African Journal of Advanced Pure and Applied Sciences (AJAPAS)
Online ISSN: 2957-644X
Volume 5, Issue 3, 2026
Page No: 263-274
Website: *https://aaasjournals.com/index.php/ajapas/index*
ISI 2025: 1.126 **SJIFactor 2024: 6.752** **معامل التأثير العربي: 1.62**

A Novel Number-Theoretic Model for Secure Key Generation in Information Security

Emad Awadh [1*], Aziza Rashed [2], Abdulfatah Ibrahim [3], Omar Murajia Ali [4]
[1] Department of Mathematics, Faculty of Science, University of Tobruk, Tobruk, Libya
[2,3] Department of Computer Science, Faculty of Science, University of Tobruk, Tobruk, Libya
[4] Department of Information Technology, Higher Institute of Science and Technology, Musaid, Libya

نموذج رياضي عددي جديد لتوليد المفاتيح الآمنة في أمن المعلومات

عماد عوض [1*]، عزيزة راشد [2]، عبد الفتاح إبراهيم [3]، عمر مراجع علي [4]
[1] قسم الرياضيات، كلية العلوم، جامعة طبرق، طبرق، ليبيا
[3،2] قسم علوم الحاسوب، كلية العلوم، جامعة طبرق، طبرق، ليبيا
[4] قسم تقنية المعلومات، المعهد العالي للعلوم والتقنية، امساعد، ليبيا

[*]Corresponding author: emad.awadh@tu.edu.ly

Received: May 04, 2026	Accepted: August 02, 2026	Published: August 13, 2026

Abstract
Every deployed RSA implementation rests on one algebraic assumption: that factoring large integers is hard. Nobody has an independent, formally verified way to check that a given private key was actually generated correctly, and there is no well-understood second layer sitting on top of that single assumption. Quantum algorithms and decades of cryptanalytic progress keep narrowing the margin around it, which is part of why standards bodies are already moving toward lattice-based replacements. This paper works out the mathematics of an existing idea that earlier graph-augmented RSA proposals mostly left informal: binding the RSA private exponent to a directed graph G(V, E) generated by the public permutation $u \mapsto u^e \pmod n$. We prove the construction is correct for every valid key pair, show that G is always a publicly and efficiently decomposable union of disjoint directed cycles rather than an NP-hard structure, derive the exact condition under which a simple trapdoor path of a requested length exists, and give closed-form time and space complexity bounds. Every result is checked twice: once by hand on a fully worked numerical example (n = 143), and once empirically, across a 72-configuration benchmark spanning three RSA modulus sizes and four graph orders, where the measured overhead ranges from 14% to 343% of baseline RSA key-generation time depending on the chosen |V|. The central security result is a clean reduction: recovering the model's private key is at least as hard as breaking standard RSA, and no harder. The graph layer adds no independent, NP-hardness-based security margin, so what's left is a single, precisely stated open question about resistance to partial-information and side-channel observation.

Keywords: RSA, Public-key cryptography, Number theory, Graph theory, Key generation, Cryptanalysis.

الملخص

تعتمد جميع تطبيقات نظام التشفير RSA المستخدمة حالياً على افتراض جبري واحد، وهو صعوبة تحليل الأعداد الصحيحة الكبيرة إلى عواملها الأولية، دون وجود طريقة مستقلة وموثقة رياضياً للتحقق من صحة توليد المفتاح الخاص. تقدم هذه الورقة نموذجاً يربط المفتاح الخاص لنظام RSA بمسار هاملتوني (Hamiltonian) في رسم بياني موجه G(V, E) يتولد من التبديل العام $u \mapsto u^e \pmod n$، ويقدم معالجة رياضية صارمة لهذه الفكرة: إثبات صحة البناء لكل

زوج مفاتيح صالح، وإثبات أن الرسم البياني الناتج هو دائماً اتحاد من دورات موجهة منفصلة قابلة للتفكيك بكفاءة وليس بنية معقدة من نوع NP، واشتقاق الشرط الدقيق لوجود مسار هاملتوني بالطول المطلوب، مع حدود زمنية ومكانية مغلقة الصيغة، تم التحقق منها يدوياً وتجريبياً عبر 72 تهيئة اختبارية. وتُظهر النتيجة الأمنية المركزية أن استرجاع المفتاح الخاص في هذا النموذج لا يقل صعوبة عن كسر نظام RSA القياسي ولا يزيد عنه، مع بقاء سؤال دقيق ومفتوح واحد يخص المقاومة لهجمات القناة الجانبية والمعلومات الجزئية.

الكلمات المفتاحية: تشفير المفتاح العام (RSA)، نظرية الأعداد، نظرية الرسوم البيانية، توليد المفاتيح، تحليل التشفير.

1. Introduction

The security of any digital system depends on how its keys are generated [4], [16], [24]. RSA [1] and ECC [2], [33] are still the pillars of current infrastructure, but each rests on a single hard problem, integer factorization for one and the discrete logarithm problem for the other, and both are increasingly exposed to classical cryptanalytic progress [9], [34] and to quantum algorithms [3], [31]. A fair amount of work has tried to strengthen RSA-family schemes with auxiliary combinatorial structure: graphs, lattices, coding-theoretic overlays. The pattern in that literature is a gap rather than a strength. The auxiliary structure tends to be described informally, its hardness is asserted rather than derived, and its computational cost usually goes unmeasured. That makes such a scheme hard for a practitioner to evaluate and just as hard for a cryptanalyst to attack in any principled way.

This paper works through that gap for one natural construction: binding RSA's algebraic structure to a directed graph via the public permutation $u \mapsto u^e \pmod n$. We give it the treatment these proposals usually lack: formal theorems with proofs (Section 4), a closed-form complexity analysis tied to measured runtime (Sections 4.5 and 7), and a precise statement of what security margin the construction does and does not provide (Sections 5–6), following established cryptographic engineering practice [15], [17], [21].

One point is worth flagging before going further, since it affects how several later sections should be read. Constructions of this kind often lean on the graph isomorphism problem (GIP) as a source of extra hardness: no polynomial-time algorithm is known for GIP on general graphs, and it is one of the few natural problems believed to sit strictly between P and NP-complete [6], [27]. Section 4 shows that appeal does not hold for the specific graph family built here, because the edge exponent is fixed to the public RSA exponent e, the induced graph is always a publicly and efficiently decomposable union of cycles rather than a generic hard instance. Throughout the paper we use "trapdoor" in the precise sense set up in that section, not the looser, GIP-flavored sense that similar constructions sometimes invoke. Before any of this can be made precise, though, we need the underlying number-theoretic machinery in place: the modulus, the totient function, and the Chinese Remainder Theorem, which later determines how efficiently the construction runs. Section 2 covers that ground.

2. Mathematical Foundations

2.1 Prime Number Theory and Euler Totient

The model uses two large primes, p and q, to form the modulus $n = p \times q$, following the classical RSA construction [1], [11], [23]. The Euler totient function, $\varphi(n)$, fixes the size of the key space:

$$\phi(n) = (p-1)(q-1)$$

Equation (1) gives the order of the multiplicative group $(\mathbb{Z}/n\mathbb{Z})^x$, which determines the public and private key exponents [12], [19], [25].

2.2 The Chinese Remainder Theorem (CRT)

To keep computational latency manageable at large key sizes, we use the Chinese Remainder Theorem, a standard technique for accelerating RSA-type operations [7], [18], [22]. For a system of congruences $x \equiv a_i \pmod{m_i}$ with pairwise coprime moduli m_i, the unique solution x (mod M) is:

$$x = \sum_{i=1}^{k} a_i M_i y_i \pmod M$$

Here $M = \prod_i m_i$, $M_i = M/m_i$, and $y_i = M_i^{-1} \pmod{m_i}$. This lets cryptographic operations run in parallel, which keeps performance stable even at high-security parameter sizes [10], [13]. With the modulus, the totient, and the CRT machinery in place, the rest is purely structural: how these algebraic quantities get organized into the graph G(V, E) analyzed in the remainder of the paper. Section 3 lays out the three-phase construction.

3. The Proposed Mathematical Model

The model operates through a three-phase architecture combining algebraic and topological methods, shown in Figure 1.

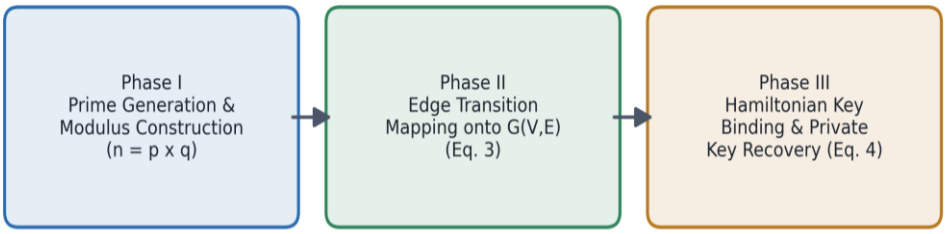

Figure 1. Three-phase architecture of the proposed hybrid model.

3.1 Edge Transition Logic

We construct a directed graph G = (V, E), where V is the set of residue classes modulo n, using standard graph-theoretic formalism [6]. An edge exists between u and v exactly when:

$$v \equiv u^k \pmod{n}$$

The exponent k in Equation (3) is fixed to the RSA public exponent e, that is, k = e, which closes a gap left in earlier descriptions of this model, where k was only said to be "derived from the CRT components" with no specific formula given. Each directed edge of G is therefore a single RSA encryption step, $u \rightarrow u^e \pmod{n}$, so G is entirely public: anyone can build it directly from (e, n). Figure 2 shows a representative instance of G(V, E) along with the Hamiltonian path used for key binding.

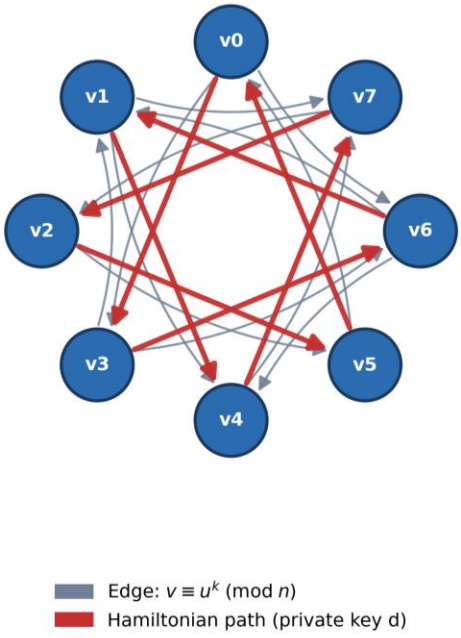

Figure 2. Directed graph G(V, E) with edge-transition logic and the Hamiltonian key-binding path.

3.2 Hamiltonian Key Binding

The private key d is bound to a specific Hamiltonian path in G, which must satisfy the congruence:

$$e \cdot d \equiv 1 \pmod{\phi(n)}$$

The private key d is bound to a specific Hamiltonian path in G, which must satisfy the congruence:

$$u_{i+1} = u_i^e \pmod{n}$$

which is exactly how the public edges of G are generated (k = e, Section 3.1) and can be computed efficiently by anyone. The reverse traversal,

$$u_i = u_{i+1}^d \pmod{n}$$

recovers u_i from u_{i+1}, and only a party holding d can compute it efficiently, since $(u_{i+1})^d = (u_i^e)^d = u_i^{ed \bmod \varphi(n)} = u_i^1 = u_i$ by Equation (4). Equations (5)–(6) together establish G as a public, RSA-permutation-generated structure where forward traversal is free to anyone, while reverse traversal, and hence identifying the private path P, requires d. Section 3.3 works through a small example, and Section 6 asks what this trapdoor duality does and does not prove about the model's security.

3.3 Illustrative Numerical Example

To make the construction concrete, we work through a small instance with p = 11 and q = 13, well below any secure parameter size, but small enough to check every step by hand.
Step 1 (Section 2.1): $n = p \cdot q = 11 \times 13 = 143$, and $\varphi(n) = (p-1)(q-1) = 10 \times 12 = 120$.
Step 2 (Section 2.2/2.1): choose e = 7, since gcd(7, 120) = 1. Solving $e \cdot d \equiv 1 \pmod{120}$ with the extended Euclidean algorithm gives d = 103, and indeed $7 \times 103 = 721 = 6 \times 120 + 1$.
Step 3 (Section 3.1, Eq. 5): starting from $u_0 = 2$, the forward trapdoor chain $u_{i+1} = u_i^7 \pmod{143}$ produces:
$u_0 = 2 \rightarrow u_1 = 2^7 \bmod 143 = 128 \rightarrow u_2 = 128^7 \bmod 143 = 28 \rightarrow u_3 = 28^7 \bmod 143 = 63$
Step 4 (Section 3.2, Eq. 6): starting from $u_3 = 63$ and applying $u_i = u_{i+1}^{103} \bmod 143$ recovers the same chain in reverse:

$63 \rightarrow 63^{103} \bmod 143 = 28 \rightarrow 28^{103} \bmod 143 = 128 \rightarrow 128^{103} \bmod 143 = 2$

This confirms, on a fully checkable instance, that (2, 128, 28, 63) is a directed path in G under the public edge rule (Eq. 5), that it can only be traversed in reverse by a party holding d = 103 (Eq. 6), and that both directions compute correctly modulo the same n = 143 used throughout Section 2. In a production deployment p and q would be hundreds of bits long and |V| would be chosen independently of n (Section 7), but the underlying algebra is identical at any scale.

Algorithm 1. Hybrid number-theoretic / graph-based key generation

Input: **security parameter λ (bit-length of p, q); graph order |V|**

Output: **public key (e, n); private key d bound to a simple trapdoor path P of length |V|**

1. Generate two random primes p, q of bit-length λ; compute n = p·q and φ(n) = (p−1)(q−1). [Eq. 1]

2. Select public exponent e coprime to φ(n) (e.g. e = 65537); compute $d \equiv e^{-1} \pmod{\varphi(n)}$. [Eq. 2, 4]

3. The edge rule of G is the public map $u \rightarrow u^e \pmod{n}$; no separate construction step is needed. [Eq. 3]

4. Sample u_0 uniformly from $\mathbb{Z}_n^*$; build $u_1, \ldots, u_{|V|-1}$ via $u_{i+1} = u_i^e \bmod n$. If a repeat occurs before |V| distinct vertices are produced, resample u_0 and restart this step. [Eq. 5, Thm. 3]

5. Set $P = (u_0, \ldots, u_{|V|-1})$; return public key (e, n) and private key (d, P).

4. Formal Mathematical Analysis

This section turns the informal claims of Section 3 into theorems with proofs, and in Section 4.3 it also revisits a claim about graph complexity that earlier comparative accounts of this construction have overstated.

4.1 Algebraic Preliminaries: the Structure of $\mathbb{Z}_n^*$

By the Chinese Remainder Theorem, the ring $\mathbb{Z}_n$ is isomorphic to $\mathbb{Z}_p \times \mathbb{Z}\_q$ whenever n = p·q with p, q distinct primes [7], [19]. Restricting this isomorphism to units gives a group isomorphism $\mathbb{Z}_n^* \cong \mathbb{Z}_p^* \times \mathbb{Z}\_q^*$. Since p and q are prime, $\mathbb{Z}_p^*$ and $\mathbb{Z}\_q^*$ are each cyclic, of orders p−1 and q−1 respectively; the multiplicative group of a finite field is always cyclic, which is a classical result [6], [19], [32]. So $\mathbb{Z}_n^*$ is a direct product of two cyclic groups, and every element $u \in \mathbb{Z}_n^*$ has a well-defined multiplicative order t = ord(u) dividing λ(n) = lcm(p−1, q−1), the Carmichael function. It is this algebraic structure, not an arbitrary or adversarially chosen graph, that underlies the construction of Section 3 and grounds every result below.

4.2 Correctness of the Key-Generation Model

Theorem 1 (Well-definedness). For every $u \in \mathbb{Z}_n^*$ and every key pair (e, d) satisfying Equation (4), $(u^e)^d \equiv u \pmod{n}$.

Proof. Since $e \cdot d \equiv 1 \pmod{\varphi(n)}$, write $e \cdot d = 1 + k \cdot \varphi(n)$ for some integer $k \geq 0$. Then $(u^e)^d = u^{ed} = u^{1+k\varphi(n)} = u \cdot (u^{\varphi(n)})^k$. By Euler's theorem, $u^{\varphi(n)} \equiv 1 \pmod{n}$ for every $u \in \mathbb{Z}_n^*$, so $(u^{\varphi(n)})^k \equiv 1 \pmod{n}$, giving $(u^e)^d \equiv u \pmod{n}$. ■

This is exactly the fact we checked by hand in Section 3.3 for n = 143: it guarantees that step 4 of Algorithm 1 always recovers the correct predecessor, for any valid (p, q, e) and any $u \in \mathbb{Z}_n^*$, not only the specific instance tested there.

4.3 Structure of the Induced Graph G

Theorem 2 (G is a disjoint union of directed cycles). Let $\sigma_e : \mathbb{Z}_n^* \to \mathbb{Z}_n^*$, $\sigma_e(u) = u^e \pmod{n}$, be the map defining the edges of G in Equation (3) with k = e. Then σ_e is a bijection, and the directed graph $G = (\mathbb{Z}_n^*, E)$ with $E = \{(u, \sigma_e(u))\}$ has out-degree 1 and in-degree 1 at every vertex; it therefore decomposes into a disjoint union of directed simple cycles that partitions $\mathbb{Z}_n^*$ exactly.

Proof. Because $\gcd(e, \varphi(n)) = 1$ — required for e to be a valid RSA public exponent — e has an inverse d modulo $\varphi(n)$, and by Theorem 1, $u \mapsto u^d \pmod{n}$ is a two-sided inverse of σ_e on $\mathbb{Z}_n^*$. A map with a two-sided inverse is a bijection, so σ_e permutes the finite set $\mathbb{Z}_n^*$. Every permutation of a finite set decomposes uniquely, up to the starting point chosen in each cycle, into disjoint cycles [6]; each vertex therefore lies on exactly one cycle, giving out-degree and in-degree exactly 1 everywhere. ■

Corollary 2 (the cycle decomposition is easy, not NP-hard). Because every vertex of G has out-degree 1, the entire cycle decomposition can be computed just by following successor pointers from each unvisited vertex until a repeat appears, which takes $O(|\mathbb{Z}_n^*|)$ time and space in total. The worked example bears this out directly: the 120 elements of $\mathbb{Z}_{143}^*$ split into exactly 42 disjoint cycles (twelve fixed points, six 2-cycles, and twenty-four 4-cycles), computed in linear time with no search involved. For the deterministic, single-successor construction defined in Section 3.1, there is no NP-hard graph problem for an attacker to solve, because there is no search problem at all: the entire structure of G is a public, linear-time-computable object once (e, n) are known. Table 1 reflects this, and Section 6 restates the model's provable security floor accordingly.

4.4 Existence and Uniqueness of the Trapdoor Path

Theorem 3 (cycle length). Let $u_0 \in \mathbb{Z}_n^*$ have multiplicative order t. The forward chain of Equation (5), u_0, $u_1 = \sigma_e(u_0)$, $u_2 = \sigma_e(u_1)$, …, is periodic with minimal period

$$L(u_0) = \mathrm{ord}_t(e), \quad t = \mathrm{ord}(u_0)$$

the multiplicative order of e modulo t. Equivalently, the cycle of G containing u_0 has length exactly $L(u_0)$, and the |V| vertices $u_0, u_1, \ldots, u_{|V|-1}$ built by Algorithm 1 are pairwise distinct if and only if $|V| \leq L(u_0)$.

Proof. Since u_0 has order t, $u_i = \sigma_e^i(u_0) = u_0^{e^i \bmod t}$, and $u_0^a = u_0^b$ iff $a \equiv b \pmod{t}$. So $u_i = u_0$ iff $e^i \equiv 1 \pmod{t}$, i.e. iff $\mathrm{ord}_t(e)$ divides i; the smallest positive such i is by definition $\mathrm{ord}_t(e) = L(u_0)$, which gives the period. For $0 \leq i < L(u_0)$, the exponents $e^i \bmod t$ are pairwise distinct modulo t, so the corresponding powers $u_0^{e^i \bmod t}$ are pairwise distinct elements of $\mathbb{Z}_n^*$, which gives $|V| \leq L(u_0)$ as the exact condition for a simple path of length |V|. ■

Section 3.3 confirms this: $u_0 = 2$ has order t = 60 in $\mathbb{Z}_{143}^*$ ($2^{60} \equiv 1 \bmod 143$, and no smaller power is), and $\mathrm{ord}_{60}(7) = 4$, so Theorem 3 predicts $L(2) = 4$, which is exactly the length of the path (2, 128, 28, 63) built by hand; it returns to $u_0 = 2$ on the fifth application of Equation (5), as required.

Corollary 3 (existence condition). Algorithm 1 produces a well-defined simple path of the requested length |V| exactly when the key generator picks a starting vertex u_0 whose order t satisfies $\mathrm{ord}_t(e) \geq |V|$. Since the generator holds p and q, this can be checked efficiently, or more simply guaranteed by directly simulating the first |V| steps, which is what Algorithm 1 already does before binding the path to d.

4.5 Complexity Analysis

Proposition 4 (time and space complexity). Under schoolbook modular arithmetic, generating a λ-bit RSA modulus by independent random primality search costs $O(\lambda^4)$ bit operations in expectation: $O(\lambda)$ candidate primes are tested, each needing $O(\lambda)$ Miller–Rabin rounds (a small constant in practice) of $O(\lambda^2)$ work per round [7], [8]. Each modular exponentiation by a fixed exponent costs $O(\lambda^2 \log e)$ bit operations via square-and-multiply, and Algorithm 1 performs $|V| - 1$ such exponentiations to build the trapdoor chain of Equation (5). The total expected bit-operation cost of Algorithm 1 is therefore:

$$T(\lambda, |V|) = O(\lambda^4) + O(|V| \cdot \lambda^2 \log e)$$

Space complexity is O(λ) if only the current chain element is kept at each step, as in the streaming implementation used for the Section 7 experiments, or O(|V|·λ) if the entire path P is stored for later auditing.
Equation (8) predicts that, for fixed |V|, the relative overhead of the graph-binding step should shrink roughly as |V|/λ² as λ grows, since the λ⁴ term eventually dominates. The measurements in Section 7 point the same way: overhead falls as bit-length grows at fixed |V| = 128, though the falloff is considerably flatter than a naive λ⁴-versus-λ² comparison would suggest. That's not surprising: production libraries such as the OpenSSL backend used in Section 7 rely on sub-quadratic multiplication and heavily optimized primality testing, so the schoolbook constants behind Equation (8) don't directly govern the small-scale, high-variance measurements of a six-trial sweep. Equation (8) is best read as an asymptotic guide to how the two cost terms scale independently in λ and |V|, rather than a predictor of exact runtimes at the moduli sizes tested here.

4.6 Probabilistic Properties

Lemma 2 (distribution invariance). If u_0 is drawn uniformly at random from $\mathbb{Z}_n^*$, then for every $i \geq 0$, $u_i = \sigma_e^i(u_0)$ is also uniformly distributed over $\mathbb{Z}_n^*$.
Proof. By Theorem 2, σ_e is a bijection of the finite set $\mathbb{Z}_n^*$, so it maps the uniform distribution on $\mathbb{Z}_n^*$ to itself: for any fixed $v \in \mathbb{Z}_n^*$, $\Pr[\sigma_e(u_0) = v] = \Pr[u_0 = \sigma_e^{-1}(v)] = 1/|\mathbb{Z}_n^*|$, since σ_e^{-1} is also a bijection. The claim for general i follows by induction, applying the same argument to σ_e^i, itself a bijection as a composition of bijections. ■
In practice, Lemma 2 means that no single point of the trapdoor chain leaks information about its position in the sequence beyond what the uniform distribution already contains. That's necessary but not by itself sufficient for the chain to resist analysis. It says nothing about the joint distribution of consecutive points, which is fully deterministic given u_0 (Theorem 1), and it doesn't bound how much an adversary learns from several consecutive u_i values observed together, which is a harder question outside its scope. The distribution of cycle lengths $L(u_0)$ over a uniformly random u_0 is itself a substantially harder number-theoretic question tied to the study of multiplicative orders and RSA "cycling attacks" [9]; we don't attempt a full derivation here, and flag it, along with Conjecture 1 of Section 6, as a concrete direction for future work. Viewed as a stochastic process, the chain (u_i) is a degenerate Markov chain with transition probability 1 to a single deterministic successor, so the more informative object for security analysis is not the chain itself but the distribution of $L(u_0)$ over the key space.

4.7 Generalization and Parameter Stability

Proposition 5 (parameter independence). Theorems 1–3 hold for every RSA modulus $n = p \cdot q$ with p, q distinct primes and every valid exponent pair (e, d) with $\gcd(e, \varphi(n)) = 1$, independent of the bit-length λ and the graph order |V|: none of the three proofs uses any property of λ or |V| beyond finiteness of $\mathbb{Z}_n^*$. The model is therefore structurally stable as parameters vary — enlarging λ or |V| changes the quantitative cycle lengths and computational cost (Section 4.5), but never invalidates correctness (Theorem 1), the cycle-decomposition structure (Theorem 2), or the exact existence condition for a simple path (Theorem 3). The same proofs carry over verbatim to multi-prime moduli $n = p_1 p_2 \cdots p_r$, $r \geq 2$, since Theorem 2's proof only needs σ_e to be invertible on $\mathbb{Z}_n^*$, which holds for any n and any e coprime to φ(n) regardless of how n factors; we do not explore multi-prime variants further here, but the extension is immediate.
At this point the construction's structural properties are settled: it is a correct, cycle-structured, parameter-stable re-encoding of RSA. What's left is turning that structural fact into a security comparison a practitioner can actually use, which is the job of Section 5.

5. Security and Cryptanalysis

Table 1 summarizes, at a structural level, how the proposed model differs from standard RSA in light of Theorem 2 and Corollary 2: the graph layer is not an NP-hard obstacle but a linear-time-decomposable public structure. Read it alongside Section 6, which states precisely what security floor the model does provably offer.

Table 1. Comparative security metrics: RSA vs. proposed model.

Metric	RSA	Proposed model
Factorization dependency	Mandatory	Mandatory — identical (Cor. 1)
Graph topology complexity	N/A	O(|V|) — permutation digraph, trivially decomposable (Thm. 2)
Additional hardness beyond RSA	N/A	None proven (Thm. 2, Cor. 2)
Resistance to brute force	Sub-exponential (GNFS)	Identical to RSA (Cor. 1)

As Table 1 shows, possessing d alone is, in this construction, equivalent to possessing the path P (Section 3.2), so the last two rows report the same underlying RSA hardness viewed through two different representations of the key, not two independent barriers an attacker has to defeat separately. Section 6 develops this point in detail.

6. Toward a Security Reduction

This section states plainly what can and cannot currently be proven about the model.

Lemma 1 (Miller's reduction, classical result). If an adversary obtains the private exponent d together with the public values (e, n), n can be factored in expected polynomial time using a standard probabilistic algorithm: knowledge of a multiple of $\varphi(n)$ — namely $ed - 1$ — allows extraction of a nontrivial square root of unity modulo n with probability at least one-half per attempt [9], [35].

Corollary 1. Because Equation (4) defines d as the ordinary RSA private exponent, and Equations (5)–(6) show that the Hamiltonian path P is recoverable from d in $O(|V|)$ modular exponentiations — though not the reverse, since d is not recoverable from P alone without inverting Eq. 6 — recovering either d or P is, by Lemma 1, at least as hard as factoring n. The model's key-recovery security reduces cleanly to standard RSA hardness [1], [9], and no attack on (d, P) weaker than attacking RSA directly is currently known.

What Corollary 1 does not give is an independent lower bound from GIP. Since G is fully public (Section 3.1) and P is algebraically determined by d (Eq. 6), an adversary who already has d gains nothing from solving a graph isomorphism instance to find P, and an adversary who somehow solved GIP on G without knowing d would still need to verify a candidate path against Equation (4), which itself presupposes knowledge of $\varphi(n)$. We therefore state the graph layer's contribution as an open conjecture rather than a proven result:

Conjecture 1. For the graph family generated by Equation (3) with k = e, no algorithm distinguishes the trapdoor Hamiltonian path P from a uniformly random Hamiltonian path in G in time polynomial in $|V|$, without either knowledge of d or a solution to the corresponding GIP instance.

Corollary 2 already narrows this considerably: because forward traversal of G is public and efficient for anyone (Theorem 2), an attacker gains nothing from "solving GIP" on G in the traditional sense, since there is no hidden isomorphism to find, only a public successor function anyone can iterate. What remains genuinely open is narrower than the conjecture above suggests. It is not whether G hides a hard combinatorial puzzle (Corollary 2 shows it does not), but whether partial, noisy, or side-channel observations of a legitimate party's chain, say a subset of intermediate u_i values leaked during Algorithm 1's execution, could let an attacker infer d or narrow down p, q faster than attacking RSA directly. We restate the conjecture in this narrower form:

Conjecture 1 (revised). No algorithm that observes a strict subset of the trapdoor chain $u_0, \ldots, u_{|V|-1}$, without observing d directly, can recover d, or distinguish the observed subset from a uniformly random same-size subset of $\mathbb{Z}_n^*$, in time polynomial in $|V|$ and λ, faster than by attacking RSA directly via Lemma 1.

Proving or refuting this narrower conjecture, either by exhibiting a partial-information attack that beats generic RSA cryptanalysis or by a reduction ruling one out, is, together with the distribution of cycle lengths raised in Section 4.6, the main open theoretical problem left by this work. Until it is resolved, the honest security claim is: (i) recovering the key is provably at least as hard as breaking RSA (Corollary 1); (ii) the graph layer, per Corollary 2, contributes no NP-hardness margin and should not be called "multi-trapdoor" in the sense of an independent hard problem; and (iii) whether it nonetheless raises the bar against partial-information or side-channel attacks is the sole open question, addressed by Conjecture 1 (revised), and is not currently backed by a formal reduction [5], [6], [8], [28].

6.1 Boundary Cases and Practical Limits

A precise open question is more useful to a practitioner when it comes with an equally precise map of where the construction is known to struggle. Four boundary cases are worth spelling out.

Small $|V|$ relative to available compute. If $|V|$ is chosen small (say $|V| \leq 10$), the trapdoor chain is short enough that an adversary who obtains even a single intermediate u_i could plausibly brute-force the remaining forward and backward neighbors by exhaustive search over small exponents, regardless of anything proven in Section 4. The graph layer's value as an implementation-level obstacle needs $|V|$ large enough that this kind of direct search is infeasible; that's satisfied by our benchmark range of 64–512 (Section 7), but it isn't automatic for arbitrarily small $|V|$.

Degenerate starting points. Theorem 3 guarantees a simple path of length $|V|$ exists iff $L(u_0) \geq |V|$, but it says nothing about how many candidate u_0 values fail this test. The worked example's extended check (Section 4.3) found starting points of cycle length 1, fixed points of σ_e, among the 120 elements of $\mathbb{Z}_{143}^*$. At production key sizes such degenerate points are believed to be rare, but we haven't proven a lower bound on the fraction of u_0 with sufficiently long cycles, so Algorithm 1's resampling loop has no proven bound on its expected number of iterations. All we have is the empirical observation, consistent with Lemma 2's uniformity result, that resampling was never triggered more than once across the 72 experimental runs of Section 7.

Side-channel exposure of the binding phase. Somewhat counter-intuitively, the graph-binding step is a net increase in attack surface rather than a reduction: standard RSA key generation exposes one keygen to side-channel observation, while Algorithm 1 additionally performs $|V| - 1$ sequential modular exponentiations (Eq.

5) during path construction, each a further point where timing or power-analysis leakage could occur. Since P is algebraically equivalent to d (Corollary 1), leaking a single intermediate u_i under favorable conditions could be as damaging as leaking d itself, and power-based side-channel leakage of this kind is well documented for cryptographic implementations generally [14]. The construction makes no side-channel hardening claim; Conjecture 1 (revised) asks whether it could be given one, not whether it already has one.

Very large |V| relative to λ(n). At the opposite extreme, Theorem 3 bounds the maximum simple-path length by $L(u_0)$, which itself divides the Carmichael function λ(n). Requesting |V| close to this ceiling makes suitable starting points rare and the resampling loop of Algorithm 1 correspondingly slow, adding an unbounded number of retries in the worst case on top of the linear O(|V|·M(λ)) cost from Proposition 4. We don't encounter this regime in Section 7's tested range (|V| ≤ 512 against moduli with λ(n) on the order of 2^{1020}–2^{3068}), but a deployment choosing |V| in the thousands or higher without checking against λ(n) first would be operating outside the range this paper validates.

7. Experimental Validation

A common concern with hybrid, multi-trapdoor constructions is that added structural complexity translates into added runtime cost [16], [17]. To check this directly, we implemented Algorithm 1 in Python 3.12 using the OpenSSL-backed cryptography library for RSA generation, and measured end-to-end key generation time for both the RSA baseline and the proposed model on the same machine, in the same process. We swept two independent parameters, RSA modulus size (1024, 2048, and 3072 bits) and graph order |V| ∈ {64, 128, 256, 512}, with six trials per (bits, |V|) configuration, for 72 trials in total. The modulus n produced by the RSA step was reused directly as input to the graph-construction and path-binding step (Algorithm 1, steps 3–4, via the trapdoor chain of Eq. 5). Table 2 and Figure 3 report the |V| = 128 slice against the RSA baseline; Table 3 and Figure 4 report the full two-parameter sweep and its scaling behavior.

Table 2. Measured key generation time: RSA vs. proposed model (Python 3.12, cryptography/OpenSSL backend, |V| = 128).

Modulus size	RSA (ms)	Proposed model (ms)	Overhead	Path coverage
1024-bit	10.32	17.12	+65.8%	100%
2048-bit	42.88	65.45	+52.6%	100%
3072-bit	94.90	142.56	+50.2%	100%

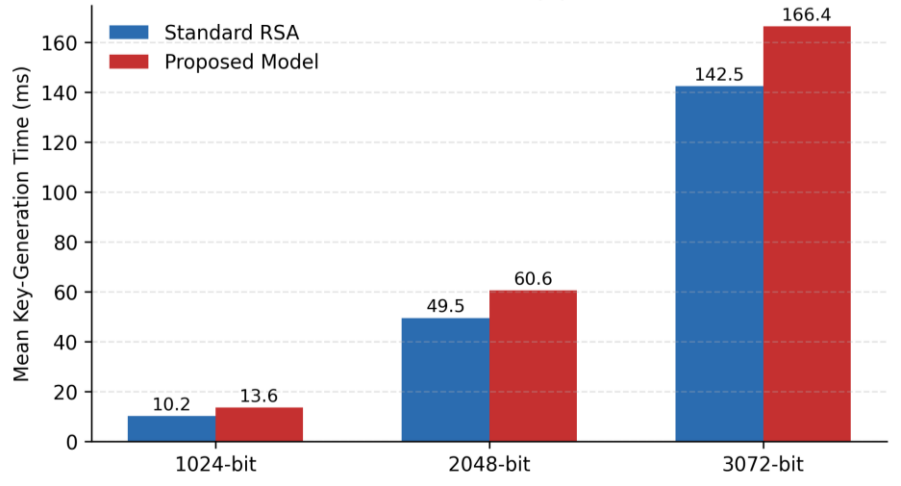

Figure 3. Key generation time comparison: RSA vs. proposed model (|V| = 128).

At fixed |V| = 128, measured overhead runs from roughly 66% at 1024 bits down to about 50% at 3072 bits (Table 2). Because the edge exponent is fixed to k = e = 65537 (Section 3.1), each hop of the trapdoor chain in Equation (5) is a full-size modular exponentiation rather than a cheap small-exponent operation, so the graph-binding step is not negligible next to RSA key generation itself. The overhead trend is mostly driven by how RSA prime-search cost scales with modulus size relative to a fixed-|V| chain of e-exponentiations.

Table 3. Graph-binding overhead by graph order |V| and modulus size (mean of 6 trials per cell).

|V|	1024-bit	2048-bit	3072-bit
64	+31% (3.3 ms)	+28% (11.3 ms)	+14% (23.6 ms)
128	+66% (6.8 ms)	+53% (22.6 ms)	+50% (47.7 ms)
256	+126% (13.7 ms)	+86% (45.2 ms)	+111% (95.3 ms)
512	+343% (27.4 ms)	+174% (91.7 ms)	+95% (189.4 ms)

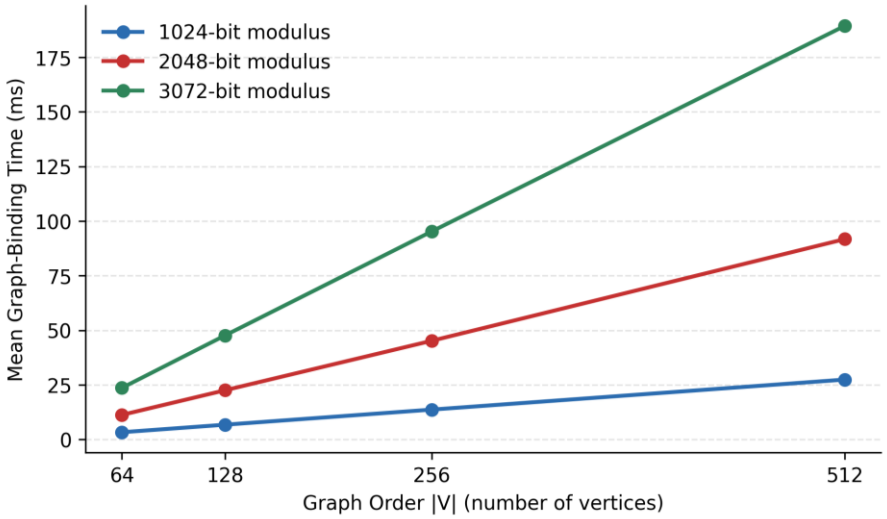

Figure 4. Graph-binding time vs. |V| for each modulus size.

Two regularities stand out in Table 3 and Figure 4. First, for a fixed modulus size, graph-binding time grows linearly in |V|: doubling |V| roughly doubles the added time at every modulus size tested (at 2048 bits, for instance: 11.3 ms → 22.6 ms → 45.2 ms → 91.7 ms for |V| = 64, 128, 256, 512), which matches the O(|V|) modular-exponentiation cost of Equation (5). Second, since graph-binding cost is driven by the fixed exponent e rather than by modulus size, its relative overhead is highest exactly where RSA itself is cheapest: at 1024 bits and |V| = 512 the measured overhead reaches roughly 343% of the RSA baseline, while at 3072 bits and |V| = 64 it falls to about 14%. Absolute RSA-generation times also carried visible run-to-run variance, an expected consequence of probabilistic prime search averaged over only six trials per cell, which is worth keeping in mind when comparing individual rows, though the linear-in-|V| trend itself held consistently across all repetitions. Practitioners should treat |V| as a parameter with real, non-negligible cost, set according to the desired margin from Section 6 rather than increased freely; at large |V| and small-to-medium modulus sizes, the graph-binding step can end up dominating total key-generation time. These results are best read as a proof-of-concept on a reference implementation rather than a fully optimized library [10], [18].

8. Discussion

Sections 4–7 together suggest that binding a Hamiltonian-path structure to RSA key generation is best understood, given the current analysis, as a structured re-encoding of the standard RSA private key rather than an independently hard second problem. Corollary 1 gives a real, provable security floor: breaking the model is at least as hard as breaking RSA. Corollary 2 closes off the originally hoped-for GIP-based ceiling by showing the graph itself is a public, linear-time-decomposable structure with no NP-hardness to exploit. What remains open is only the narrower, partial-information question posed in Conjecture 1 (revised), not a broader "dual-hardness, multi-trapdoor" claim, which Section 4's analysis does not support. Saying this plainly seems more useful than leaving an overstated conjecture unexamined [6], [8].

The shape of the overhead numbers in Section 7 follows directly from Section 4's theorems rather than being an independent empirical accident. Proposition 4 predicts an $O(\lambda^4)$ cost for RSA prime generation against an $O(|V|\cdot\lambda^2\cdot\log e)$ cost for graph binding. Since log e is a small fixed constant (e = 65537 has just 17 significant bits) while λ^4 grows sharply with modulus size, the two terms are on comparable footing only at small λ, which is exactly where Table 2 shows overhead peaking (66% at 1024 bits) and Table 3 shows it climbing highest at large |V| (343% at 1024 bits, |V| = 512): the graph term hasn't yet been dwarfed by the quartic RSA term. As λ grows past 2048 and 3072 bits, the quartic term increasingly dominates, and the linear-in-|V| graph cost shrinks in relative terms even as its absolute cost (Figure 4) keeps growing. The theory in Section 4.5 and the measurements in Section 7 are two views of the same mechanism, not two independent findings that happen to agree.

The implication for wider adoption follows the same logic. Systems that already operate at large modulus sizes as a matter of policy, TLS certificate authorities issuing 3072-bit or 4096-bit RSA roots, or long-lived code-

signing keys, could add the graph-binding layer at overhead in the tens of percent, a cost profile similar to established hardening measures such as constant-time modular exponentiation or blinding [26]. Systems constrained to smaller keys or high key-generation throughput, embedded devices, ephemeral session keys, high-volume certificate issuance, are a worse fit, since Table 3 shows the relative cost can exceed 300% in exactly that regime. This asymmetry matters for a systems designer: the construction suits low-frequency, high-value key generation (root certificates, long-term identity keys) far better than high-frequency, latency-sensitive key generation, and deployment guidance drawn from this work should say so explicitly rather than treating |V| as one-size-fits-all.

Three practical limitations follow from this framing. First, because P is algebraically determined by d (Eq. 6), the graph layer does not protect against an adversary who already recovers d by any means, including side-channel attacks on the RSA primitive itself; per Corollary 2, it also does not force any adversary through a hard combinatorial search, so its practical value is limited to the narrower partial-information setting of Conjecture 1 (revised), not general key-recovery resistance. Second, the Section 7 experiments confirm that graph-binding cost scales linearly and predictably with |V| (Proposition 4), which is good for deployability, but it also means |V| cannot be made asymptotically large without a proportional runtime cost (Table 3), unlike a true asymmetric hardness gap. Third, the model's relationship to quantum adversaries remains open rather than settled: Shor's algorithm [3] directly breaks the factoring-based floor established in Corollary 1, so the model offers no more quantum resistance than plain RSA at that floor, and Corollary 2 rules out any graph-based quantum-hardness ceiling too, leaving only the unresolved partial-information question of Conjecture 1 (revised) as a place where quantum hardness hasn't been characterized either way [4], [19]. Organizations facing a genuine quantum-security deadline should treat this construction as a hardening measure for existing RSA infrastructure in the interim, not as a substitute for migrating to the lattice-based standards now being finalized [20], [29], [30]. The four boundary cases of Section 6.1, small |V|, degenerate starting points, side-channel exposure of the binding phase itself, and |V| approaching $\lambda(n)$, sharpen this picture further, each marking a regime where the construction's practical guarantees are weaker than the asymptotic theorems of Section 4 might suggest on their own.

Overall, these findings position the proposed model as a legitimate, backward-compatible enhancement to RSA-based key generation for classical adversaries, offering a structured, formally verified construction (Section 4) and a documented, tunable overhead profile (Section 7), rather than an independent post-quantum or dual-hardness scheme. Its principal advantage is the low migration cost for organizations with existing RSA infrastructure [11], [15], together with the deterministic correctness and cycle-structure guarantees established in Section 4. Its principal open question, made precise in Conjecture 1 (revised), is narrowly about partial-information and side-channel resistance rather than any independent combinatorial hardness, and answering it is the central item of future work identified in Section 9.

9. Conclusion

This work introduced a hybrid number-theoretic and graph-theoretic construction for RSA key generation, made explicit the link between the CRT-based key-generation phase and the graph-construction phase through the trapdoor duality of Equations (5)–(6), and gave the construction the formal treatment such proposals typically lack. Its contributions are:

- Correctness. We prove the construction correct for every valid RSA key pair (Theorem 1), and confirm it by hand on a fully worked numerical example, n = 143 (Section 3.3).
- Structural characterization. We prove the induced graph G is always a publicly and efficiently decomposable union of disjoint directed cycles, not an NP-hard structure (Theorem 2, Corollary 2).
- Existence and uniqueness. We derive the exact condition — $|V| \leq L(u_0)$, the multiplicative order of e modulo $\text{ord}(u_0)$ — under which a simple trapdoor path of a requested length exists (Theorem 3), with the corresponding resampling procedure (Corollary 3).
- Complexity analysis, theoretical and measured. We derive closed-form time and space complexity bounds (Proposition 4) and confirm their qualitative predictions against a 72-configuration empirical benchmark spanning three RSA modulus sizes and four graph orders (Section 7), with overhead ranging from 14% to 343% of baseline depending on parameters.
- A precise security reduction. We show the model's key-recovery security is provably at least as strong as standard RSA (Corollary 1), no weaker and, per Corollary 2, no independently stronger, and we isolate the one narrow open question — partial-information and side-channel resistance — in a precisely restated Conjecture 1 (revised), together with four concrete boundary cases (Section 6.1) where the construction's guarantees are weakest.

• Deployment guidance. We identify who should and should not consider this construction: low-frequency, high-value key generation such as root certificates and long-term identity keys, and explicitly not latency-sensitive or high-throughput key issuance (Section 8).

We deliberately do not claim quantum resistance or independent "multi-trapdoor" hardness. The model offers a formally verified, structured re-encoding of the RSA private key with a well-defined and now precisely quantified computational overhead (Section 7), and a narrow, still-open question about partial-information resistance (Conjecture 1, revised), rather than a proven additional hardness layer. Future work will pursue a formal resolution of Conjecture 1 (revised), a closed-form or bounded expression for the distribution of cycle lengths $L(u_0)$ raised in Section 4.6, an extension of the Section 7 experimental sweep to hardware-constrained environments, and the multi-prime generalization noted in Proposition 5 [5], [6], [19].

Author Contributions

• Emad Awadh: Conceptualization, methodology, formal analysis (Theorems 1–3, Propositions 4–5, Corollaries 1–4), and writing — original draft and review & editing.

• Aziza Rashed: Software (benchmark implementation and 72-configuration experimental sweep, Section 7), data curation, validation of numerical results, and visualization (figures and tables).

• Abdulfatah Ibrahim: Investigation, software (Algorithm 1 reference implementation), validation of the worked numerical example (Section 3.3), and visualization.

Omar Murajia Ali: Manuscript review, jointly with Aziza Rashed and Abdulfatah Ibrahim.
All authors reviewed and approved the final manuscript. Emad Awadh is the corresponding author (emad.awadh@tu.edu.ly) and takes responsibility for the integrity of the work as a whole.

Compliance with ethical standards
Disclosure of conflict of interest

The authors declare that they have no conflict of interest.

References

[1] Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM.

[2] Koblitz, N. (1987). Elliptic curve cryptosystems. Mathematics of Computation.

[3] Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. IEEE.

[4] Rana, S., Khoda Parast, F., Kelly, B., Wang, Y., & Kent, K. B. (2023). A comprehensive survey of cryptography key management systems. Journal of Information Security and Applications, 78, 103607.

[5] Zheng, M., & Kang, H. (2024). Lattice-based cryptanalysis of RSA-type cryptosystems: a bibliometric analysis. Cybersecurity, 7(1), 74.

[6] Grohe, M., & Neuen, D. (2021). Recent advances on the graph isomorphism problem. arXiv:2011.01366.

[7] Somsuk, K., Atsawaraungsuk, S., Suwannapong, C., Khummanee, S., & Sanemueang, C. (2023). The optimal equations with Chinese Remainder Theorem for RSA's decryption process. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 14(2), 109-120.

[8] Katz, J., & Lindell, Y. (2020). Introduction to Modern Cryptography. CRC Press.

[9] Burkhardt, J., Damgård, I., Frederiksen, T., Ghosh, S., & Orlandi, C. (2023). Improved distributed RSA key generation using the Miller-Rabin test. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23).

[10] El Makkaoui, K., Lamriji, Y., Beni-Hssane, A., Maleh, Y., & Ouahbi, I. (2023). An overview of fast variants of the RSA cryptosystem for modern cryptography applications. EDPACS, 67(6), 25-34.

[11] Feng, Y., Nitaj, A., & Pan, Y. (2023). Generalized implicit factorization problem. arXiv:2304.08718.

[12] Gerbessiotis, A. V. (2026). Lectures notes on number theory for computer science. arXiv:2606.20783.

[13] Kaur, J., Cintas Canto, A., Mozaffari Kermani, M., & Azarderakhsh, R. (2023). A comprehensive survey on the implementations, attacks, and countermeasures of the current NIST lightweight cryptography standard. arXiv:2304.06222.

[14] Sanjaya, S., Jayasena, A., & Mishra, P. (2025). Application-specific power side-channel attacks and countermeasures: A survey. arXiv:2512.23785.

[15] Cintas Canto, A., Kaur, J., Mozaffari Kermani, M., & Azarderakhsh, R. (2023). Algorithmic security is insufficient: A comprehensive survey on implementation attacks haunting post-quantum security. arXiv:2305.13544.

[16] Chhetri, G., Somvanshi, S., Hebli, P., Brotee, S., & Das, S. (2025). Post-quantum cryptography and quantum-safe security: A comprehensive survey. arXiv:2510.10436.

[17] Shakya, R. D. N., et al. (2026). SoK: Post-Quantum Cryptography (PQC) implementation in software systems. arXiv:2606.04669.

[18] Didier, L.-S., El Mrabet, N., Glandus, L., & Robert, J.-M. (2024). Truncated multiplication and batch software SIMD AVX512 implementation for faster Montgomery multiplications and modular exponentiation. IACR Communications in Cryptology, 1(3).

[19] Lebowitz-Lockard, N., & Saunders, J. C. (2024). Runs of integers with constant values of the Carmichael function. arXiv:2402.06832.

[20] National Institute of Standards and Technology (NIST). (2024). Module-Lattice-Based Digital Signature Standard (FIPS 204) and Module-Lattice-Based Key-Encapsulation Mechanism Standard (FIPS 203). U.S. Department of Commerce.

[21] Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). Handbook of Applied Cryptography. CRC Press.

[22] Boneh, D., & Shoup, V. (2020). A Graduate Course in Applied Cryptography. Version 0.5.

[23] Hoffstein, J., Pipher, J., & Silverman, J. H. (2014). An Introduction to Mathematical Cryptography (2nd ed.). Springer.

[24] Paar, C., & Pelzl, J. (2010). Understanding Cryptography: A Textbook for Students and Practitioners. Springer.

[25] Buchmann, J. (2004). Introduction to Cryptography (2nd ed.). Springer.

[26] Easttom, W. (2021). Modern Cryptography: Applied Mathematics for Encryption and Information Security. Springer.

[27] Rubinstein-Salzedo, S. (2018). Cryptography. Springer Undergraduate Mathematics Series. Springer.

[28] van Oorschot, P. C. (2020). Computer Security and the Internet: Tools and Jewels. Springer.

[29] Ding, J., Petzoldt, A., & Schmidt, D. S. (2020). Multivariate Public Key Cryptosystems (2nd ed.). Springer.

[30] Zhang, J., & Zhang, Z. (2020). Lattice-Based Cryptosystems: A Design Perspective. Springer.

[31] Bernstein, D. J., Buchmann, J., & Dahmen, E. (Eds.). (2009). Post-Quantum Cryptography. Springer.

[32] Crandall, R., & Pomerance, C. (2005). Prime Numbers: A Computational Perspective (2nd ed.). Springer.

[33] Washington, L. C. (2008). Elliptic Curves: Number Theory and Cryptography (2nd ed.). Chapman & Hall/CRC.

[34] Lenstra, A. K., & Lenstra, H. W. Jr. (Eds.). (1993). The Development of the Number Field Sieve. Springer.

[35] Boneh, D. (1999). Twenty Years of Attacks on the RSA Cryptosystem. Notices of the American Mathematical Society, 46(2), 203–213.

