelihood ratio (25). The theorem avoids separately upper-bounding two intermediate comparisons. It also shows that for small W, the correction overhead is $1 + W + O(W^2)$ while the divergence bound is $1 + aW^2/8 + O_a(W^4)$.

Corollary D.3 (Replacement for Corollary 1 of Jia et al. (2026))**.** *Under the normalizer bounds used in Theorem 2 of Jia et al. (2026), take*

$$u_i = Z_{s_i}(0), \qquad \ell_i = \frac{1 - \varepsilon_i}{1 + \varepsilon_i} Z_{s_i}(0).$$

Then $W = \log(1 + \delta_{B,s})$*, and* (18) *holds. The same right side bounds* $R_a(P_B \| Q)$.

D.5 Adaptive composition and event transfer

Lemma D.5.1 (Conditional composition)**.** *Consider two finite, adaptive experiments with the same initial distribution. Suppose at most N conditional sampling operations differ. For every common history, the conditional laws at a differing operation have common support and log-likelihood range at most W. All other conditional kernels are identical. Then their extended transcript distributions $P_{\mathrm{tr}}, Q_{\mathrm{tr}}$ satisfy*

$$\log R_a(P_{\mathrm{tr}} \| Q_{\mathrm{tr}}) \le \frac{aNW^2}{8}. \tag{26}$$

For every event E determined by the transcript,

$$P_{\mathrm{tr}}(E) \le \exp\left(\frac{(a-1)NW^2}{8}\right) Q_{\mathrm{tr}}(E)^{1-1/a}. \tag{27}$$

Proof. The transcript log likelihood is the sum of the conditional log likelihoods L_i. Lemma D.3.1 gives

$$\mathbb{E}_P[e^{(a-1)L_i} \mid \text{history}] \le e^{a(a-1)W^2/8}.$$

Iterated conditional expectation bounds the moment of their sum by $e^{a(a-1)NW^2/8}$. Identical operations contribute zero; stopped experiments can be padded with identical operations. This proves (26). Finally, Hölder's inequality gives

$$P(E) \le R_a(P\|Q)^{(a-1)/a} Q(E)^{(a-1)/a},$$

which implies (27). □

The dependence is NW^2, not N^2W^2. Applying the one-step range theorem only to a total range bound NW would lose this advantage. Rejected raw samples may be included in the extended transcript even when the adversary does not see them. The actual visible experiment and its winning event are obtained by deterministic projection.

Corollary D.4 (Optimized event exponent)**.** *Assume $Q_{\mathrm{tr}}(E) \le 2^{-\lambda}$ and put $\kappa = NW^2/(8\log 2)$. Then*

$$P_{\mathrm{tr}}(E) \le 2^{-(\max\{0,\sqrt{\lambda}-\sqrt{\kappa}\})^2}. \tag{28}$$

When $0 < \kappa < \lambda$, the optimizing order is $a_* = \sqrt{\lambda/\kappa}$.

Proof. The exponent furnished by (27) is $\lambda - \lambda/a - \kappa(a-1)$. Differentiation gives the stated maximum for $0 < \kappa < \lambda$. For $\kappa = 0$, take $a \to \infty$; for $\kappa \ge \lambda$, the probability bound 1 applies. □

In particular, to retain exponent $\lambda_0 < \lambda$ in this event term, it suffices that

$$W \le \sqrt{\frac{8\log 2}{N}}(\sqrt{\lambda} - \sqrt{\lambda_0}). \tag{29}$$

Additive abort terms remain separate.

D.6 A standalone UF-CMA theorem for a Falcon+-type core

D.6.1 The scheme and the hardness experiment

Let $R = \mathbb{Z}[X]/(X^n+1)$, with n a power of two, and $R_q = R/qR$ for an integer $q \ge 2$. Identify R^2 with $\mathbb{Z}^{2n}$ using coefficient vectors and the Euclidean norm. For $h \in R_q$, define

$$A_h(x_1, x_2) = x_1 + hx_2 \pmod q, \qquad \Lambda_h = \ker A_h.$$

This map is surjective, so $[\mathbb{Z}^{2n} : \Lambda_h] = q^n$. The Falcon parameter families discussed in Jia et al. (2026) use $n = 512$ or 1024, $q = 12289$, and tailcut factor $\tau = 1.1$; the lattice dimension here is $m = 2n$. A key generator outputs (h, B) with B a basis of Λ_h. Fix public $s > 0$, norm bound $\beta > 0$, and a salt space $\mathcal{S}$ of size S. The random oracle is $H(h, \mathsf{m}, r) \in R_q$.

The scheme $\mathsf{CoreFalcon}^+$ is deliberately specified at the mathematical core level. To sign message m, repeatedly choose a fresh independent $r \leftarrow \mathrm{Unif}(\mathcal{S})$, set $c = H(h, \mathsf{m}, r)$, and sample $x \leftarrow K_{B,C_c}$,

where $C_c = A_h^{-1}(c)$. Return (r, x) as soon as $\|x\| \le \beta$. Verification checks that norm bound and $A_h x = H(h, \mathsf{m}, r)$. UF-CMA success requires a valid signature on a message that was never submitted to the signing oracle. This is *plain* unforgeability, not strong unforgeability.

Define the t-target R-ISIS experiment as follows. Draw h from the same key generator's public-key marginal; draw independent uniform $c_1, \dots, c_t \in R_q$; give $(h, c_1, \dots, c_t)$ to the solver. It wins by outputting (j, x) with $1 \le j \le t$, $x \in R^2$, $\|x\| \le \beta$, and $A_h x = c_j$. Write its success probability as $\mathrm{Adv}^{t\text{-R-ISIS}}(\mathcal{B})$. No assertion equating this problem to homogeneous SIS is required.

D.6.2 The security bound

Theorem D.5 (Single-transition UF-CMA reduction)**.** *Suppose every key in the support of the key generator admits the certificate of Definition D.1 with $W(B, s) \le W_*$. Suppose also that, for every such key and syndrome,*

$$\mathbb{P}_{x \leftarrow K_{B,C_c}}[\|x\| \le \beta] \ge p_*, \qquad 0 < p_* \le 1. \tag{30}$$

Let an adversary $\mathcal{A}$ make at most Q_s signing queries and Q_H direct random-oracle queries. For any integer $C_s \ge Q_s$ with $C_s \ge 1$, there is a $(Q_H + 1)$-target R-ISIS solver $\mathcal{B}_\mathcal{A}$ such that, for every $a > 1$,

$$\boxed{\begin{aligned}\mathrm{Adv}^{\text{UF-CMA}}_{\mathsf{CoreFalcon}^+}(\mathcal{A}) \le{}& \delta_{\mathrm{cap}} + \delta_{\mathrm{salt}} \\ &+ \exp\left(\frac{(a-1)C_s W_*^2}{8}\right)\left(\mathrm{Adv}^{(Q_H+1)\text{-R-ISIS}}(\mathcal{B}_\mathcal{A})\right)^{1-1/a},\end{aligned}} \tag{31}$$

where

$$\delta_{\mathrm{cap}} = \mathbb{P}[\mathrm{Bin}(C_s, p_*) < Q_s], \tag{32}$$

$$\delta_{\mathrm{salt}} = \min\left\{1, \frac{C_s Q_H + C_s(C_s-1)/2}{S}\right\}. \tag{33}$$

For $Q_s = 0$, interpret $\delta_{\mathrm{cap}} = 0$. The solver runs $\mathcal{A}$ once and makes at most C_s ambient Gaussian draws from $D_{\mathbb{Z}^{2n},s}$, besides polynomial-time arithmetic and oracle-table management. This cost statement uses the exact-sampling model, not a verified bit-complexity bound for a deployed implementation.

Proof. We describe every distributional change.

Cap the raw attempts. Modify the real experiment to abort before executing attempt $C_s + 1$. On any history, a raw attempt succeeds with conditional probability at least p_*. Such adaptive Bernoulli successes stochastically dominate independent Bernoulli trials of parameter p_*: realize each success indicator using a fresh uniform number and its history-dependent success probability, and compare to threshold p_*. Therefore failure to complete at most Q_s signatures in C_s trials has probability at most (32). Artificial trials can pad an experiment that completes earlier.

Abort on a reused signing input. Further abort whenever a raw signing attempt selects an input (h, m, r) already in the oracle table. Before raw attempt j, that table contains at most $Q_H + j - 1$ entries. Since its salt is uniform and independent of the past, the conditional collision probability is at most $(Q_H + j - 1)/S$. Summing over $j \le C_s$ proves (33). This counts rejected attempts as well as accepted ones. Denote the resulting real experiment by P.

Replace a fresh syndrome and preimage jointly. In a new experiment Q, every fresh raw signing input is handled by sampling $x \leftarrow D_{\mathbb{Z}^{2n},s}$ and setting the oracle-table value to $A_h x$. Apply the same norm test and keep the programmed entry even if the attempt is rejected. All direct oracle queries at previously unset inputs still receive independent uniform values. Salt selection, stopping, abort rules, and the adversary are unchanged.

In P, a fresh signing input produces exactly the joint law P_B of Section D.4; in Q, it produces exactly Q from that section. Conditional on each common history and on the key, Theorem D.2 gives log-likelihood range at most W_*. Include the secret basis in the mathematical extended transcript of both experiments; it has the same distribution and is not given to the adversary or the solver. There are at most C_s differing operations. Lemma D.5.1 gives

$$P[\mathcal{A} \text{ wins}] \le e^{(a-1)C_s W_*^2/8} Q[\mathcal{A} \text{ wins}]^{1-1/a}.$$

Solve multi-target R-ISIS. Given $(h, c_1, \ldots, c_{Q_H+1})$, the solver implements Q without knowing a trapdoor. Each fresh direct adversarial oracle query is answered using the next unused challenge syndrome; repeated inputs use the table. Raw signing attempts use ambient Gaussian draws and program fresh inputs as above. Abort rules are identical. Because unused challenge values remain independent uniform samples, this is an exact simulation of Q.

At the end, evaluate the oracle on the forgery input if necessary, using one further challenge value. On a successful UF-CMA forgery, its message was never signed. Its input therefore cannot be a signing-programmed entry, including an entry from a rejected attempt. It is associated with one of the supplied challenge syndromes. The forged vector itself solves that target R-ISIS instance. Thus $Q[\mathcal{A}\text{ wins}]$ equals the solver's success probability when it reports success exactly in this case. Combining the two abort bounds and the Rényi comparison proves (31). □

Relation to the earlier two-transition bound. Theorem 1 of Jia et al. (2026) compares sampler and syndrome distributions separately. Theorem D.5 instead uses the cancellation in (25); direct adversarial oracle queries remain uniform and are supplied by the R-ISIS challenge. Hence there is one order a and one exponent $1 - 1/a$, with no separate r_p and r_u in this core reduction. Our salt term is derived directly, not silently substituted with the sharper-looking term in another proof. The parameter p_* is a *lower* bound on conditional acceptance, as required for binomial domination.

If the solver advantage is at most $2^{-\lambda}$ at the stated resource bound, Corollary D.4 gives

$$\mathrm{Adv}^{\text{UF-CMA}}_{\mathsf{CoreFalcon}^+}(\mathcal{A}) \le \delta_{\mathrm{cap}} + \delta_{\mathrm{salt}} + 2^{-(\max\{0,\sqrt{\lambda}-\sqrt{C_s W_*^2/(8\log 2)}\})^2}.$$

This formula separates the analytical improvement from the computational hardness estimate.

D.6.3 Keys and implementation scope

The pointwise sampling theorems hold for each fixed basis. The cryptographic theorem needs a uniform certificate over its actual key distribution (or a separately proved distributional substitute). The maxima over 1,000 sampled bases reported in Jia et al. (2026), Table 6 and Remark 4, do not alone provide such a uniform bound. A key-generation check using (29) is one way to impose it, but then the hardness experiment must use that checked key distribution or explicitly account for conditioning.

The scheme here returns the full integer vector. More generally, the same reduction applies when an accepted (r, x) is passed through a deterministic encoding and every valid encoded forgery allows public recovery of a salt and a short vector satisfying the syndrome equation. The simulator applies the encoding to its Gaussian proposals; extraction uses that public recovery map. Injectivity is unnecessary for plain unforgeability. If encoding can fail, include its success in the acceptance predicate and re-establish p_* for that predicate. These properties must be proved for a particular format, not assumed from its name.

Finite-precision sampling, support truncation, compression failures, side channels, and the relation to a particular FFO implementation are separate proof obligations. No QROM theorem is claimed.

D.7 Explicit certificates from one-dimensional Gaussians

Put, for $u > 0$,

$$\Theta_+(u) = \sum_{k\in\mathbb{Z}} e^{-\pi u^2 k^2}, \qquad \Theta_-(u) = \sum_{k\in\mathbb{Z}} (-1)^k e^{-\pi u^2 k^2}.$$

The following classical identities give exact, computable intervals without assuming a small smoothing error.

Proposition D.6 (Theta certificate)**.** *For every $u > 0$ and $d \in \mathbb{R}$,*

$$0 < u\Theta_-(u) \le Z_u(d) \le u\Theta_+(u).$$

The upper extreme is attained at integer centers and the lower at half-integer centers. Therefore Theorems D.2 and D.5 can use

$$W_\theta(B, s) = \sum_i \log \frac{\Theta_+(s/g_i)}{\Theta_-(s/g_i)}. \tag{34}$$

Proof. Poisson summation for the Gaussian gives

$$\frac{Z_u(d)}{u} = \sum_{k\in\mathbb{Z}} r^{k^2} e^{2\pi i k d}, \qquad r = e^{-\pi u^2} \in (0,1).$$

The Jacobi product formula NIST Digital Library of Mathematical Functions rewrites the right side as

$$\prod_{j\geq 1}(1-r^{2j})\big(1+2r^{2j-1}\cos(2\pi d)+r^{4j-2}\big).$$

Each center-dependent factor is positive and increasing in $\cos(2\pi d)$. The product is minimized at $d = 1/2$ and maximized at $d = 0$; positivity follows also from the strictly positive Gaussian sum. Substitution gives the two theta series. □

If $2\sum_{k\geq 1} e^{-\pi s_i^2 k^2} \leq \varepsilon_i < 1$, the same Fourier series gives

$$s_i(1-\varepsilon_i) \leq Z_{s_i}(d) \leq s_i(1+\varepsilon_i),$$

and hence $W_\theta \leq \sum_i \log((1+\varepsilon_i)/(1-\varepsilon_i))$. Thus the theta certificate can only sharpen that particular interval certificate.

Proposition D.7 (Product bound for dual Gaussian mass)**.** *Let* $\Lambda^* = \{y : \langle y, x\rangle \in \mathbb{Z}$ *for every* $x \in \Lambda\}$. *Then*

$$\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \prod_i \Theta_+(s/g_i) - 1. \tag{35}$$

If $T = 2\sum_i e^{-\pi(s/g_i)^2} < 1$, *then*

$$\prod_i \Theta_+(s/g_i) - 1 \leq \frac{T}{1-T}. \tag{36}$$

Proof. The reversed dual basis has Gram–Schmidt lengths $1/g_i$, in reverse order. Sum the Gaussian successively in its integer coordinates. Each conditional one-dimensional sum is bounded by its centered sum, using Proposition D.6. For Gaussian width $1/s$, those centered sums equal $\Theta_+(s/g_i)$. This proves (35) after removing the zero vector.

For the comparison, write $r_i = e^{-\pi(s/g_i)^2}$. When $2r_i < 1$,

$$\Theta_+(s/g_i) = 1 + 2\sum_{k\geq 1} r_i^{k^2} \leq \frac{1}{1-2r_i}.$$

For example, first use $k^2 \geq k$ and then $(1+r_i)/(1-r_i) \leq 1/(1-2r_i)$. Finally, $\prod_i(1-2r_i) \geq 1 - 2\sum_i r_i = 1 - T$. □

Theorem 3 of Jia et al. (2026) uses $T/(1-T)$ after rewriting its profile parameters as the above T. Thus (35) is a pointwise strengthening of that displayed bound. It remains meaningful as a mass bound even when $T \geq 1$, although a smoothing application still needs the resulting mass to be small enough.

Corollary D.8 (A uniform certificate at the original global width)**.** *Suppose* $0 < \varepsilon < 1$, $g_i \leq g_{\max}$ *for every* i, *and*

$$s \geq g_{\max}\sqrt{\frac{\log(2m(1+1/\varepsilon))}{\pi}}.$$

Then

$$W_\theta(B,s) \leq m\log\frac{1+\varepsilon/m}{1-\varepsilon/m}, \qquad \rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \varepsilon.$$

Both bounds are uniform over every basis satisfying the stated Gram–Schmidt bound, not empirical statements about sampled keys.

Proof. Let $r_i = e^{-\pi(s/g_i)^2} \le \varepsilon/(2m(1+\varepsilon))$. Then

$$2\sum_{k\ge 1} r_i^{k^2} \le \frac{2r_i}{1-r_i} \le \frac{\varepsilon}{m(1+\varepsilon)-\varepsilon/2} \le \frac{\varepsilon}{m}.$$

Use the Fourier interval bound above with $\varepsilon_i = \varepsilon/m$. Also, $T = 2\sum_i r_i \le \varepsilon/(1+\varepsilon)$, so Proposition D.7 gives the dual-mass bound. □

For the global width specification in Jia et al. (2026), take $g_{\max} = 1.17\sqrt{q}$. This provides a fully uniform instantiation of the core theorem whenever its other hypotheses hold. Actual-profile certificates can be substantially smaller, but their uniform validity requires the separate key-distribution justification discussed in Section D.6.

For reproducible numerical evaluation, if $r = e^{-\pi u^2}$, the tail after index K satisfies

$$2\sum_{k=K+1}^{\infty} r^{k^2} \le \frac{2r^{(K+1)^2}}{1-r^{2K+3}}.$$

This follows by bounding the ratio of successive terms. A certified implementation must combine such analytic remainder bounds with outward-rounded arithmetic and a verified connection between the supplied GSO lengths and the actual basis.

D.8 A SHARPER DISCRETE-GAUSSIAN DISTANCE BOUND

This section improves the fixed-vector lemma used in the strong-unforgeability discussion of Jia et al. (2026). It is independent of the UF-CMA reduction above.

For a full-rank lattice $\Lambda \subset \mathbb{R}^m$, assume

$$\rho_{1/s}(\Lambda^* \setminus \{0\}) \le \varepsilon < 1, \qquad \varepsilon \ge 0. \tag{37}$$

Poisson summation gives, uniformly in t,

$$\rho_s(\Lambda + t) = \frac{s^m}{\det \Lambda} \sum_{y\in\Lambda^*} e^{-\pi s^2\|y\|^2} e^{2\pi i\langle y,t\rangle}. \tag{38}$$

Pairing conjugate terms and bounding the nonzero terms in absolute value yields

$$\frac{s^m}{\det\Lambda}(1-\varepsilon) \le \rho_s(\Lambda+t) \le \frac{s^m}{\det\Lambda}(1+\varepsilon).$$

At any larger width, the same ε remains a valid bound on the dual mass.

Theorem D.9 (Direct squared-distance Chernoff bound)**.** *Let $V \leftarrow D_{\Lambda+t,s}$, set $\sigma = s/\sqrt{2\pi}$ and $\beta = \tau\sigma\sqrt{m}$ with $\tau > 0$, and assume* (37)*. For every fixed $w \in \mathbb{R}^m$ with $\|w\| \le \beta$, every $c > 0$, and every $u \in (0,1)$,*

$$\mathbb{P}[\|V - w\| > c\beta] \le \frac{1+\varepsilon}{1-\varepsilon} e^{-mJ_{\tau,c}(u)}, \tag{39}$$

where

$$J_{\tau,c}(u) = \frac{1}{2}\left[uc^2\tau^2 + \log(1-u) - \frac{u\tau^2}{1-u}\right].$$

If $c > \sqrt{1+\tau^{-2}}$, the maximum of $J_{\tau,c}$ is positive and is attained at

$$z_* := \frac{1+\sqrt{1+4c^2\tau^4}}{2c^2\tau^2}, \qquad u_* := 1 - z_* \in (0,1). \tag{40}$$

Writing $I_\tau(c) = J_{\tau,c}(u_*)$, we obtain

$$\boxed{\mathbb{P}[\|V - w\| \le c\beta] \ge \max\left\{0,\, 1 - \frac{1+\varepsilon}{1-\varepsilon} e^{-mI_\tau(c)}\right\}.} \tag{41}$$

Proof. Complete the square inside the exact Gaussian sum:

$$\mathbb{E}e^{u\|V-w\|^2/(2\sigma^2)} = \exp\left(\frac{u\|w\|^2}{2\sigma^2(1-u)}\right)\frac{\rho_{s/\sqrt{1-u}}(\Lambda+t+\frac{u}{1-u}w)}{\rho_s(\Lambda+t)}$$
$$\leq \frac{1+\varepsilon}{1-\varepsilon}(1-u)^{-m/2}\exp\left(\frac{u\tau^2 m}{2(1-u)}\right).$$

The inequality follows from (38), since the numerator width is larger than s. Exponential Markov at threshold $c^2\beta^2$ proves (39).

Moreover,

$$J'_{\tau,c}(u) = \frac{1}{2}\left[c^2\tau^2 - (1-u)^{-1} - \tau^2(1-u)^{-2}\right],$$

and

$$J''_{\tau,c}(u) = -\frac{1}{2}\left[(1-u)^{-2} + 2\tau^2(1-u)^{-3}\right] < 0.$$

The hypothesis makes $J'(0) > 0$, while $J'(u) \to -\infty$ as $u \to 1$. Its unique zero is equivalent to $c^2\tau^2 z^2 - z - \tau^2 = 0$, giving (40). Since $J(0) = 0$, the interior maximum is positive. The complement gives (41). □

Unlike Lemma 9 of Jia et al. (2026), this proof does not split the bad event into a large-norm event and a large-inner-product event. It bounds the squared-distance moment directly. The condition $\tau^2(c^2-1) > 1$ is the same threshold already present in their lemma, not a newly relaxed hypothesis. Our statement additionally does not require $w \in \Lambda + t$.

Dimension	ε	Prior lower bound	New bound	$-\log_2 p_{\text{new}}$
$m = 1024$	$2^{-35.5}$	0.2627	0.9562590813	0.0645266
$m = 2048$	2^{-36}	0.867	0.9980867320	0.00276291

Table 5: Fixed-vector distance probabilities at $\tau = 1.1$ and $c = \sqrt{2}$. Prior numbers are from Jia et al. (2026), Remark 5, p. 24. New entries are high-precision numerical evaluations of the exact bound (41), rounded for display; they are not simulations or interval-certified decimal inequalities. The last column is the logarithmic penalty of this probability factor alone, not a complete security-loss calculation.

For example, choosing $c = 1.39$ instead gives numerical lower-bound values about 0.693853 and 0.906274 in the same two dimensions. Radius and probability can therefore be optimized together. A reduction using this smaller radius must still establish the conditional sampling requirement below and use a corresponding hardness estimate.

Corollary D.10 (A sufficient acceptance certificate)**.** *Under* (37)*, if* $\tau > 1$ *and* $\beta = \tau\sigma\sqrt{m}$*, then, uniformly over cosets,*

$$G_C[\|x\| > \beta] \leq \frac{1+\varepsilon}{1-\varepsilon}\exp\left[-\frac{m}{2}(\tau^2 - 1 - \log\tau^2)\right].$$

Consequently a sufficient lower bound for the uncorrected Klein–GPV acceptance probability is

$$\max\left\{0, e^{-W}\left(1 - \frac{1+\varepsilon}{1-\varepsilon}e^{-\frac{m}{2}(\tau^2-1-\log\tau^2)}\right)\right\}. \tag{42}$$

If positive and uniform over the key distribution, this can serve as p_* *in Theorem D.5.*

Proof. Use the same moment calculation with $w = 0$ and choose $u = 1 - \tau^{-2}$. Also, $K_{B,C}(x)/G_C(x) = \rho_s(C)/Z_B(x,C) \geq L_B/U_B = e^{-W}$, by the proof of Theorem D.2. Sum over the acceptance event. □

D.9 THE PRECISE CONDITIONAL STATEMENT NEEDED FOR EXTRACTION

The distance theorem is not an independence theorem about an adaptive forgery. The following is a sufficient, explicit interface.

Proposition D.11 (Hidden-preimage extraction)**.** *Let T be a countable transcript, F an event determined by T, and $w = w(T)$. Suppose that on each successful transcript t, the vector $w(t)$ belongs to a specified coset C_t and has norm at most β, while*

$$\mathcal{L}(V \mid T = t) = D_{C_t,s}.$$

Assume the distance theorem supplies a uniform lower bound p and that every atom of these conditional Gaussian laws has mass at most μ. Then

$$\mathbb{P}[F \text{ and } 0 < \|V - w(T)\| \le c\beta] \ge \max\{0, p - \mu\}\, \mathbb{P}[F]. \tag{43}$$

The extracted vector lies in the corresponding lattice.

Proof. Condition on $T = t$ with F true. The fixed-vector theorem gives probability at least p for the distance condition, and equality $V = w(t)$ has probability at most μ. Subtract and average over successful transcripts. Both vectors are in C_t, so their difference is in the lattice. □

Under (37), one available atom bound is $\mu \le \min\{1, \det\Lambda/(s^m(1-\varepsilon))\}$, since the unnormalized Gaussian weight is at most one. If the conditional law is instead a norm-truncated Gaussian with conditioning probability at least $r_0 > 0$, the corresponding extraction lower bound becomes at least

$$\max\{0, 1 - (1 - p + \mu)/r_0\}$$

per successful transcript; this follows by dividing the probability of the excluded bad events by the conditioning probability.

The final factorization in Appendix A of Jia et al. (2026), p. 32, does not follow from a fixed-vector distance bound alone. In particular, when a preimage has already been revealed as a signature, a forgery can depend on it. Proposition D.11 states a sufficient condition rather than assuming that dependence away. We therefore do not replace the SUF-CMA security numbers in Jia et al. (2026) with new full-scheme claims on the basis of Table 5.

D.10 CONCLUSION AND VERIFICATION STATUS

The main reusable statement is the single-certificate theorem: the normalizer log-range W gives $\log R_a \le aW^2/8$ for both a fixed-coset sampler comparison and a fresh joint syndrome–preimage comparison, while bounding rejection-correction overhead by e^W. It sharpens the displayed sampler coefficient in Jia et al. (2026), preserves its basis-specific advantage, and supports the single-transition core UF-CMA theorem proved here.

The direct distance theorem separately improves the numerical probabilities in their Lemma 9/Remark 5 analysis. Its use for adaptive strong-unforgeability extraction requires the conditional-law statement in Section D.9; no independent-preimage hypothesis is hidden in the UF-CMA theorem.

These are mathematical proofs in an explicitly idealized model. They are not Lean-checked proofs, verified implementation claims, new lattice-hardness estimates, or an audit of the complete Falcon standard. The accompanying script `appendix_falcon_normalizer_verify.py` reproduces the distance-bound values and evaluates profile certificates, but uses high-precision rather than interval-certified arithmetic.

Algorithm 10: Falcon key recovery from sign leakage

Input: Public key h; m sign-data records $(\boldsymbol{s}_j, \boldsymbol{c}_j, \mathrm{sign}(\boldsymbol{z}_j))$ for $j = 1, \ldots, m$; allowed coefficient values $\mathcal{K}$.
Output: A complete key (f, g, F, G), or $\perp$

// Fit the dimension-halved model and determine its integer scale
$\boldsymbol{u} \leftarrow \arg\min_{\boldsymbol{v} \in \mathbb{R}^n} L_{\mathrm{half}}(\boldsymbol{v})$
$\mathcal{S} \leftarrow \{u_j/k > 0 : j \in [n],\ k \in \mathcal{K} \setminus \{0\}\}$
$\alpha \leftarrow \arg\min_{\sigma \in \mathcal{S}} \|\boldsymbol{u}/\sigma - \mathrm{round}(\boldsymbol{u}/\sigma)\|_2$
$\boldsymbol{f}_0 \leftarrow \mathrm{round}(\boldsymbol{u}/\alpha)$

// Fit the observed signs, using $\boldsymbol{f}_0$ to anchor the scale
$\lambda_f \leftarrow 10^{-4}$; $\lambda_g \leftarrow 10^{-5}$
$(\widetilde{\boldsymbol{f}}, \widetilde{\boldsymbol{g}}) \leftarrow \arg\min_{\boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^n} L_{\mathrm{full}}(\boldsymbol{v}, \boldsymbol{w}; \boldsymbol{f}_0)$
$\bar{\boldsymbol{f}} \leftarrow \mathrm{round}(\widetilde{\boldsymbol{f}})$
Try $f = \bar{\boldsymbol{f}}$: derive $g = hf \bmod q$, solve $fG - gF = q$, return the key if it passes the public checks

// Prioritize uncertain rounding decisions
$r_j \leftarrow |\widetilde{f}_j - \bar{f}_j|$ and $\boldsymbol{d}_j \leftarrow \mathrm{sign}(\widetilde{f}_j - \bar{f}_j)\boldsymbol{e}_j$ for $j \in [n]$
$\pi \leftarrow$ coordinates sorted by decreasing r_j

// Meet in the middle over uncertain rounding decisions
Take the highest-ranked coordinates in π and split them into two balanced groups
In each group, enumerate sparse edits of the form $\boldsymbol{\delta}_A = \sum_{j \in S_A} \boldsymbol{d}_j$, where $A \in \{L, R\}$ and S_A is a small subset of that group
Index each left edit by selected coefficients of $h\boldsymbol{\delta}_L \bmod q$
foreach *right edit* $\boldsymbol{\delta}_R$ **do**
  $\boldsymbol{v}_R \leftarrow h(\bar{\boldsymbol{f}} + \boldsymbol{\delta}_R) \bmod q$
  Retrieve left edits whose indexed coefficients nearly cancel the corresponding coefficients of $\boldsymbol{v}_R$ modulo q
  // The centered sum is small on those coefficients; check the full key
  foreach *retrieved* $\boldsymbol{\delta}_L$ **do**
    $\boldsymbol{f} \leftarrow \bar{\boldsymbol{f}} + \boldsymbol{\delta}_L + \boldsymbol{\delta}_R$
    $\boldsymbol{g} \leftarrow hf \bmod q$
    if *f is invertible and $(\boldsymbol{f}, \boldsymbol{g})$ meets the Falcon norm bound* **then**
      Solve $fG - gF = q$ over $\mathbb{Z}[X]/(X^n + 1)$
      if *a solution exists, its public key is h, and a fresh signature verifies* **then**
        return (f, g, F, G)
return $\perp$

E Detailed Algorithm for Falcon Key Recovery

The following pseudocode accompanies the Falcon case study. The loss functions and regularization parameters are defined in the main text.