



From Integers to Lattices: Number-Theoretic Foundations of the Post-Quantum Transition in Public-Key Cryptography

Tripti Gautam, Research Scholar¹,

Mahakaushal University, Jabalpur, (M.P.)
triptigautam002@gmail.com¹

Dr. Narendra Bahadur Singh, Professor²,

Mahakaushal University, Jabalpur, (M.P.)²

Abstract- Factoring, discrete logarithms and their elliptic-curve analogues are number-theoretic problems that a large quantum computer can solve in polynomial time. This paper explains why, describes the lattice-based replacements now standardised by NIST, and shows that number theory still does much of the work inside them. Three computational experiments support the argument. First, the classical reduction inside Shor's algorithm was applied to 300 semiprime moduli: a random base yielded a nontrivial factor 74.3% of the time on average, and never less than 50% for any modulus, matching theory. Second, a toy learning-with-errors (LWE) instance modulo $q = 3329$ was solved exactly by Gaussian elimination without noise, but in none of 300 trials per dimension once errors as small as ± 1 were added. Third, a number-theoretic transform over $\mathbb{Z}_{3329}$, using the 256th root of unity 17, reproduced schoolbook polynomial multiplication exactly while needing 18 times fewer modular multiplications (3,584 versus 65,536). The paper also quantifies the price of migration: an ML-KEM-768 exchange transmits 2,272 bytes against 64 for X25519.

Keywords- post-quantum cryptography, Shor's algorithm, learning with errors, lattice cryptography, number-theoretic transform, ML-KEM, ML-DSA.

I. Introduction

The security of RSA, Diffie–Hellman and elliptic-curve cryptography depends on the assumption that integer factorisation and discrete logarithms cannot be computed efficiently. On classical computers that assumption has held for decades. Shor showed that it fails on a quantum computer, because both problems reduce to finding the period of a modular function, a task at which quantum computers excel [1]. A 2021 analysis estimated that 2,048-bit RSA could be factored in about eight hours using roughly 20 million noisy physical qubits [2]. No such machine exists, but keys and recorded ciphertexts have lifetimes measured in decades, so the transition is already under way. In response, NIST ran a multi-year standardisation process [3] and in 2024 published three standards: a module-lattice key-encapsulation mechanism, ML-KEM [4]; a module-lattice signature scheme, ML-DSA [5]; and a stateless hash-based signature scheme, SLH-DSA [6]. Fig. 1 relates each family to the problem it depends on and to the best known quantum attack; for a broader survey see [7].

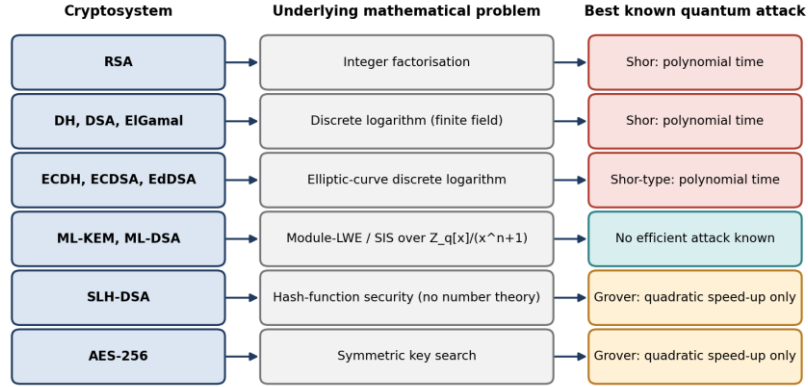


Fig. 1. Cryptosystems, their underlying problems and the best known quantum attacks. Elliptic-curve schemes fall to a variant of Shor's period-finding idea, and symmetric or hash-based schemes lose only a quadratic factor to Grover search [1], [8].

It is tempting to read this as the end of number theory in cryptography. It is not. This paper argues, and demonstrates experimentally, that (i) the quantum break is itself a number-theoretic argument, (ii) lattice cryptography replaces one hard problem with another but keeps modular arithmetic at its core, and (iii) the choice of the modulus $q = 3329$ in ML-KEM is dictated by number theory. Section 2 analyses the break, Section 3 introduces the lattice replacements, Section 4 reports three experiments, and Sections 5 and 6 discuss deployment costs and limits.

II. Why Number-Theoretic Schemes Fall

1. Order finding and factoring

Let N be an odd composite and a a base coprime to N . The order r of a modulo N is the smallest positive integer with

$$a^r \equiv 1 \pmod{N}. \quad (1)$$

If r is even and $a^{\frac{r}{2}} \not\equiv -1 \pmod{N}$, then N divides $(a^{\frac{r}{2}} - 1)(a^{\frac{r}{2}} + 1)$ without dividing either factor, so

$$\gcd\left(\frac{a^r}{2} \pm 1, N\right) \quad (2)$$

yields a nontrivial factor of N . Everything except the computation of r is classical and cheap. The quantum part of Shor's algorithm computes r in polynomial time [1], whereas no classical algorithm is known to do so. For a modulus that is a product of two distinct odd primes, at least half of all bases satisfy the two conditions, which is why a few random attempts suffice. Section 4.1 tests this claim.

2. Classical versus quantum cost

The best classical attack on RSA, the general number field sieve, runs in subexponential time, roughly $\exp\left(1.92 (\ln N)^{\frac{1}{3}} (\ln \ln N)^{\frac{2}{3}}\right)$. Fig. 2 plots this heuristic, with lower-order terms dropped, for moduli up to 8,192 bits; it places 2,048-bit and 3,072-bit moduli near 2^{117} and 2^{139} operations, in reasonable agreement with the 112- and 128-bit strength levels that NIST assigns to them [9]. Shor's algorithm, in contrast, needs a number of quantum gates polynomial in the bit length, so enlarging the modulus buys almost nothing once a large enough machine exists. Grover's algorithm, by comparison, offers only a quadratic speed-up for exhaustive search [8], which is why doubling symmetric key lengths is regarded as sufficient.

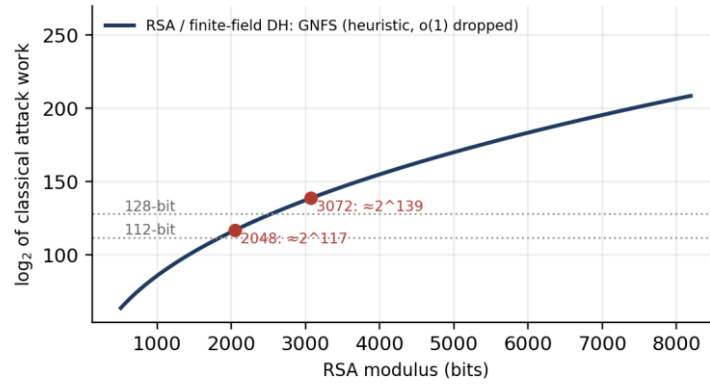


Fig. 2. Heuristic classical cost of the number field sieve against RSA modulus size ($o(1)$ terms omitted, so values are indicative only). A quantum computer running Shor's algorithm removes this growth altogether.

III. Lattice Cryptography: A New Hard Problem, Familiar Arithmetic

1. Learning with errors

Regev's LWE problem [10] asks for a secret vector s in Z_q^n given many noisy linear equations

$$b = As + e \pmod{q}, \quad (3)$$

where A is uniformly random and each entry of e is a small error. Without e , the system is solved by Gaussian elimination in cubic time. With e , no efficient classical or quantum algorithm is known, and Regev connected the average-case difficulty of LWE to worst-case lattice problems; the earlier work of Ajtai [11] had established the first such worst-case-to-average-case link. Plain LWE requires large matrices, so practical schemes use structure: polynomial rings $R_q = \frac{Zq[x]}{x^{n+1}}$ [12] and, in ML-KEM and ML-DSA, modules of small rank over such rings [13], [14]. A comprehensive account is given in [15].



2. The standardised parameter sets

Table I lists the standardised sets. ML-KEM fixes $n = 256$ and $q = 3329$ and varies the module rank k from 2 to 4 [4]. ML-DSA has three sets with larger keys and signatures [5]. The sizes follow directly from the module dimensions, and they exceed those of classical schemes by one to two orders of magnitude (Section 5).

Table 1. Standardised Lattice-Based Parameter Sets

Parameter set	Module rank k (KEM) / dimension (DSA)	Public key (bytes)	Ciphertext or signature (bytes)	NIST security category
ML-KEM-512	2	800	768	1
ML-KEM-768	3	1,184	1,088	3
ML-KEM-1024	4	1,568	1,568	5
ML-DSA-44	(4, 4)	1,312	2,420	2
ML-DSA-65	(6, 5)	1,952	3,309	3
ML-DSA-87	(8, 7)	2,592	4,627	5

Sizes and categories as specified in FIPS 203 and FIPS 204 [4], [5].

3. Number theory inside the implementation

Multiplying two degree-255 polynomials naively takes $256^2 = 65,536$ coefficient products. The number-theoretic transform (NTT) is the finite-field analogue of the fast Fourier transform [16]: it evaluates the polynomials at roots of unity in $\mathbb{Z}_q$, multiplies pointwise, and interpolates back. Its feasibility is a question of elementary number theory. The prime $q = 3329$ satisfies

$$q - 1 = 3328 = 2^8 \cdot 13, \quad (4)$$

so $\mathbb{Z}_q^*$ contains elements of order 256 (for instance 17, used in the standard) but not of order 512. The polynomial $x^{256} + 1$ would need primitive 512th roots to split completely into linear factors; instead it splits into 128 quadratic factors $x^2 - \gamma$, where γ ranges over odd powers of 17. The transform therefore runs for seven layers rather than eight, and the final step multiplies degree-one polynomials modulo each quadratic. This “incomplete NTT” is a direct consequence of the factorisation in Eq. (4) and an application of the Chinese remainder theorem to polynomial rings, closely mirroring the CRT trick used in RSA.

IV. Experiments

All experiments were run in Python 3.12 on a single CPU core with fixed random seeds. They are small-scale demonstrations of mechanisms, not security evaluations.

1. Success probability of the classical reduction in Shor’s algorithm

For 300 randomly generated semiprimes $N = pq$ with distinct primes between 50 and 400, every base a coprime to N was tested. The order r was computed exactly, standing in for the quantum subroutine, and the outcome was classified. As Fig. 3 shows, 74.3%

of bases yielded a nontrivial factor, 11.4% failed because r was odd, and 14.4% failed because $a^{\frac{r}{2}} \equiv -1 \pmod{N}$. The per-modulus success probability ranged from 50.0% upward with a mean of 73.9%, in line with the theoretical lower bound of one half. Repeating with fresh bases therefore drives the failure rate down geometrically, so the quantum step is the only obstacle.

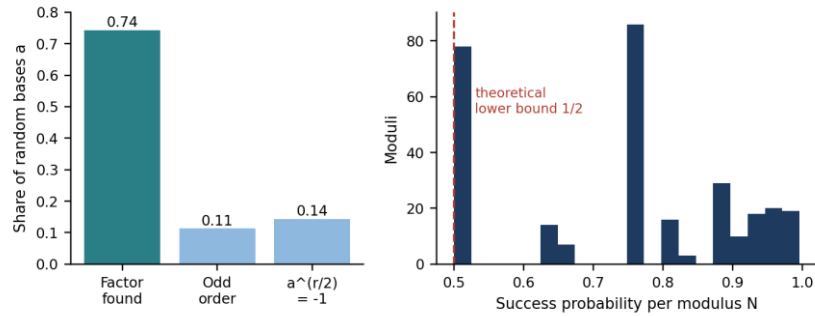


Fig. 3. Left: outcomes of the classical post-processing step over all coprime bases for 300 semiprimes. Right: distribution of the per-modulus success probability, which never fell below the theoretical bound of $1/2$.

2. Noise as the source of hardness in LWE

For dimensions $n = 8, 16$ and 32 with $q = 3329$, random square systems $b = As + e$ were generated with each error entry uniform on $[-B, B]$, and solved by Gaussian elimination modulo q . Results are shown in Fig. 4, based on 300 trials per setting. With $B = 0$ the secret was recovered in every trial. With $B = 1$, and for all larger B tested, it was recovered in none. Exact solving amplifies even a single unit of noise into an unrelated solution, because inverting a random matrix mod q spreads each error across all coordinates. Real attacks use many samples and lattice reduction, and their cost grows steeply with n ; the toy dimensions here show only why noise breaks the linear-algebra shortcut.

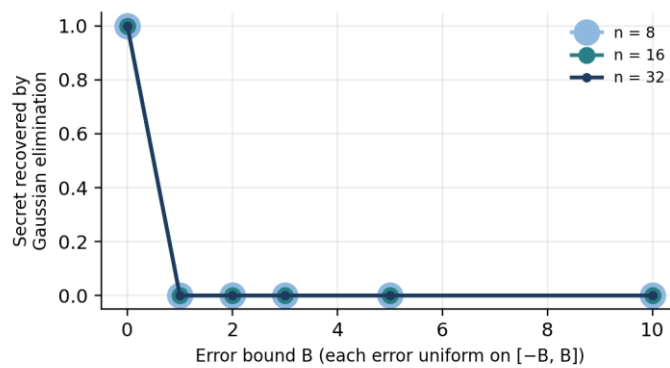


Fig. 4. Fraction of instances in which Gaussian elimination recovered the LWE secret. The curves for $n = 8, 16$ and 32 coincide (all equal 1 at $B = 0$ and 0 for $B \geq 1$); marker sizes differ so that all three remain visible.

3. A number-theoretic transform over $\mathbb{Z}_{3329}$

The NTT of the standard [4] was implemented, including the seven-layer forward and inverse transforms and the base-case multiplication of degree-one polynomials. First, the arithmetic assumptions were checked: 17 has multiplicative order exactly 256 modulo 3329, and $3329 \equiv 1 \pmod{256}$ but not $\pmod{512}$. Second, for 20 random pairs of polynomials in $\frac{\mathbb{Z}_q[x]}{x^{256}+1}$, the NTT-based product matched schoolbook negacyclic multiplication in every case, and forward-then-inverse transforms returned the original input. Third, operations were counted. Schoolbook multiplication uses 65,536 modular multiplications; the NTT route uses 3,584 (1,792 for two forward transforms, 1,152 for the inverse transform including scaling, and 640 for the 128 base-case products), a reduction by a factor of 18.3. Even in unoptimised Python the transform-based product was about seven times faster (0.91 ms versus 6.50 ms, Fig. 5).

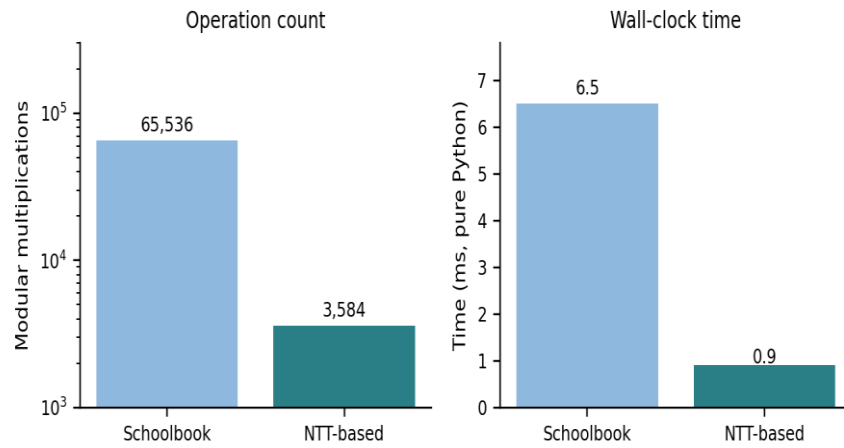


Fig. 5. Cost of multiplying two polynomials in $\mathbb{Z}_{3329}[x]/(x^{256}+1)$: modular multiplications (left, logarithmic axis) and wall-clock time in pure Python (right).

V. Deployment Implications

The main practical cost of lattice schemes is size. Fig. 6 compares the bytes exchanged by each party in a key agreement. X25519 needs 32 bytes per side, or 64 in total; an ML-KEM-768 exchange needs 1,184 bytes for the encapsulation key plus 1,088 for the ciphertext, or 2,272 bytes, about 35 times more. Signatures show a similar gap: Ed25519 uses a 32-byte public key and 64-byte signature, whereas ML-DSA-65 uses 1,952 and 3,309 bytes, respectively (Tables I and II). Computation, by contrast, is competitive, since the NTT keeps polynomial arithmetic cheap [13], [14].

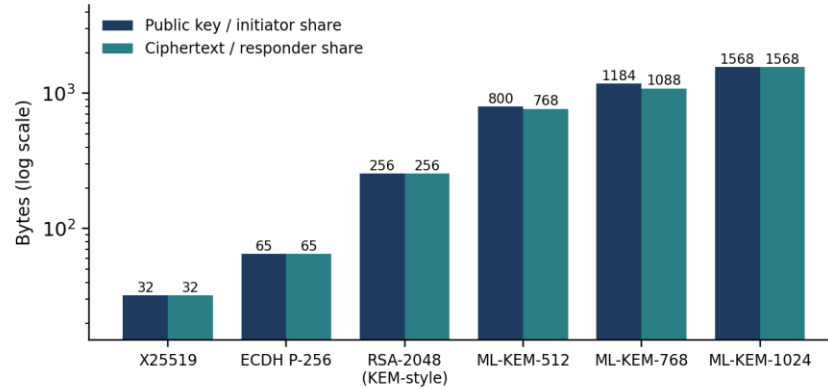


Fig. 6. Bytes sent by each party in a key exchange, classical versus ML-KEM (logarithmic axis). Byte counts follow the sizes of the respective specifications [4], [17].

Table 2. Classical Baselines for Comparison with Table I

Scheme	Public key (bytes)	Signature (bytes)	Hardness basis
Ed25519	32	64	Elliptic-curve DLP
ECDSA P-256	65 (uncompressed)	64 (raw r, s)	Elliptic-curve DLP
RSA-2048	256 (modulus)	256	Integer factorisation

Typical encoded sizes; values are for uncompressed or raw encodings as noted.

Three consequences follow. First, larger handshake messages matter most for constrained links and for protocols with strict packet limits, so protocol designers must plan for fragmentation and certificate-chain growth. Second, because encrypted traffic recorded today could be decrypted later, key establishment is the most urgent component to migrate, while signatures can follow on the timescale of certificate lifetimes. Third, organisations often combine a classical and a post-quantum key exchange during the transition, so that security holds if either component survives; this hedges against unexpected weaknesses in the newer schemes.

VI. Discussion and Limitations

The experiments illustrate a continuity that the headline “quantum-safe” obscures. The reduction in Shor’s algorithm uses only orders, gcds and the structure of the unit group, and the same ideas (CRT, roots of unity, group orders) reappear in the design of the NTT. What changes is the source of hardness: in place of the difficulty of recovering an exponent or a factor, security now rests on the difficulty of separating a small error from a linear structure.



Several limitations apply. The order-finding experiment computes r classically and therefore says nothing about quantum resource needs, which are addressed in [2]. The LWE experiment uses tiny dimensions and a naive solver, so it explains the role of noise but does not measure security. The NTT experiment validates correctness and operation counts for one parameter ($q = 3329$, $n = 256$); timing in pure Python does not predict optimised performance. Finally, lattice assumptions are younger than factoring, and cryptanalysis continues; standards may evolve, which supports designing systems for algorithm agility.

VII. Conclusion

Number theory both explains the vulnerability of today's public-key cryptography and supplies machinery for its successors. The quantum break reduces to order finding, which succeeds with probability at least one half per random base; lattice schemes remove the structure that period finding exploits by adding noise, and they rely on the factorisation of $q - 1$ to compute efficiently. Migration will cost bandwidth, roughly 35 times more for a key exchange in the comparison above, but not prohibitive computation. Organisations should inventory their public-key uses, prioritise key establishment, adopt the NIST standards, and retain the agility to change algorithms as understanding of lattice problems matures.

References

1. P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997, doi: <https://doi.org/10.1137/S0097539795293172>.
2. C. Gidney and M. Ekerå, "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits," *Quantum*, vol. 5, p. 433, Apr. 2021, doi: <https://doi.org/10.22331/q-2021-04-15-433>.
3. G. Alagic et al., "Status report on the third round of the NIST post-quantum cryptography standardization process," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, NISTIR 8413, Sep. 2022, doi: <https://doi.org/10.6028/NIST.IR.8413-upd1>.
4. National Institute of Standards and Technology, "Module-lattice-based key-encapsulation mechanism standard," Gaithersburg, MD, USA, FIPS 203, Aug. 2024, doi: <https://doi.org/10.6028/NIST.FIPS.203>.
5. National Institute of Standards and Technology, "Module-lattice-based digital signature standard," Gaithersburg, MD, USA, FIPS 204, Aug. 2024, doi: <https://doi.org/10.6028/NIST.FIPS.204>.
6. National Institute of Standards and Technology, "Stateless hash-based digital signature standard," Gaithersburg, MD, USA, FIPS 205, Aug. 2024, doi: <https://doi.org/10.6028/NIST.FIPS.205>.
7. D. J. Bernstein and T. Lange, "Post-quantum cryptography," *Nature*, vol. 549, no. 7671, pp. 188–194, Sep. 2017, doi: <https://doi.org/10.1038/nature23461>.
8. L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Comput. (STOC)*, 1996, pp. 212–219, doi: <https://doi.org/10.1145/237814.237866>.



9. E. Barker, "Recommendation for key management: Part 1 – General," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, NIST SP 800-57 Part 1 Rev. 5, May 2020, doi: <https://doi.org/10.6028/NIST.SP.800-57pt1r5>.
10. O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, Art. no. 34, Sep. 2009, doi: <https://doi.org/10.1145/1568318.1568324>.
11. M. Ajtai, "Generating hard instances of lattice problems," in *Proc. 28th Annu. ACM Symp. Theory Comput. (STOC)*, 1996, pp. 99–108, doi: <https://doi.org/10.1145/237814.237838>.
12. V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *Advances in Cryptology—EUROCRYPT 2010 (LNCS 6110)*, Berlin, Germany: Springer, 2010, pp. 1–23, doi: https://doi.org/10.1007/978-3-642-13190-5_1.
13. J. Bos et al., "CRYSTALS–Kyber: A CCA-secure module-lattice-based KEM," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, 2018, pp. 353–367, doi: <https://doi.org/10.1109/EuroSP.2018.00032>.
14. L. Ducas et al., "CRYSTALS-Dilithium: A lattice-based digital signature scheme," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2018, no. 1, pp. 238–268, 2018, doi: <https://doi.org/10.13154/tches.v2018.i1.238-268>.
15. C. Peikert, "A decade of lattice cryptography," *Found. Trends Theor. Comput. Sci.*, vol. 10, no. 4, pp. 283–424, 2016, doi: <https://doi.org/10.1561/04000000074>.
16. J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comput.*, vol. 19, no. 90, pp. 297–301, Apr. 1965, doi: <https://doi.org/10.1090/S0025-5718-1965-0178586-1>.
17. A. Langley, M. Hamburg, and S. Turner, "Elliptic curves for security," IETF, RFC 7748, Jan. 2016, doi: <https://doi.org/10.17487/RFC7748>.