



Applications of Number Theory in Modern Cryptographic Systems: Foundations, Public-Key Schemes and Empirical Performance

Tripti Gautam, Research Scholar¹,

Mahakaushal University, Jabalpur, (M.P.)

triptigautam002@gmail.com¹

Dr. Narendra Bahadur Singh, Professor²,

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

Abstract- Almost every public-key cryptosystem deployed today rests on a small set of results from elementary and algebraic number theory: modular arithmetic, Euler's theorem, the Chinese remainder theorem, primality testing and the group structure of finite fields and elliptic curves. This paper explains how these results yield the RSA, Diffie-Hellman/ElGamal and elliptic-curve families, and then tests the central claims with reproducible experiments. Over 1,070 odd composites below 3,000, no Miller-Rabin base misclassified more than 23.7% of candidates (theoretical bound 25%), whereas the Fermat test was fooled by up to 76.6%. Baby-step giant-step discrete-logarithm cost followed the square-root law across 16–44-bit primes. Applying the Chinese remainder theorem made RSA private-key operations 2.8–3.8 times faster for 1,024–4,096-bit moduli. In a library benchmark, RSA-2048 signing was about 11 times slower than ECDSA P-256 signing, while RSA verification was several times faster than elliptic-curve verification. The paper closes with the parameter and implementation lessons that follow from these results and notes the limit imposed by quantum computing.

Keywords- number theory, public-key cryptography, RSA, discrete logarithm, elliptic curves, primality testing, Chinese remainder theorem.

I. INTRODUCTION

Public-key cryptography began with the observation that some arithmetic operations are easy to perform but hard to reverse [1]. Multiplying two large primes is cheap; recovering them from the product is not. Raising a group element to a power is cheap; recovering the exponent is not. These asymmetries, formalised in the RSA cryptosystem [2], in Diffie-Hellman key agreement [1] and in



ElGamal encryption [3], and later transferred to elliptic-curve groups [4], [5], are all statements of number theory. They now protect web traffic through TLS 1.3 [6] and underpin the digital signature standards published by NIST [7].

Textbook treatments describe the mathematics, and benchmark papers describe performance, but the two are rarely connected. This paper makes three contributions. First, it maps the number-theoretic results used in practice to the schemes that depend on them (Section 2 and Section 3). Second, it verifies four of those connections experimentally, using code and settings that are fully described (Section 4). Fig. 1 gives the overall map. Third, it draws practical conclusions on parameter choice, implementation and the limits of these schemes (Sections 5 and 6).

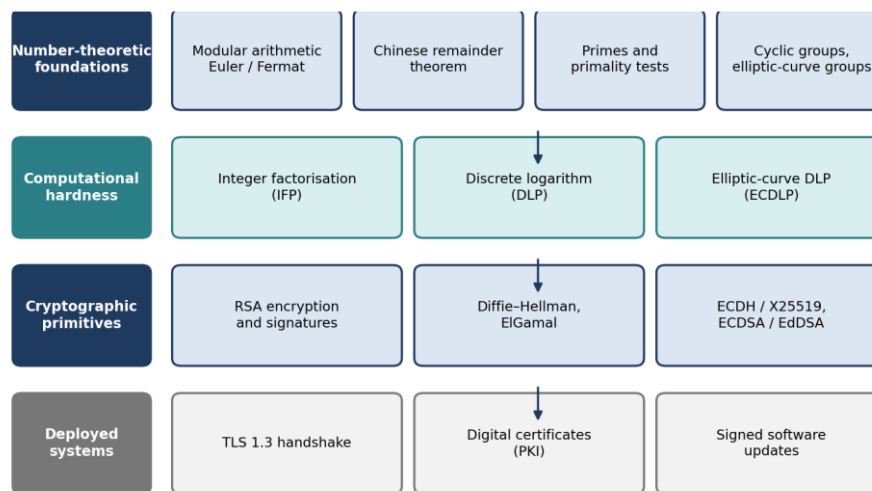


Fig. 1. From number theory to deployed systems. Each layer relies on the one above it: mathematical results define hard problems, hard problems define primitives, and primitives are assembled into protocols.

II. NUMBER-THEORETIC FOUNDATIONS

1. Modular arithmetic and Euler's theorem

Cryptographic arithmetic takes place in the ring of integers modulo n . The elements coprime to n form a multiplicative group of order $\phi(n)$, and Euler's theorem states that $a^{\phi(n)} \equiv 1 \pmod{n}$ whenever $\gcd(a, n) = 1$. For a prime p this reduces to Fermat's little theorem, $a^{p-1} \equiv 1 \pmod{p}$. Efficient modular multiplication, for example by Montgomery reduction [8], makes exponentiation practical for numbers of thousands of bits.

2. The Chinese remainder theorem

If $n = pq$ with p and q coprime, the ring $\mathbb{Z}_n$ is isomorphic to $\mathbb{Z}_p \times \mathbb{Z}_q$. A computation modulo n can therefore be split into two computations on numbers half the size and recombined. Because modular



exponentiation costs roughly the cube of the operand length with schoolbook multiplication, halving the operands makes each half about eight times cheaper, and the pair about four times cheaper. Section 4.4 measures this effect.

3. Primes and primality testing

Key generation needs large random primes. The Fermat test checks whether $a^{(n-1)} \equiv 1 \pmod{n}$ for chosen bases a , but it is defeated by Carmichael numbers, which satisfy the congruence for every coprime base. The Miller–Rabin test [9], [10] strengthens the condition by examining the square roots of 1 encountered while computing $a^{(n-1)}$; for any odd composite, at most one quarter of the bases fail to expose it. A deterministic polynomial-time test exists [11], but randomised Miller–Rabin remains faster in practice, and repeating it with independent bases drives the error probability down geometrically.

4. Hard problems and elliptic-curve groups

Three problems carry most of the load. The integer factorisation problem (IFP) underlies RSA. The discrete logarithm problem (DLP) asks for x given g and g^x in a finite field. The elliptic-curve discrete logarithm problem (ECDLP) asks the same in the group of points on a curve $y^2 = x^3 + ax + b$ over a finite field. The order of that group lies within $2\sqrt{p}$ of $p + 1$ (Hasse's bound) and can be computed in polynomial time [12], which is what allows safe curves to be generated. For well-chosen curves, no attack better than generic square-root methods is known, so much shorter keys suffice than for IFP or DLP over the integers [13].

III. PUBLIC-KEY SCHEMES BUILT ON THESE RESULTS

1. RSA

RSA chooses two large primes p and q and defines

$$n = pq, \varphi(n) = (p-1)(q-1), ed \equiv 1 \pmod{\varphi(n)}. \quad (1)$$

Encryption computes $c \equiv m^e \pmod{n}$ and decryption $m \equiv c^d \pmod{n}$; Euler's theorem guarantees recovery [2]. Signing reverses the roles of the exponents. With the Chinese remainder theorem, the private operation becomes

$$m_1 = cd \pmod{(p-1) \pmod{p}}, m_2 = cd \pmod{(q-1) \pmod{q}}, m = m_2 + q \cdot [q-1(m_1 - m_2) \pmod{p}]. \quad (2)$$

2. Diffie–Hellman and ElGamal

Two parties agree on a prime p and generator g . Alice sends $A = g^a \pmod{p}$, Bob sends $B = g^b \pmod{p}$, and both derive

$$K = Ba \pmod{p} = Ab \pmod{p} = gab \pmod{p}. \quad (3)$$



An eavesdropper sees g , A and B and must solve a discrete-logarithm-type problem to obtain K [1]. ElGamal turns the same structure into public-key encryption and signatures [3]. When the group order is smooth, the Pohlig–Hellman method reduces the DLP to the prime-power factors of the order [14], so parameters must be chosen with a large prime-order subgroup.

3. Elliptic-curve cryptography

Replacing the multiplicative group with an elliptic-curve group turns exponentiation into scalar multiplication, $Q = kP$. The protocols keep the same shape, but the relevant key sizes shrink sharply because subexponential index-calculus attacks do not apply to well-chosen curves. Curve25519 [15] and the X25519 function standardised in RFC 7748 [16] were designed for fast, constant-time implementation, and the Digital Signature Standard now includes elliptic-curve variants [7]. Table 1 summarises the correspondence between number-theoretic ideas and their cryptographic use.

Table 1. Number-Theoretic Results and Their Cryptographic Role

Result	Statement (informal)	Cryptographic role
Euler's theorem	$a^{\varphi(n)} \equiv 1 \pmod{n}$ for $\gcd(a, n) = 1$	Correctness of RSA decryption and signing [2]
Chinese remainder theorem	$\mathbb{Z}_n \cong \mathbb{Z}_p \times \mathbb{Z}_q$ for $n = pq$	Faster RSA private operations; also a source of fault attacks
Miller–Rabin test	At most $1/4$ of bases fail for any odd composite	Fast generation of random primes [9], [10]
Cyclic group structure	Exponentiation is easy, logarithm is hard	Diffie–Hellman, ElGamal, DSA [1], [3]
Hasse's bound, Schoof's algorithm	Point count near $p + 1$; computable in polynomial time	Curve parameter validation [12]
Pohlig–Hellman reduction	DLP cost governed by largest prime factor of group order	Requires prime-order subgroups [14]

IV. EXPERIMENTAL EVALUATION

1. Setup

All experiments ran on a single CPU core in a Python 3.12 environment. Primality, discrete-logarithm and CRT experiments used custom code built on Python's arbitrary-precision integers; the comparative benchmark used the Python *cryptography* library, version 46.0.6. Timings are means over repeated runs on one machine and are meant to show relative behaviour, not to serve as absolute performance figures.

2. Primality tests on composite numbers

For each of the 1,070 odd composites between 9 and 3,000, every base a in $[2, n - 2]$ was tried, and the fraction of bases that wrongly signalled “probably prime” was recorded for the Fermat and Miller–Rabin tests. Fig. 2 shows the result. The Fermat test was fooled by as many as 76.6% of bases (for example, 57.0% of bases for the Carmichael number 561), while the worst Miller–Rabin case was 23.7%, consistent



with the one-quarter bound [10]. For 561, only 1.4% of bases deceived Miller–Rabin. The mean liar fraction was 0.6% for Miller–Rabin and 2.2% for Fermat.

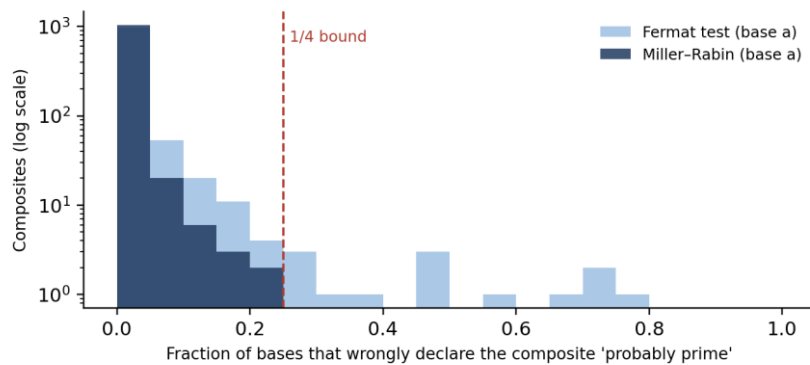


Fig. 2. Distribution of “liar” fractions over 1,070 odd composites below 3,000 (vertical axis logarithmic). Miller–Rabin never exceeded the theoretical bound of $1/4$, whereas the Fermat test failed for a large share of bases on Carmichael-type inputs.

3. Cost of discrete-logarithm computation

Baby-step giant-step was applied to random prime moduli of 16 to 44 bits (12 instances per size, random exponent), counting group operations. As Fig. 3 shows, the measured cost tracks a square-root law and rises by about a factor of four for every four extra bits of modulus, far below the 2^b of exhaustive search. This is the generic bound that any cyclic group must withstand, and it is why group sizes are chosen so that $2^{(b/2)}$ is infeasible [13], [17].

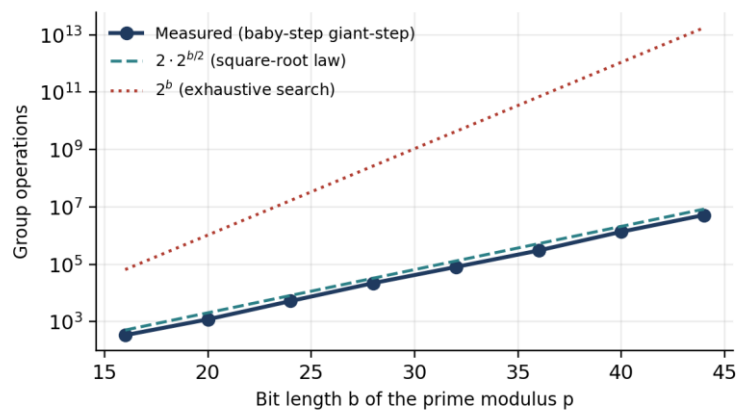


Fig. 3. Measured baby-step giant-step cost against prime size b , with the square-root law and brute-force cost for reference.



4. Effect of the Chinese remainder theorem on RSA

For random RSA keys of 1,024, 2,048, 3,072 and 4,096 bits ($e = 65,537$), private-key exponentiation was timed directly and via Eq. (2); both methods gave identical outputs. The CRT method was faster by factors of 2.8, 3.3, 3.5 and 3.8, respectively (Fig. 4), approaching the theoretical value of four as operands grew. The public operation was one to two orders of magnitude cheaper than the private one because e is small.

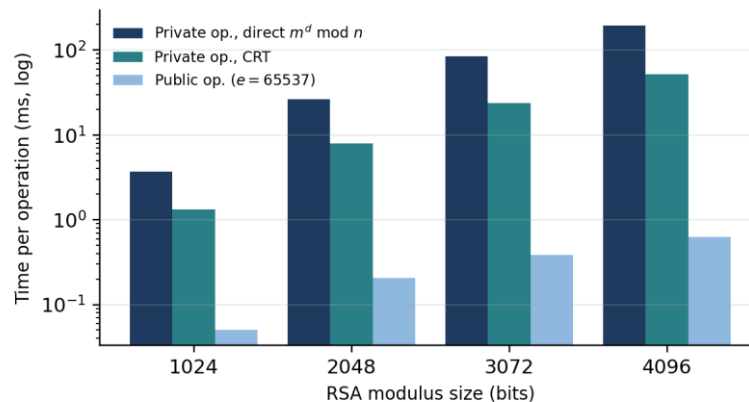


Fig. 4. Time per RSA operation for direct exponentiation, CRT exponentiation and the public operation (logarithmic axis).

5. RSA compared with elliptic-curve schemes

Fig. 5 reports library timings for signing, verification and key agreement. RSA-2048 signing took 0.350 ms and RSA-3072 signing 1.03 ms, against 0.031 ms for ECDSA P-256 and 0.037 ms for Ed25519, so the elliptic-curve schemes signed about 11 and 10 times faster than RSA-2048. The picture reverses for verification: RSA-2048 took 0.024 ms, against 0.089 ms for P-256 and 0.108 ms for Ed25519. X25519 key agreement took 0.036 ms. RSA therefore remains attractive when many signatures are verified and few are produced, while elliptic curves suit signers, constrained devices and ephemeral key exchange.

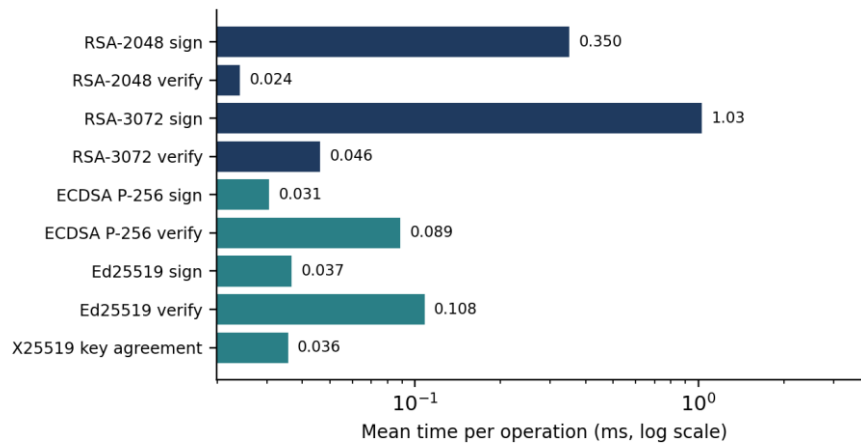


Fig. 5. Mean time per operation for RSA and elliptic-curve schemes (logarithmic axis, single core). Lower is faster.

V. SECURITY CONSIDERATIONS

1. Algorithmic attacks and parameter sizes

The strongest known attack on RSA moduli and finite-field discrete logarithms is the number field sieve. A 768-bit RSA modulus was factored in 2010 [18], and a 240-digit (795-bit) factorisation and discrete-logarithm computation followed in 2020 [19]. Older methods remain relevant for badly chosen parameters: Pollard's $p - 1$ [20] and Lenstra's elliptic-curve method [21] find small or structured factors, and the MOV reduction [22] maps some elliptic-curve logarithms into finite fields. Because elliptic-curve groups resist subexponential attacks, NIST's equivalence table pairs a 3,072-bit RSA modulus with roughly a 256-bit curve at the 128-bit security level (Table II) [13].

Table 2. Comparable Key Sizes for Equivalent Security Strength (Nist Sp 800-57)

Security strength (bits)	RSA / finite-field DH modulus (bits)	Elliptic-curve key size (bits)
112	2,048	224–255
128	3,072	256–383
192	7,680	384–511
256	15,360	512 and above

Values are taken from the key-management recommendation [13].

2. Implementation and deployment weaknesses

Most real failures have come from parameters and implementations rather than from breaking the underlying problem. Widespread reuse of a few 512- and 1,024-bit Diffie–Hellman primes made large-scale precomputation attacks feasible [23]. A flawed prime-generation routine in a widely used library



produced RSA moduli with structure that permitted practical factorisation [24]. Timing variations in modular exponentiation leaked private-key bits [25]. These cases motivate validated parameters, constant-time arithmetic and careful randomness.

3. The quantum limit

Shor's algorithm solves factorisation and discrete logarithms in polynomial time on a sufficiently large quantum computer [26], and a 2021 resource analysis estimated that 2,048-bit RSA could be factored using about 20 million noisy qubits in roughly eight hours [27]. No machine of that scale exists today, but the result explains why NIST has standardised lattice-based replacements [28], [29]. That transition is outside the scope of this paper.

VI. DISCUSSION

The experiments support several practical points. The Miller–Rabin bound is not merely theoretical: even the worst case observed stayed below 24%, so a handful of random bases is enough for key generation, whereas the Fermat test alone is unsafe for adversarial inputs. The square-root law for discrete logarithms explains the doubling rule of thumb, in which a group must be twice as many bits as its target security level. The CRT gain shows how a purely algebraic identity can be worth a four-fold speed-up, but it also enlarges the attack surface, since a fault in one half-computation can reveal a factor; implementations should verify results before releasing them. Finally, the RSA/ECC contrast shows that “faster” depends on the operation: system designers should choose by workload rather than by slogan. Limitations apply. The primality experiment covers only small composites, the discrete-logarithm experiment uses prime fields and a generic algorithm, and the benchmark reflects one library on one core. Results should be read as confirmation of mechanisms, not as performance guarantees.

VII. CONCLUSION

Modular arithmetic, the Chinese remainder theorem, primality testing and group theory are not background material for modern cryptography; they are its working parts. The experiments confirmed the Miller–Rabin error bound, the square-root cost of generic discrete-logarithm search, the CRT speed-up of about four for RSA, and a workload-dependent contrast between RSA and elliptic-curve schemes. Sound practice combines validated parameters, constant-time implementations and awareness that the number-theoretic assumptions involved cannot resist a large quantum computer.

REFERENCES

1. W. Diffie and M. E. Hellman, “New directions in cryptography,” *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976, doi: <https://doi.org/10.1109/TIT.1976.1055638>.
2. R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978, doi: <https://doi.org/10.1145/359340.359342>.



3. T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985, doi: <https://doi.org/10.1109/TIT.1985.1057074>.
4. N. Koblitz, "Elliptic curve cryptosystems," *Math. Comput.*, vol. 48, no. 177, pp. 203–209, Jan. 1987, doi: <https://doi.org/10.1090/S0025-5718-1987-0866109-5>.
5. V. S. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology—CRYPTO '85 (LNCS 218)*, Berlin, Germany: Springer, 1986, pp. 417–426, doi: https://doi.org/10.1007/3-540-39799-X_31.
6. E. Rescorla, "The Transport Layer Security (TLS) protocol version 1.3," IETF, RFC 8446, Aug. 2018, doi: <https://doi.org/10.17487/RFC8446>.
7. National Institute of Standards and Technology, "Digital signature standard (DSS)," Gaithersburg, MD, USA, FIPS 186-5, Feb. 2023, doi: <https://doi.org/10.6028/NIST.FIPS.186-5>.
8. P. L. Montgomery, "Modular multiplication without trial division," *Math. Comput.*, vol. 44, no. 170, pp. 519–521, Apr. 1985, doi: <https://doi.org/10.1090/S0025-5718-1985-0777282-X>.
9. G. L. Miller, "Riemann's hypothesis and tests for primality," *J. Comput. Syst. Sci.*, vol. 13, no. 3, pp. 300–317, Dec. 1976, doi: [https://doi.org/10.1016/S0022-0000\(76\)80043-8](https://doi.org/10.1016/S0022-0000(76)80043-8).
10. M. O. Rabin, "Probabilistic algorithm for testing primality," *J. Number Theory*, vol. 12, no. 1, pp. 128–138, Feb. 1980, doi: [https://doi.org/10.1016/0022-314X\(80\)90084-0](https://doi.org/10.1016/0022-314X(80)90084-0).
11. M. Agrawal, N. Kayal, and N. Saxena, "PRIMES is in P," *Ann. Math.*, vol. 160, no. 2, pp. 781–793, Sep. 2004, doi: <https://doi.org/10.4007/annals.2004.160.781>.
12. R. Schoof, "Elliptic curves over finite fields and the computation of square roots mod p," *Math. Comput.*, vol. 44, no. 170, pp. 483–494, Apr. 1985, doi: <https://doi.org/10.1090/S0025-5718-1985-0777280-6>.
13. 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>.
14. S. C. Pohlig and M. E. Hellman, "An improved algorithm for computing logarithms over GF(p) and its cryptographic significance," *IEEE Trans. Inf. Theory*, vol. 24, no. 1, pp. 106–110, Jan. 1978, doi: <https://doi.org/10.1109/TIT.1978.1055817>.
15. D. J. Bernstein, "Curve25519: New Diffie-Hellman speed records," in *Public Key Cryptography—PKC 2006 (LNCS 3958)*, Berlin, Germany: Springer, 2006, pp. 207–228, doi: https://doi.org/10.1007/11745853_14.
16. A. Langley, M. Hamburg, and S. Turner, "Elliptic curves for security," IETF, RFC 7748, Jan. 2016, doi: <https://doi.org/10.17487/RFC7748>.
17. J. M. Pollard, "Monte Carlo methods for index computation (mod p)," *Math. Comput.*, vol. 32, no. 143, pp. 918–924, Jul. 1978, doi: <https://doi.org/10.1090/S0025-5718-1978-0491431-9>.
18. T. Kleinjung et al., "Factorization of a 768-bit RSA modulus," in *Advances in Cryptology—CRYPTO 2010 (LNCS 6223)*, Berlin, Germany: Springer, 2010, pp. 333–350, doi: https://doi.org/10.1007/978-3-642-14623-7_18.
19. F. Boudot, P. Gaudry, A. Guillevis, N. Heninger, E. Thomé, and P. Zimmermann, "Comparing the difficulty of factorization and discrete logarithm: A 240-digit experiment," in *Advances in Cryptology—CRYPTO 2020 (LNCS 12171)*, Cham, Switzerland: Springer, 2020, pp. 62–91, doi: https://doi.org/10.1007/978-3-030-56880-1_3.



20. J. M. Pollard, "Theorems on factorization and primality testing," *Math. Proc. Cambridge Philos. Soc.*, vol. 76, no. 3, pp. 521–528, Nov. 1974, doi: <https://doi.org/10.1017/S0305004100049252>.
21. H. W. Lenstra, Jr., "Factoring integers with elliptic curves," *Ann. Math.*, vol. 126, no. 3, pp. 649–673, Nov. 1987, doi: <https://doi.org/10.2307/1971363>.
22. A. J. Menezes, T. Okamoto, and S. A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE Trans. Inf. Theory*, vol. 39, no. 5, pp. 1639–1646, Sep. 1993, doi: <https://doi.org/10.1109/18.259647>.
23. D. Adrian et al., "Imperfect forward secrecy: How Diffie-Hellman fails in practice," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2015, pp. 5–17, doi: <https://doi.org/10.1145/2810103.2813707>.
24. M. Nemec, M. Šys, P. Švenda, D. Klinec, and V. Matyáš, "The return of Coppersmith's attack: Practical factorization of widely used RSA moduli," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2017, pp. 1631–1648, doi: <https://doi.org/10.1145/3133956.3133969>.
25. P. C. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," in *Advances in Cryptology—CRYPTO '96 (LNCS 1109)*, Berlin, Germany: Springer, 1996, pp. 104–113, doi: https://doi.org/10.1007/3-540-68697-5_9.
26. 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>.
27. 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>.
28. 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>.
29. 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>.