

Beyond NIST Levels: Exploring Non-Standard Parameter Sets for ML-DSA Security

Mr. Jitendra Kumar

Dr. Balaji Rajendran

Dr. S. D. Sudarsan

RISE, C-DAC Bangalore
Bengaluru, India

PKIA 2026

Motivation and Research Question

- ▶ Quantum computing threatens classical asymmetric cryptography such as RSA and ECC.
- ▶ NIST standardized ML-DSA in three discrete security categories: ML-DSA-44, ML-DSA-65 and ML-DSA-87.
- ▶ PKI must support both resource-constrained IoT devices and high-assurance, long-lived systems.
- ▶ The paper therefore studies **non-standard parameter sets** to expose the security-performance trade-off.

Core question

How far can ML-DSA parameters be changed to reduce computational and payload overhead before the resulting security becomes unacceptable?

Profiles studied

- ▶ Lite / Ultra-Lite
- ▶ High-Noise
- ▶ Ultra-High Security

ML-DSA Background

Algebraic setting

$$R_q = \mathbb{Z}_q[x]/(x^n + 1)$$

$$n = 256, \quad q = 8,380,417$$

The public-key relation is

$$t = As_1 + s_2 \pmod{q}$$

where $A \in R_q^{k \times l}$.

- ▶ Security derives from Module Learning With Errors (MLWE).
- ▶ Signature integrity is related to Module Short Integer Solution (MSIS).

NIST standardized profiles

Variant	Category	$k \times l$	Target
ML-DSA-44	2	4×4	$\sim$ AES-128
ML-DSA-65	3	6×5	$\sim$ AES-192
ML-DSA-87	5	8×7	$\sim$ AES-256

Key point

The module rank and error distribution shape the underlying lattice and therefore the estimated attack hardness.

BKZ Reduction and Block Size β

BKZ: Block Korkine–Zolotarev

BKZ is a lattice-reduction algorithm used to obtain a shorter and more structured lattice basis. The reduction strength is controlled by the **block size** β .

Role in Primal-uSVP

For the MLWE security evaluation, the attack flow is:

MLWE instance $\rightarrow$ Primal lattice $\rightarrow$ uSVP $\rightarrow$ BKZ $\rightarrow$ Attack-cost estimate.

Block Size β

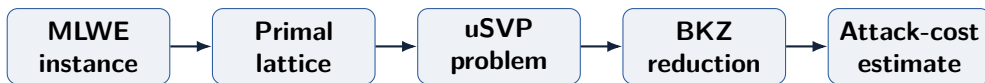
$$\beta = \text{BKZ block size}$$

A larger β represents stronger lattice reduction and generally requires greater computational effort. The Lattice Estimator determines the appropriate β when estimating the primal-uSVP attack cost.

Interpretation

The β indicates the lattice-reduction strength also reports security bits and attack cost.

From MLWE to a Primal uSVP Attack



Lattice mapping

Polynomial unrolling converts the $k \times l$ matrix of degree-256 polynomials into a

$$(256k) \times (256l)$$

integer matrix.

Block Korkine–Zolotarev (BKZ) hardness

The block size β controls reduction strength. Larger β gives stronger reduction but exponentially higher computational complexity.

Parameters and Experimental Method

Independent variables

- ▶ k, l : module rank
- ▶ η : secret/error distribution bound

Fixed parameters

$$n = 256, \quad q = 8,380,417$$

The standardized γ_1, γ_2 values are retained.

Evaluation toolchain

- 1 Instantiate bounded MLWE parameters.
- 2 Run primal-uSVP estimation with the Lattice Estimator.
- 3 Extract security bits and BKZ block-size threshold β .
- 4 Estimate verification cost and payload footprint.
- 5 Compare security density and computational hardness.

Empirical Constant σ : Payload Size KB/Rank Element

Why a linear model?

The public-key high-bit vector has dimension k , while the signature response vector has dimension l . Therefore the combined structural payload is approximated as

$$S = (k + l)\sigma.$$

Each module-vector element contains 256 coefficients.

ML-DSA-44 example

For $k = l = 4$, $k + l = 8$:

$$1312\text{B} \times 8 = 10496\text{bits}$$

$$2420\text{B} \times 8 = 19360\text{bits}$$

$$10496 + 19360 = 29856\text{bits}$$

$$\frac{29856}{8} = 3732 \text{ bits/element.}$$

Empirical Constant σ : Cross-Variant Average

Cross-variant average

Averaging across the ML-DSA variants gives

3686 bits/element.

Convert to KB:

$$\sigma \approx \frac{3686}{8 \times 1024} \approx 0.45 \text{ KB.}$$

Empirical constant

$$\sigma = 0.45 \text{ KB/element}$$

Used to estimate the key/signature footprint of custom profiles.

Empirical Constant ω : Verify/Gen-Cycle Anchor (Cycle/Rank Element)

Measured verification medians and normalized cost

	Dilithium2 $4 \times 4 = 16$	Dilithium3 $6 \times 5 = 30$	Dilithium5 $8 \times 7 = 56$
Verify median (cycles)	118,412	179,424	279,936
cycles/ $(k * l)$	$118412/16 = 7400.75$	$179424/30 = 5980.80$	$279936/56 = 4998.85$
Generation median (cycles)	124,031	256,403	298,050
cycles/ $(k * l)$	$124031/16 = 7751.93$	$256403/30 = 8546.76$	$298050/56 = 5322.32$

Why 7400?

The paper therefore chooses a conservative baseline as below.

$$\omega = 7400$$

Use in the experiment

The verification-cost estimate is

$$C = \frac{(k * l)\omega}{1000}$$

where C is expressed in k-cycles.

Algorithm : Security and Performance Evaluation

Inputs

$$\mathcal{A} = \{(k_i, l_i, \eta_i)\}, \quad q = 8,380,417, \quad n = 256, \quad \omega = 7400, \quad \sigma = 0.45 \text{ KB}$$

- 1 Initialize evaluation registry: $R \leftarrow \emptyset$.
- 2 For each profile $(k, l, \eta) \in \mathcal{A}$, define $X_s, X_e \leftarrow U(-\eta, \eta)$.
- 3 Instantiate the bounded Module-LWE system: $P \leftarrow (n * l, q, X_s, X_e, n * k)$.
- 4 Execute progressive primal-uSVP estimation: $M_{\text{attack}} \leftarrow \text{primal_usvp}(P)$.
- 5 Extract security and BKZ block size:

$$b_{\text{sec}} = \log_2(M_{\text{attack}}[\text{rop}]), \quad \beta = M_{\text{attack}}[\text{beta}].$$

- 6 Compute performance:

$$C = \frac{(k l) \omega}{1000}, \quad S = (k + l) \sigma, \quad E = \frac{b_{\text{sec}}}{S}.$$

- 7 Store (label, $k \times l$, b_{sec} , β , C , S , E) in R and generate the summary plots/tables.

Python Lattice Estimator

SageMath Lattice Estimator by **Martin R. Albrecht**, **Rachel Player**, and **Sam Scott** evaluates the bounded Module-LWE instance with a **primal-uSVP** model and obtains the attack cost and BKZ block size β .

Experimental Profiles and Results

Variant	$k \times l$	η	Sec. bits	β	Cycles	Size (KB)	Efficiency
ML-DSA-44 (Std)	4×4	2	150.19	424	118,400	3.60	41.72
ML-DSA-65 (Std)	6×5	4	190.00	566	222,000	4.95	38.38
ML-DSA-87 (Std)	8×7	2	273.43	863	414,400	6.75	40.51
Ultra-Lite-22	2×2	2	76.24	160	29,600	1.80	42.36
Lite-33	3×3	3	112.09	288	66,600	2.70	41.51
High-Noise-55	5×5	4	206.45	625	185,000	4.50	45.88
Ultra-99	9×9	9	360.67	1173	599,400	8.10	44.53

Lowest footprint

Ultra-Lite-22
29.6k cycles, 1.80 KB

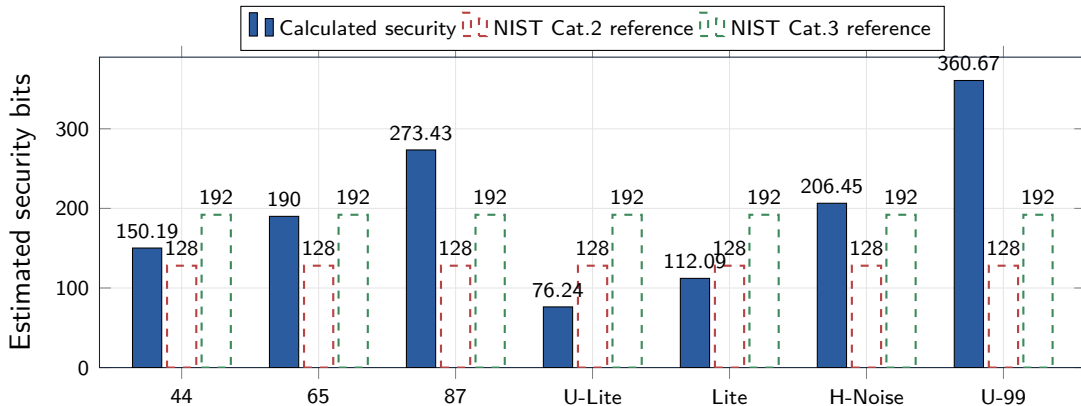
Standard reference

ML-DSA-44
150.19 bits, 118.4k cycles

Highest hardness

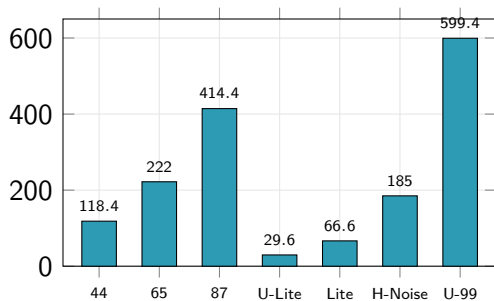
Ultra-99
360.67 bits, 599.4k cycles

Security Hardness Comparison

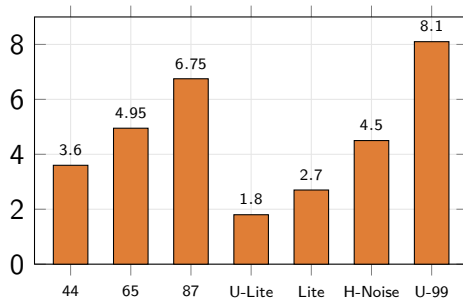


Latency and Payload Footprint

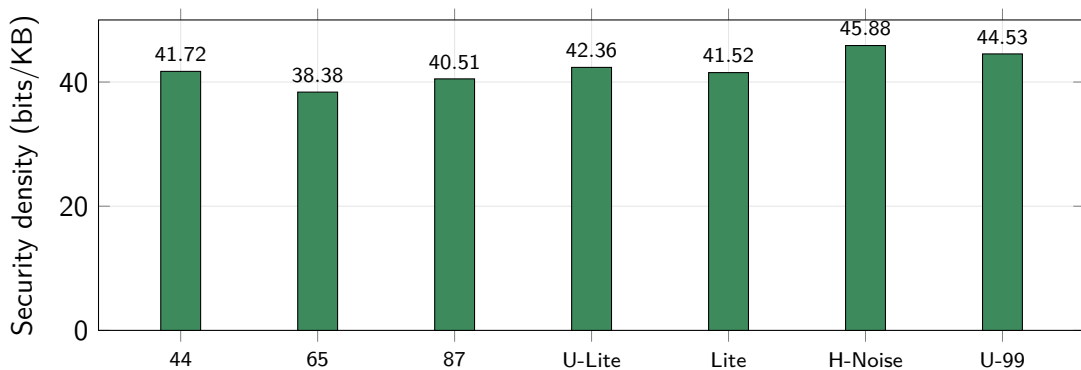
Verification cost (k-cycles)



Estimated payload (KB)



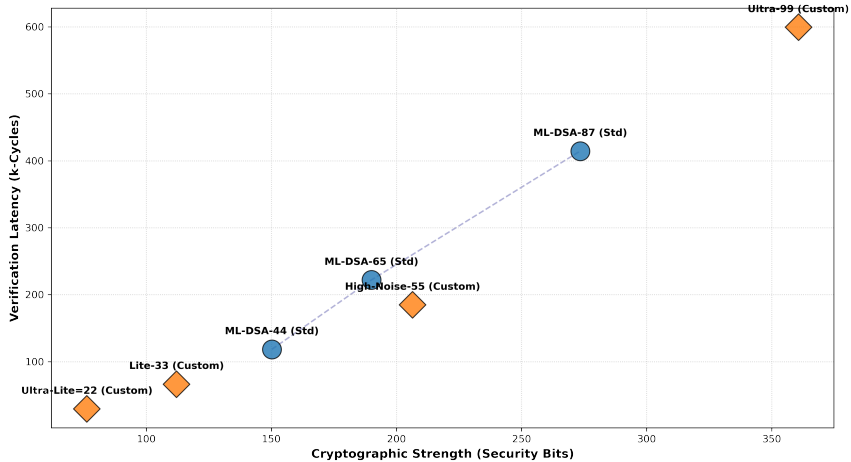
Security Density



The investigated profiles range from **38.38 to 45.88 bits/KB**. As rank increases, security gains are smaller.

Security Performance

Computational Cost vs. Cryptographic Hardness



Security Floor and Parameter Selection

Lite-33: below the floor

$$k = l = 3, \quad 112.09 \text{ bits}$$

The smaller footprint and lower cycle count are attractive, but the paper identifies the drop below the NIST Category 2 region as unacceptable for the post-quantum setting.

Implication

The experiment provides empirical support for retaining the 4×4 rank of ML-DSA-44 rather than optimizing below it when Category 2-level security is required.

Rank vs. noise

The comparison indicates that increasing matrix rank provides a better trade-off against BKZ attacks than relying only on a larger error/noise bound.

Architectural scaling

Verification cost grows relatively linearly with the tested rank scale, while estimated security hardness grows non-linearly.

Key Findings

- 1 **Standardization baseline:** ML-DSA-44, ML-DSA-65 and ML-DSA-87 provide robust security/performance trade-offs.
- 2 **Matrix rank is critical:** Reducing the architecture below 4×4 can reduce footprint but also produces a nonlinear degradation in estimated security.
- 3 **High-noise alternative:** High-Noise-55 reaches 206.45 estimated security bits with 185,000 cycles and 4.50 KB payload.
- 4 **Ultra scaling:** Ultra-99 reaches 360.67 estimated security bits, but requires 599,400 cycles and 8.10 KB.
- 5 **Main conclusion:** Any performance improvement beyond ML-DSA-44 in the direction of smaller parameters comes with a reduction in estimated security strength.

Practical Takeaway

For constrained PKI / IoT deployments

Choose parameters only after defining an explicit acceptable security floor. Smaller profiles reduce communication and computational overhead, but the security loss must be treated as a cryptographic design consequence.

For general-purpose deployments

The standardized ML-DSA profiles provide established security categories and a useful baseline for interoperability and deployment.

For high-assurance systems

Larger custom configurations can provide substantially greater estimated hardness, but the computational and payload overhead rises accordingly.

Key Take Away

Parameter tuning is a security decision—not only a performance decision.

References



E. W. Postlethwaite and F. Virdia, "On the Success Probability of Solving Unique SVP via BKZ," in *Public-Key Cryptography – PKC 2021*, LNCS, vol. 12710, Springer, 2021, pp. 68–98.



C. P. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," *Mathematical Programming*, vol. 66, no. 1, pp. 181–199, 1994.



National Institute of Standards and Technology, "Module-Lattice-Based Digital Signature Standard," Federal Information Processing Standards Publication, FIPS 204, 2024.



T. Laarhoven, M. Mosca, and J. van de Pol, "Finding shortest lattice vectors faster using quantum search," *Designs, Codes and Cryptography*, vol. 77, pp. 375–400, 2015.



S. Bai and S. D. Galbraith, "Lattice Decoding Attacks on Binary LWE," in *Information Security and Privacy – ACISP 2014*, LNCS, vol. 8544, Springer, 2014.



M. R. Albrecht, R. Player, and S. Scott, "On the Concrete Hardness of Learning with Errors," *Journal of Mathematical Cryptology*, vol. 9, no. 3, pp. 169–203, 2015.

References (continued)



M. Grassl, B. Langenberg, R. Roetteler, and R. Steinwandt, "Applying Grover's Algorithm to AES: Quantum Resource Estimates," in *Post-Quantum Cryptography – PQCrypto 2016*, LNCS, vol. 9606, Springer, 2016.



S. Bai, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Dilithium: Algorithm Specifications and Supporting Documentation (Version 3.1)," NIST Post-Quantum Cryptography Standardization Project, Round 3 Submission, 2021.



W. Stein et al., "Software for Algebra and Geometry Experimentation (SageMath): Open Source Mathematics Software," Python Package Index (PyPI), v10.7, 2025.



M. R. Albrecht, "The Lattice Estimator," GitHub Repository, 2026.



National Institute of Standards and Technology, "Security (Evaluation Criteria)," Post-Quantum Cryptography Standardization Project, NIST, 2017.



Thank You

Questions?