A CRYPTOGRAPHIC ONTOLOGY OF NON-CANONICITY

Foundations, Formalization, and Implementations

Technical–Scientific Report
Foundational Research Document

São Paulo — 2026

Marcos Elias*

ABSTRACT

This handbook presents a cryptographic architecture grounded not in computational inversion hardness, nor in information-theoretic secrecy, but in **controlled non-identifiability**: a regime in which public data is complete, deterministic, and verifiable, yet **structurally underdetermining** with respect to global identity.

The construction arises from the mathematical theory of **mock modular and mock theta functions**, whose defining characteristic is that their holomorphic part—fully observable and computable—does not canonically determine their global completion. Identity emerges only after the introduction of additional, non-local data (the *shadow* and normalization), which cannot be recovered through local evaluation or finite observation.

We formalize this phenomenon as a cryptographic principle, define attacker models based on **restricted observables**, and introduce a family of problems (MMIP: Mock Modular Identification Problems) that replace inversion with identification under ambiguity. We then show how this epistemic asymmetry survives discretization and truncation, enabling concrete implementations over finite fields, with verifiable transcripts, consistency operators, and reference constructions for key exchange and signatures.

This work does not claim absolute quantum immunity. Instead, it establishes an architecture **orthogonal** to known quantum attack paradigms (Shor, Grover, HSP reductions), and proposes a diversified cryptographic foundation suitable for a post-quantum world.

READER'S MAP

This document is written for three audiences simultaneously:

- **Mathematicians** will find a precise articulation of non-canonicity, torsor structures, and global-versus-local information asymmetry, with careful avoidance of informal impossibility claims.
- **Cryptographers and engineers** will find explicit threat models, attacker games, discretization strategies, transcript formats, and implementable primitives.
- **Qualified investors and technical decision-makers** will find a clear separation between foundational truth, engineering artifacts, and economic optionality.

No prior familiarity with mock modular forms is assumed beyond graduate-level mathematical maturity.

NOTATION AND CONVENTIONS

Throughout this work:

- $\mathbb{H}$ denotes the upper half-plane $\{\tau \in \mathbb{C} : \mathrm{Im}(\tau) > 0\}$
- $q = \exp(2\pi i \tau)$
- $\Gamma \subseteq SL(2,\mathbb{Z})$ denotes a congruence subgroup
- k denotes modular weight
- $f(\tau)$ denotes the holomorphic mock modular form
- $g(\tau)$ denotes its shadow (a cusp form)
- $R_g(\tau)$ denotes the non-holomorphic completion term
- $F(\tau) = f(\tau) + R_g(\tau)$ denotes the completed object
- "Transcript" always refers to **finite, public, verifiable data**
- "Completion data" always refers to **global, private, identity-selecting data**

We avoid categorical claims of impossibility. All security statements are made **relative to explicit access models**.

PART I — THE ONTOLOGICAL CRISIS OF CRYPTOGRAPHY

CHAPTER 1 — THE HIDDEN AXIOM OF PUBLIC-KEY CRYPTOGRAPHY

1.1 The Unspoken Premise

All modern public-key cryptography, from RSA to lattice-based post-quantum candidates, shares a premise so fundamental that it is almost never stated:

Public data uniquely determines the secret.

Security, under this premise, is achieved by ensuring that recovering the secret from the public data is computationally infeasible.

This premise holds for:

- RSA: the modulus N uniquely determines (p, q)
- ECC: the public point Q uniquely determines the scalar d
- Lattice schemes: the public basis uniquely determines the secret vector

The difference between schemes lies only in *how hard* inversion is believed to be.

This is not a computational assumption.
It is an **epistemic assumption**.

1.2 Shor Did Not Create the Crisis

Shor's algorithm did not weaken RSA or ECC by making factoring or discrete logs "faster". It exposed something deeper:

The secrecy of these schemes depends entirely on the assumption that inversion is the only way to learn the secret.

Once inversion is efficient, secrecy collapses completely.

The real fragility is therefore not computational—it is **ontological**.

1.3 Two Ways a System Can Fail

A cryptographic system can fail in two fundamentally different ways:

1. **Computational failure**
   An adversary finds a faster algorithm for inversion.
2. **Ontological failure**
   The system reveals that the public data *already contains* the secret uniquely.

Shor's algorithm is merely a catalyst that forces us to confront the second type of failure.

CHAPTER 2 — THREE REGIMES OF SECURITY

2.1 Regime I: Information-Theoretic Secrecy

The public data contains *zero information* about the secret.

Example: One-time pad.

This regime is absolute, but operationally restrictive.

2.2 Regime II: Computational Inversion

The public data uniquely determines the secret, but inversion is computationally hard.

Examples:

- RSA
- ECC
- Lattice-based PQC
- Isogeny-based schemes

Quantum computing threatens this regime directly.

2.3 Regime III: Controlled Non-Identifiability

The public data does **not** uniquely determine the secret.

Instead:

- Multiple mathematically valid secrets are consistent with the public data.
- The correct secret is selected only by additional, global information.
- This information is categorically different from the public observables.

This regime is neither Shannon secrecy nor inversion hardness.

It is **epistemic asymmetry by construction**.

2.4 Why Regime III Is Not “Obfuscation”

This regime is often misunderstood as “hiding more cleverly”.

It is not.

The public transcript is:

- Deterministic
- Verifiable
- Finite

- Complete in its own category

What it is **not** is canonically identifying.

This distinction is the core of the entire architecture.

CHAPTER 3 — RAMANUJAN'S INTERRUPTION

3.1 The Mathematical Anomaly

Ramanujan's mock theta functions were not proposed as part of a theory. They appeared as finished objects that **refused classification**.

They had:

- Well-defined q-expansions
- Stable local behavior
- Predictable asymptotics

And yet:

- They were not modular
- They failed closure globally
- They resisted canonical identification

For decades, mathematics could *use* them but could not *explain* them.

This is not an accident. It is a signal.

3.2 Zwegers' Resolution: Incompleteness, Not Failure

Zwegers showed that mock theta functions are the **holomorphic parts** of larger objects.

The missing piece is:

```
R_g(τ) = ∫_{-τ̄}^{i∞} g(z) / (z + τ)^k dz
```

Where g(τ) is a cusp form (the *shadow*).

This term is:

- Global
- Non-holomorphic
- Not reconstructible from local data

The holomorphic object is not broken.
It is **incomplete by design**.

3.3 Zagier's Reframing: Non-Canonicity

Zagier emphasized the crucial conceptual point:

The holomorphic data does not canonically determine the completion.

Even with infinite local precision, the identity is not fixed.

Multiple completions coexist, forming a **structured ambiguity class**.

This is the exact opposite of classical cryptography's premise.

CHAPTER 4 — THE CORE CRYPTOGRAPHIC INSIGHT

4.1 Observation Does Not Imply Identification

Mock modular phenomena demonstrate a principle that cryptography has never exploited:

Local observability does not entail global identifiability.

Two objects may:

- Agree arbitrarily well on all observable data
- Differ globally in a way that only non-local information resolves

This is not noise.
It is structure.

4.2 The New Security Primitive

The cryptographic primitive is not a function to invert.

It is a **family to select from**.

Public data defines a **space of valid identities**.
Private data selects **one identity**.

Breaking the system means collapsing ambiguity without access to global data.

4.3 Why This Survives Quantum Scrutiny

Quantum algorithms accelerate:

- Period finding
- Linear algebra
- Unstructured search

They do **not** bypass the absence of information.

No algorithm—classical or quantum—can extract what is not present in the accessible category.

This is not immunity.
It is **orthogonality**.

END OF PART I

What is now fixed:

- The security regime (controlled non-identifiability)
- The epistemic distinction between transcript and completion
- The legitimacy of mock modular phenomena as a cryptographic substrate

What is not yet fixed:

- The exact transcript format
- The discrete operators
- The concrete primitives

These are built next.

PART II — FORMAL NON-CANONICITY AND THE RESTRICTED OBSERVABLES MODEL

CHAPTER 5 — NON-CANONICITY AS A MATHEMATICAL STRUCTURE

5.1 From Ambiguity to Structure

Ambiguity, in mathematics, is usually a defect: a sign that an object has been insufficiently constrained. The mock modular setting is unusual because its ambiguity is not accidental but **structural**. The space of admissible completions is neither arbitrary nor unbounded; it is organized, finite-dimensional, and governed by well-understood cohomological mechanisms.

This observation is essential. Cryptography cannot be built on vagueness. It requires that any ambiguity be:

1. **Mathematically precise**
2. **Stable under perturbation**
3. **Verifiable without collapse**

Mock modular forms satisfy all three.

5.2 Definition: Non-Canonical Object

We formalize the central notion.

Definition (Non-Canonical Object).
Let L be a class of observable data and G a class of global parameters. An object O is said to be **non-canonical relative to L** if there exists a family $\{O_g\}$ indexed by $g \in G$ such that:

1. For all $g_1, g_2 \in G$,
   O_{g_1} and O_{g_2} are indistinguishable under all observables in L.
2. For $g_1 \neq g_2$,
   $O_{g_1} \neq O_{g_2}$ as global objects.
3. The family $\{O_g\}$ carries a structured action of G (affine, torsorial, or cohomological).

In this sense, non-canonicity is not lack of definition, but **dependence on global choice**.

5.3 Mock Modular Forms as Non-Canonical Objects

Let $f(\tau)$ be a mock modular form of weight k for Γ.

There exists a space $S_{2-k}(\Gamma)$ of cusp forms (the shadows) such that each $g \in S_{2-k}(\Gamma)$ defines a completion:

```
F_g(τ) = f(τ) + R_g(τ)
```

where:

```
R_g(τ) = ∫_{-τ̄}^{i∞} g(z) / (z + τ)^k dz
```

Crucially:

- The holomorphic part f(τ) is **fixed and public**
- The shadow g(τ) is **not determined** by f
- Different g produce globally distinct F_g
- All F_g satisfy the same modular transformation laws

The set of completions {F_g} forms an **affine space** over a subspace of S_{2−k}(Γ), modulo normalization equivalence.

This is non-canonicity in its purest form.

5.4 Torsor Interpretation

More precisely, the completions of f form a torsor under a vector space V:

```
{completions of f} ≅ v₀ + V
```

where:

- There is no distinguished origin
- Differences are meaningful
- Absolute identification is not

This absence of a canonical origin is exactly what cryptography exploits.

CHAPTER 6 — LOCAL OBSERVABILITY VS GLOBAL IDENTITY

6.1 The Local–Global Divide

The defining feature of mock modular phenomena is the separation between:

- **Local observables:**
  Values of f(τ), derivatives, truncated q-coefficients, finite evaluations.
- **Global identity:**
  The shadow g and the completion R_g, which depend on integration over infinite paths.

No amount of local probing reconstructs the global choice.

6.2 Formal Separation of Information Categories

We define two categories of information:

Local information (L):

- Finite truncations of q-series
- Pointwise evaluations
- Finite modular consistency checks
- Any data computable via oracle access to f

Global information (G):

- Shadow g
- Normalization conditions
- Cohomological class selecting the completion

These categories are **not refinements of each other**. G is not "more L".

This categorical separation is what survives discretization.

6.3 The "Difference Below the Noise Floor"

One of the most subtle—and cryptographically powerful—features of mock modular forms is the following phenomenon:

Given two distinct shadows $g_1 \neq g_2$, the corresponding completions F_{g_1} and F_{g_2} can satisfy:

$$|F_{g_1}(\tau) - F_{g_2}(\tau)| < \varepsilon$$

uniformly on any compact subset of $\mathbb{H}$, for arbitrarily small ε, provided the difference is absorbed into the non-holomorphic tail.

Thus:

- Local evaluation cannot distinguish the identities
- Precision does not help
- Identity lives outside the observable category

This is not approximation error.
It is structural underdetermination.

CHAPTER 7 — THE RESTRICTED OBSERVABLES MODEL (ROM)

7.1 Why an Access Model Is Mandatory

Any cryptographic security claim must specify **what the adversary can see and do**.

Claims about "impossibility" without an access model are meaningless.

We therefore define a **Restricted Observables Model (ROM)** that formalizes precisely what information is accessible.

7.2 Definition: Restricted Observables Model

An adversary operating under ROM is granted access to:

1. A finite public transcript T, derived from f
2. Evaluation oracles for f at chosen points
3. Verification oracles that check modular consistency
4. Polynomial-time classical or quantum computation

The adversary is **not** granted access to:

- The shadow g
- The normalization data
- Any oracle that computes R_g directly
- Infinite-precision analytic continuation

ROM is not artificial; it mirrors what real implementations expose.

7.3 What ROM Excludes (and Why That's Legitimate)

ROM excludes global integration data because:

- It is not present in the public transcript
- It cannot be inferred from finite evaluation
- It would require embedding the private key itself into the oracle

ROM is no more restrictive than the standard black-box models used for RSA, ECC, or lattice schemes.

7.4 ROM and Quantum Power

ROM explicitly allows:

- Quantum superposition queries over allowed oracles
- Quantum Fourier sampling on accessible data
- Quantum linear algebra on transcript representations

ROM does **not** allow quantum access to non-existent information.

This distinction is essential: quantum speedups do not create information ex nihilo.

CHAPTER 8 — FORMALIZING THE CRYPTOGRAPHIC PROBLEM

8.1 From Inversion to Identification

Classical cryptography asks:

Given $y = f(x)$, find x.

Mock-theta cryptography asks:

Given a transcript T compatible with many identities $\{I_1, I_2, \ldots\}$, identify which one is correct.

This is a **qualitatively different problem class**.

8.2 The Mock Modular Identification Problem (MMIP)

We now define the central hardness assumption family.

MMIP (Identification Form).
Given:

- A public transcript T derived from a mock modular form f
- Access to all ROM oracles
- Knowledge that T is consistent with exactly M completions $\{F_{g_1}, \ldots, F_{g_M}\}$

Output the index i of the true completion with non-negligible advantage over random guessing.

8.3 MMIP (Distinguishability Form)

MMIP-D.
Given two transcripts T_0 and T_1, one derived from a true completion and one from a randomly chosen alternative completion consistent with f, distinguish which is which.

8.4 MMIP (Forgery Form)

MMIP-F.
Given a valid transcript T, produce a distinct transcript T′ that passes verification but corresponds to no valid private completion.

8.5 Why MMIP Is Not Known to Reduce to Existing Problems

MMIP is:

- Not a group inversion problem
- Not a lattice short-vector problem
- Not an isogeny path problem
- Not a linear system
- Not an abelian hidden subgroup problem

This does **not** imply hardness in an absolute sense.
It implies **novelty of attack surface**.

CHAPTER 9 — SECURITY CLAIMS WE MAKE (AND DO NOT MAKE)

9.1 Claims We Make

We claim that:

1. Under ROM, transcripts do not canonically determine completion identity.
2. MMIP admits no known polynomial-time classical or quantum solution.
3. Known quantum algorithms do not apply directly or give exponential advantage.
4. Non-canonicity survives truncation and discretization.

These claims are testable, falsifiable, and precise.

9.2 Claims We Explicitly Refuse

We do **not** claim:

- Information-theoretic secrecy
- Absolute impossibility
- Unconditional quantum resistance
- Reduction to a well-known NP-hard problem

This refusal is not weakness; it is discipline.

9.3 What "Security" Means in This Architecture

Security means:

An adversary cannot collapse structured ambiguity without access to global completion data, even with quantum resources.

Nothing more.
Nothing less.

END OF PART II

What is now fixed:

- The formal meaning of non-canonicity
- The local/global information divide
- The Restricted Observables Model
- The MMIP hardness family
- The precise scope of security claims

What comes next:

- How to **encode** this structure into finite transcripts
- How to build **verifiers** that accept families, not identities
- How ambiguity **survives discretization**

PART III — CRYPTOGRAPHIC FORMALIZATION: GAMES, INTERFACES, AND THREAT MODELS

CHAPTER 10 — FROM MATHEMATICAL PHENOMENON TO CRYPTOGRAPHIC OBJECT

10.1 Why Formalization Is the Point of No Return

A mathematical phenomenon becomes cryptography only when it is forced to survive the discipline of formal attack models. This requires translating non-canonicity—originally expressed in analytic and cohomological language—into **interfaces**, **oracles**, and **games** that admit adversarial interaction.

The objective of this part is not to prove security in the classical reductionist sense. It is to define a **clean, falsifiable framework** in which claims about security can be tested, attacked, and improved.

We therefore proceed by fixing:

1. What constitutes a public key
2. What constitutes a private key
3. What operations are allowed
4. What it means to "win" as an adversary

Only after these are fixed does it make sense to speak about security.

10.2 Cryptographic Objects and Interfaces

We define a family of schemes collectively denoted **Θ-Crypt**, parameterized by a tuple of public parameters Π.

Public Parameters Π include:

- A prime p defining the base field
- A truncation bound N
- A weight k
- A congruence subgroup Γ (or an abstract action set replacing it)
- A transcript format specification
- A verifier specification

Π is fixed globally and assumed known to all parties, including adversaries.

CHAPTER 11 — KEYS, TRANSCRIPTS, AND COMPLETIONS

11.1 Public Transcript

The **public key** in Θ-Crypt is not a single algebraic object. It is a **transcript**, denoted T, consisting of finite data derived from the holomorphic component f and its consistency constraints.

A transcript T may include:

- Truncated coefficients $a_0,\ldots,a_{N-1}$ modulo p
- A finite set of evaluation pairs $(\tau_i, f(\tau_i))$
- Seeds for deterministic generation of evaluation points
- Commitments to consistency operators
- Auxiliary metadata enabling verification

Crucially, **T is verifiable without knowing the private key**.

11.2 Private Completion Data

The **private key** consists of completion data Y:

- A shadow selector g (represented discretely)
- Normalization / rigidification data η
- Optional auxiliary data enabling efficient completion

Y is never exposed through any public oracle.

11.3 Completion Operator

We define an abstract operator:

```
Comp(T, Y) → F
```

such that:

- F is a completed object consistent with T
- For fixed T, different Y produce distinct F
- Verification of F against T is efficient
- Decomposition of F into (T, Y) is non-unique under ROM

This operator is **conceptual**, not necessarily implemented explicitly in all schemes. What matters is that its existence is well-defined.

CHAPTER 12 — VERIFICATION AS A FIRST-CLASS PRIMITIVE

12.1 Verifiers Accept Families, Not Identities

Classical cryptographic verifiers check whether a given object equals a unique expected value.

Θ-Crypt verifiers check something subtler:

Whether a candidate object lies within the **admissible completion family** determined by the transcript.

This distinction is essential. Verification must **not collapse ambiguity**.

12.2 Verification Interface

We define a deterministic polynomial-time verifier:

```
Verify(Π, T, X) → {accept, reject}
```

where X is a candidate object (e.g., ciphertext component, signature witness, or completed evaluation).

Correctness requires:

- For the true completion F = Comp(T, Y), Verify accepts
- For malformed or inconsistent X, Verify rejects
- For alternative valid completions F′ consistent with T, Verify accepts

This last condition is the architectural heart of the system.

12.3 Consistency Operators

Verification is implemented via **consistency operators**, which may include:

- Modular transport checks
- Convolution invariants
- Cusp-adjacent stress tests
- Structured linear or non-linear relations among transcript components

Each operator is designed to be:

- Efficient to compute
- Deterministic
- Blind to the specific completion identity

The verifier is a composition of these operators.

CHAPTER 13 — ADVERSARY MODELS

13.1 Computational Power

Adversaries are allowed:

- Polynomial-time classical computation
- Polynomial-time quantum computation
- Superposition access to allowed oracles

We do not restrict adversaries by computational class beyond polynomial bounds.

13.2 Oracle Access

Under the Restricted Observables Model (ROM), adversaries may access:

- Transcript T
- Evaluation oracles for f
- Verification oracles Verify(Π, T, ·)
- Encryption and decryption oracles, where applicable

Adversaries may **not** access:

- Completion operator Comp
- Shadow data g
- Any oracle that directly evaluates R_g

13.3 Adaptive Interaction

Adversaries may:

- Choose queries adaptively
- Correlate queries across sessions
- Perform chosen-ciphertext or chosen-message attacks, depending on the scheme

The model is intentionally strong.

CHAPTER 14 — SECURITY GAMES

14.1 Identification Game (MMIP-ID)

Setup:

- Challenger samples (T, Y)
- T is given to adversary
- Y is kept secret

Challenge:

- Adversary outputs an index i or candidate Y′

Win Condition:

- Adversary identifies the correct completion with advantage ε over random guessing

Security requires ε negligible in Π.

14.2 Distinguishability Game (MMIP-D)

Setup:

- Challenger samples T
- Challenger samples Y_0 (true) and Y_1 (alternative)
- Chooses $b \in \{0,1\}$ uniformly
- Computes X = Comp(T, Y_b)

Challenge:

- X is given to adversary

Win Condition:

- Adversary outputs b with advantage ε

This game captures indistinguishability of completions.

14.3 Forgery Game (MMIP-F)

Setup:

- Challenger gives adversary T

Challenge:

- Adversary outputs $T' \neq T$

Win Condition:

- Verify(Π, T′, ·) accepts, but no valid Y′ exists such that T′ = Transcript(Y′)

This models transcript malleability and integrity.

14.4 Scheme-Specific Games

Higher-level schemes (KEM, signatures) are defined by composing MMIP with standard cryptographic games:

- IND-CCA for KEM
- EUF-CMA for signatures

The crucial difference is that **the secret key is not uniquely determined by the public key**, altering the structure of reductions.

CHAPTER 15 — WHY STANDARD REDUCTIONS DO NOT APPLY CLEANLY

15.1 Absence of a Unique Preimage

Classical reductions assume a function f such that:

```
f(x) = y  ⇒  x is unique
```

Θ-Crypt violates this assumption by construction.

There is no canonical preimage to recover.

15.2 Why Shor-Type Attacks Do Not Map

Shor's algorithm requires:

- A group
- An efficiently computable function with hidden periodicity

Θ-Crypt provides neither:

- The ambiguity class is not a group
- Identity selection is not periodic
- Verification does not expose algebraic structure exploitable by QFT

This is not a proof of resistance; it is a structural mismatch.

15.3 Grover-Style Search Is Parameterizable Away

An adversary may attempt:

- Quadratic-speedup search over the space of completions

This is mitigated by:

- Exponentially large ambiguity classes
- High cost per completion test
- Tunable parameters increasing entropy

Grover provides at most a square-root improvement, which is compensated parametrically.

15.4 Non-Abelian HSP Considerations

One may attempt to embed MMIP into a non-abelian hidden subgroup problem.

At present:

- No efficient quantum algorithms exist for general non-abelian HSP
- The mapping from MMIP to HSP is non-obvious and lossy
- Verification operators are not homomorphisms

This remains an open line of cryptanalysis, not a known vulnerability.

CHAPTER 16 — WHAT "SECURITY" MEANS HERE

16.1 Security Is Not Inversion Resistance

Security does not mean:

- "You cannot compute the secret"
- "The secret is hidden in the public key"

Security means:

You cannot identify which valid global identity is correct using accessible information.

16.2 Falsifiability and Scientific Posture

The claims in Θ-Crypt are intentionally framed to be:

- Testable
- Attackable
- Refutable

Any successful attack must:

- Exploit ROM-accessible information
- Collapse ambiguity
- Identify completion identity with non-negligible advantage

This defines a clear research program, not a marketing claim.

END OF PART III

What is now fixed:

- Cryptographic interfaces
- Verifier semantics
- Adversary models
- Security games
- Precise meaning of "breaking" the system

What comes next:

- How to **discretize** mock modular structure without losing non-canonicity
- How to design **finite transcripts and operators**
- How ambiguity **survives truncation**

PART IV — DISCRETIZATION, FINITE TRANSCRIPTS, AND VERIFIABLE AMBIGUITY

CHAPTER 17 — THE DISCRETIZATION PROBLEM STATED HONESTLY

17.1 Why Most “Deep Math Cryptography” Fails Here

Many cryptographic proposals grounded in sophisticated mathematics fail at a predictable point: discretization. Continuous objects, infinite series, analytic integrals, or geometric structures are translated into finite arithmetic in a way that accidentally restores canonicity, linearizes structure, or exposes a hidden group.

If non-canonicity vanishes under truncation, the entire architecture collapses.

Thus, the central engineering question is not:

“Can we compute mock theta functions approximately?”

but rather:

“Can we preserve epistemic underdetermination under finite, discrete representation?”

17.2 The Correct Abstraction Boundary

The key insight—already implicit in Part II—is that **we do not discretize the object; we discretize the observables**.

The analytic object lives at the **semantic layer**.
The implementation lives at the **transcript layer**.

What must survive discretization is not analytic fidelity, but **category separation**:

- Local observables remain local
- Global identity remains global
- Verification remains possible
- Identification remains impossible

CHAPTER 18 — THE SEMANTIC LAYER VS THE TRANSCRIPT LAYER

18.1 Two Layers, One Contract

We explicitly separate the system into two layers:

Semantic layer (idealized):

- Mock modular forms on $\mathbb{H}$
- Infinite q-series
- Non-holomorphic completions

- Cohomological ambiguity

Transcript layer (implemented):

- Finite data over $\mathbb{F}_p$ or $\mathbb{Z}/p\mathbb{Z}$
- Truncated series or sampled evaluations
- Deterministic consistency operators
- Verifiers that accept families

The only contract between the layers is:

Any valid semantic completion must induce an accepted transcript;
transcripts do not uniquely determine semantic completion.

18.2 What the Transcript Must Guarantee

A transcript T must satisfy four properties:

1. **Finiteness:** bounded size, polynomial in security parameter
2. **Verifiability:** correctness checkable without private data
3. **Stability:** small perturbations do not collapse ambiguity
4. **Non-identifiability:** multiple completions remain consistent

Failure of any one property invalidates the scheme.

CHAPTER 19 — FINITE REPRESENTATIONS OF HOLOMORPHIC DATA

19.1 Truncated Coefficient Representation

The most direct transcript representation is via truncated q-series:

$$\hat{f}(q) = \Sigma_{n=0}^{N-1} a_n q^n \quad \text{with } a_n \in \mathbb{F}_p$$

This representation is:

- Compact
- Deterministic
- Easy to generate and manipulate

However, **coefficients alone are insufficient**. They are vulnerable to:

- Linear reconstruction attempts
- Interpolation attacks
- Algebraic lifting if used naively

Thus, coefficient data must be paired with **consistency structure**.

19.2 Evaluation-Based Representation

An alternative (or complementary) approach uses sampled evaluations:

```
E = {(τ_i, f(τ_i)) | i = 1,…,M}
```

where τ_i are deterministically generated points (often near cusps, roots of unity, or along modular orbits).

Evaluation-based transcripts:

- Break linear structure
- Resist coefficient interpolation
- Align naturally with verification operators

They are more expensive but often more robust.

19.3 Hybrid Transcripts

In practice, **hybrid transcripts** dominate:

```
T = (coefficients, evaluations, operator seeds, metadata)
```

This allows:

- Fast arithmetic on coefficients
- Nonlinear constraints via evaluations
- Redundant structure that increases ambiguity robustness

Hybrid transcripts are the default in Θ-Crypt.

CHAPTER 20 — CONSISTENCY OPERATORS AS VERIFIABILITY ENGINES

20.1 Why Equality Is the Wrong Test

Classical cryptographic verification checks equality:

```
f(x) ?= y
```

Θ-Crypt verification checks **consistency under transformations**.

The verifier never asks:

“Is this the correct identity?”

It asks:

“Is this consistent with the transcript constraints?”

20.2 Classes of Consistency Operators

We define several operator families. Implementations may use a subset.

(a) Modular Transport Operators

Given a transformation $\gamma \in \Gamma$, define:

```
Op_γ(T):   T → T_γ
```

The verifier checks that:

```
T and T_γ satisfy prescribed relations
```

without resolving identity.

(b) Cusp-Stress Operators

Evaluations near cusps or roots of unity amplify global sensitivity.

The verifier checks:

- Bounded growth
- Expected oscillatory behavior
- Absence of algebraic closure

These tests are **agnostic to the shadow**.

(c) Convolution and Folding Operators

Discrete analogues of Eichler-type behavior are enforced via:

- Structured convolutions
- Folding under involutions
- Symmetry-breaking invariants

These preserve ambiguity while rejecting malformed data.

20.3 Determinism and Auditability

All operators are:

- Deterministic
- Parameterized by public seeds
- Reproducible bit-for-bit

This is critical for:

- Auditability
- Consensus protocols
- Hardware implementation

CHAPTER 21 — ERROR TOLERANCE AND "DIFFERENCE BELOW THE NOISE FLOOR"

21.1 Numerical Error Is Not the Threat

A common misconception is that numerical error weakens security.

In Θ-Crypt, **controlled tolerance is a feature**, not a bug.

Two completions may differ globally while remaining:

$|\Delta| < \varepsilon$

for all transcript observables.

This preserves ambiguity even under high precision.

21.2 Defining Operational Indistinguishability

We define an equivalence relation:

$F_1 \approx_\varepsilon F_2 \quad \Leftrightarrow \quad$ all verification operators accept both

Security requires that:

- The equivalence class size is large
- Membership testing does not shrink the class

- Precision increases do not collapse equivalence

21.3 Parameterizing ε

The tolerance ε is:

- Explicit
- Public
- Parameterized by security level

Increasing security increases:

- Transcript redundancy
- Operator diversity
- Ambiguity width

ε is not hidden; it is engineered.

CHAPTER 22 — SURVIVAL OF NON-CANONICITY UNDER TRUNCATION

22.1 The Central Engineering Theorem (Informal)

Truncation preserves non-canonicity provided verification enforces global consistency without identity selection.

This is not a mathematical theorem; it is a design theorem.

22.2 Why Truncation Does Not Collapse Ambiguity

Ambiguity survives because:

- Shadow data influences only non-local structure
- Truncation affects all candidates symmetrically
- Verification operators are blind to shadow choice

Thus, no completion is privileged.

22.3 What Would Break the System

The system fails if:

- The verifier enforces a unique normal form
- Operators accidentally impose linear constraints
- Transcripts leak global data implicitly
- Discretization introduces hidden group structure

Every implementation choice is evaluated against these failure modes.

CHAPTER 23 — TRANSCRIPT SPECIFICATION v0.1

23.1 Public Transcript Format (Abstract)

A Θ-Crypt transcript T contains:

```
T = {
  params: (p, N, k, Γ_id),
  coeffs: [a_0,…,a_{N−1}],
  evals: [(τ_i, v_i)]_{i=1}^M,
  ops: [op_seed_1,…,op_seed_L],
  tolerance: ε,
  metadata
}
```

All fields are public.

23.2 Verifier Pseudocode (Conceptual)

```
function Verify(Π, T, X):
    for each operator Op in T.ops:
        if Op(T, X) fails within tolerance ε:
            reject
    accept
```

Note:

- X need not be unique
- X need not encode identity
- Acceptance implies consistency, not correctness

23.3 Transcript Generation Contract

A transcript generator must ensure:

- At least K valid completions exist
- No operator collapses ambiguity
- Transcript entropy scales with parameters

These are measurable engineering criteria.

CHAPTER 24 — IMPLEMENTATION CONSEQUENCES

24.1 Software Implications

- No single “decrypt-and-compare” path
- Verification dominates cost
- Constant-time discipline applies to operators, not identity

---

24.2 Hardware Implications

- Operators are embarrassingly parallel
- Convolution and folding map well to FPGA/ASIC
- No large integer factorization or lattice reduction units required

Θ-Crypt hardware looks unlike RSA, ECC, or PQC accelerators.

24.3 Testing Philosophy

Testing does not ask:

“Does it decrypt correctly?”

It asks:

“Does ambiguity survive under adversarial stress?”

This changes the entire validation mindset.

END OF PART IV

What is now fixed:

- The semantic/transcript boundary

- Finite transcript representations
- Verifier and operator philosophy
- Survival of non-canonicity under truncation

What comes next:

- Concrete cryptographic constructions
- Key exchange and signatures
- End-to-end correctness and security reasoning

PART V — CONCRETE CONSTRUCTIONS: Θ-KEM AND Θ-SIGN

CHAPTER 25 — DESIGN PHILOSOPHY FOR CONCRETE PRIMITIVES

25.1 Why Start with KEM and Signatures

Public-key cryptography, in practice, reduces to two primitives:

- **Key establishment** (for confidentiality)
- **Authenticity** (for integrity and non-repudiation)

If a new cryptographic ontology cannot express these primitives cleanly, it is not cryptography—it is mathematics without leverage.

Θ-Crypt therefore commits early to:

- a **KEM-first** design (as modern PQC does),
- and a **proof-of-knowledge-style signature**, aligned with non-canonicity.

25.2 Non-Goals (Explicit)

Θ-Crypt primitives are **not** designed to:

- Minimize arithmetic cost at all costs
- Beat lattice schemes in raw throughput
- Provide drop-in replacement without protocol adaptation

They are designed to:

- Preserve epistemic asymmetry
- Admit formal security games
- Produce measurable, attackable artifacts

CHAPTER 26 — Θ-KEM: KEY ENCAPSULATION VIA AMBIGUOUS COMPLETION

26.1 High-Level Intuition

Θ-KEM replaces the classical Diffie–Hellman intuition:

"Both parties derive the same secret from public algebraic structure"

with a different one:

One party creates ambiguity; the other collapses it.

The ciphertext is deliberately ambiguous.
Only possession of the private completion data collapses it to a unique shared secret.

26.2 Algorithms (Abstract Interface)

Θ-KEM consists of four algorithms:

```
( pk, sk ) ← KeyGen(Π)
( ct, K )  ← Encaps(pk)
K          ← Decaps(sk, ct)
accept/reject via Verify
```

Correctness requires:

- Decapsulation succeeds with overwhelming probability
- Invalid ciphertexts are rejected or map to ⊥

CHAPTER 27 — Θ-KEM.KEYGEN

27.1 Inputs

- Public parameters Π
- Security parameter λ

27.2 Procedure

1. **Sample semantic object**
    - Choose a mock-modular holomorphic component f
    - Choose completion data Y = (g, η)
2. **Generate transcript**
    - Compute transcript T = Transcript(f, Π)
    - Ensure T admits ≥ 2^λ valid completions
3. **Fix verification operators**
    - Derive deterministic operator seeds
    - Bind them into T
4. **Output keys**

```
5. pk = T
6. sk = Y
```

27.3 Correctness Condition

There exists a function Comp such that:

```
Comp(pk, sk) ∈ ValidCompletions(pk)
```

and Verify accepts it.

CHAPTER 28 — Θ-KEM.ENCAPS

28.1 Core Idea

Encapsulation injects a **secret selector s** into the ambiguity class defined by pk, producing a ciphertext that:

- Is consistent with pk
- Is ambiguous without sk
- Becomes unique with sk

28.2 Procedure

1. **Sample ephemeral selector**

```
2. s ← {0,1}^λ
```

3. **Derive ephemeral completion**
    - Map s → Y_s (pseudo-shadow or completion modifier)
    - This mapping is public but non-invertible
4. **Generate ambiguous ciphertext**

```
5. ct = Embed(pk, Y_s)
```

    where Embed produces an object accepted by Verify(pk, ·)

6. **Derive shared key**

```
7. K = KDF(s || pk || ct)
```

28.3 Ciphertext Properties

- ct ∈ ValidCompletions(pk)
- ct does **not** reveal s
- Multiple s′ map to ciphertexts indistinguishable under ROM

This is intentional.

CHAPTER 29 — Θ-KEM.DECAPS

29.1 Procedure

1. **Verify ciphertext**

```
2. if Verify(Π, pk, ct) == reject:
3.     return ⊥
```

4. **Collapse ambiguity**
   - Use private completion data sk = Y
   - Compute:
   - `s = Extract(pk, ct, sk)`
5. **Derive shared key**

```
6. K = KDF(s || pk || ct)
```

29.2 Failure Modes

- If ct was malformed → Verify fails
- If ct is valid but adversarially crafted → Extract fails or yields random s

Both are acceptable and secure.

CHAPTER 30 — Θ-KEM SECURITY

30.1 Security Notion

Θ-KEM targets **IND-CCA security**, interpreted through MMIP.

30.2 IND-CCA Game (Sketch)

- Challenger generates (pk, sk)

- Adversary receives pk
- Adversary queries:
  - Encapsulation oracle
  - Decapsulation oracle (except on challenge ct*)
- Challenger issues challenge (ct*, K*)
- Adversary must distinguish K* from random

30.3 Why the Game Holds (Intuition)

- ct* admits many valid interpretations
- Without sk, adversary cannot identify s
- Quantum queries do not reveal global data
- Decapsulation oracle does not leak sk because ambiguity remains

Formal proof reduces advantage to MMIP-D.

CHAPTER 31 — Θ-SIGN: SIGNATURES VIA NON-CANONICAL WITNESSES

31.1 Why Signatures Are Different Here

In classical schemes:

- Signature proves possession of a unique secret

In Θ-Crypt:

- Signature proves possession of **some valid completion data**
- Identity is not revealed
- Only **consistency and legitimacy** are proven

This aligns naturally with non-canonicity.

CHAPTER 32 — Θ-SIGN.KEYGEN

Identical to Θ-KEM.KeyGen:

```
(pk, sk) ← KeyGen(Π)
```

CHAPTER 33 — Θ-SIGN.SIGN

33.1 Inputs

- Message m
- Private key sk = Y
- Public transcript pk = T

33.2 Procedure

1. **Compute challenge**
2. `c = H(m || T)`
3. **Derive constrained completion**
    - Use sk to compute a witness W such that:
    - `Verify(Π, T, W) == accept`

      and W is bound to c

4. **Output signature**
5. `σ = (W, aux)`

aux may include randomness or proof components.

33.3 Signature Semantics

The signature asserts:

“I know some completion data Y such that W is consistent with T and bound to message m.”

It does **not** reveal Y.

CHAPTER 34 — Θ-SIGN.VERIFY

34.1 Procedure

1. Recompute:
2. `c = H(m || T)`
3. Check:
4. `Verify(Π, T, W) == accept`
5. Check binding of W to c

If all checks pass, accept.

CHAPTER 35 — Θ-SIGN SECURITY

35.1 Security Notion

Θ-Sign targets **EUF-CMA** (existential unforgeability under chosen-message attack).

35.2 Why Forgery Is Hard

To forge σ on a new message m′, adversary must:

- Produce a W′ accepted by Verify
- Bind W′ to c′ = H(m′ || T)
- Without knowing any valid completion data Y

This requires solving **MMIP-F**, not inversion.

35.3 Aggregation and Extensions

Because signatures live in a family-accepting verification regime:

- Aggregation is possible
- Batch verification is natural
- Multi-signature variants are plausible

These are future extensions, not required for core security.

CHAPTER 36 — CORRECTNESS, FAILURE, AND IMPLEMENTATION NOTES

36.1 Correctness Guarantees

- Honest Encaps/Decaps succeed with overwhelming probability
- Honest Sign/Verify succeed deterministically

36.2 Failure Is Not Catastrophic

Unlike inversion-based systems:

- Failure does not leak secret structure
- Failure does not narrow ambiguity

- Failure degrades to randomness, not compromise

36.3 Implementation Discipline

- All Verify logic must be constant-time
- No operator may encode identity
- Transcript entropy must be audited

END OF PART V

What is now fixed:

- End-to-end cryptographic primitives
- KEM and signature semantics
- Security games linked to MMIP
- Implementation-ready algorithmic skeletons

What comes next:

- Parameter selection and performance
- Attack harnesses and benchmarking
- Software and hardware realizations

PART VI — SECURITY ENGINEERING, PARAMETERS, AND ATTACK HARNESSES

CHAPTER 37 — WHY ENGINEERING DISCIPLINE DEFINES CREDIBILITY

37.1 From Correctness to Assurance

A cryptographic construction is not validated by elegance or novelty. It is validated by its behavior under stress: malformed inputs, adaptive adversaries, side channels, parameter drift, and implementation mistakes.

Θ-Crypt introduces a new security regime—controlled non-identifiability—which **cannot be validated by traditional "does it decrypt?" tests**. It requires a different assurance discipline:

- Measure **ambiguity width**
- Measure **adversarial advantage curves**
- Measure **collapse thresholds** under parameter changes
- Measure **leakage**, not inversion

This part formalizes that discipline.

37.2 What Engineering Must Guarantee (and What It Need Not)

Engineering must guarantee:

1. Deterministic verification
2. Stable acceptance under tolerance ε
3. Large equivalence classes of valid completions
4. No hidden identity leaks

Engineering need **not** guarantee:

- Unique decoding
- Perfect reconstruction
- Zero failure probability

Failure is acceptable if it does not reduce ambiguity.

CHAPTER 38 — PARAMETERIZATION FRAMEWORK

38.1 Parameter Categories

Θ-Crypt parameters fall into four orthogonal categories:

1. **Arithmetic parameters**
   - Prime p
   - Field representation
2. **Transcript parameters**
   - Truncation length N
   - Number of evaluations M
3. **Operator parameters**
   - Operator count L
   - Operator diversity
4. **Tolerance parameters**
   - Acceptance ε
   - Numerical precision bounds

Security tuning consists of balancing these categories.

38.2 Security Levels

We define three **engineering security levels**, not "bits of hardness" in the classical sense, but **bits of ambiguity**.

Level Θ-1 (Exploratory / Lightweight)

```
p ≈ 2^128
N ≈ 512
M ≈ 32
L ≈ 8
Ambiguity width ≈ 2^80
```

Use cases:

- Research prototypes
- Embedded authentication
- Early integration testing

Level Θ-2 (Standard)

```
p ≈ 2^256
N ≈ 1024
M ≈ 64
L ≈ 16
Ambiguity width ≈ 2^128
```

Use cases:

- General-purpose KEM
- TLS-style handshakes
- Enterprise authentication

Level Θ-3 (High Assurance)

```
p ≈ 2^512
N ≈ 2048
M ≈ 128
L ≈ 32
Ambiguity width ≈ 2^256
```

Use cases:

- Long-term confidentiality
- Strategic infrastructure
- High-value key management

38.3 Why “Bits of Security” Are Reinterpreted

In Θ-Crypt, “security bits” do not measure:

cost of inversion

They measure:

minimum entropy of the ambiguity class under ROM

This is measured empirically and analytically via attack harnesses.

CHAPTER 39 — PARAMETER SELECTION RULES

39.1 Non-Negotiable Constraints

Any valid parameter set must satisfy:

1. The verifier accepts ≥ 2^λ completions
2. No operator collapses equivalence classes
3. Increasing precision does not reduce ambiguity
4. Transcript entropy grows at least linearly with N

Failure of any constraint invalidates the set.

39.2 Anti-Patterns (What Not to Do)

Do **not**:

- Choose operators that implicitly normalize identity
- Overconstrain evaluation points
- Allow deterministic ordering of completions
- Leak shadow-dependent statistics

Most failures of exotic cryptosystems occur here.

CHAPTER 40 — ATTACK HARNESSES: A NEW TESTING PHILOSOPHY

40.1 Why Classical Cryptanalysis Tooling Is Insufficient

Traditional cryptanalysis attempts to:

- Recover the secret
- Break correctness
- Find a collision

Θ-Crypt must be attacked differently:

Try to collapse ambiguity using only ROM-accessible information.

This requires dedicated harnesses.

40.2 The Θ-Attack Harness Architecture

A Θ-Crypt attack harness consists of:

1. **Transcript generator**
2. **Adversarial oracle interface**
3. **Strategy modules**
4. **Advantage estimator**
5. **Logging and reproducibility engine**

The harness outputs:

- Advantage vs time curves
- Ambiguity width decay
- Operator sensitivity metrics

40.3 Baseline Attacks (Required)

Every implementation must ship with results against:

1. **Brute-force completion enumeration**
2. **Heuristic identification**
3. **Linearization attempts**
4. **Precision escalation attacks**
5. **Transcript perturbation attacks**

Failure to test against these invalidates claims.

CHAPTER 41 — CLASSICAL ATTACK CLASSES

41.1 Enumeration and Pruning

Attack:

- Enumerate candidate completions
- Prune using Verify

Defense:

- Exponential candidate space
- High cost per verification
- Parameterizable ambiguity width

Metrics:

- Surviving candidates vs time

41.2 Linear Regression and Interpolation

Attack:

- Attempt to fit coefficients or evaluations
- Recover shadow-like structure

Defense:

- Non-linear operators
- Mixed representation
- Operator diversity

Metrics:

- Correlation leakage
- Residual predictability

41.3 Statistical Fingerprinting

Attack:

- Cluster transcripts
- Identify bias correlated with completion data

Defense:

- Deterministic transcript generation
- Uniform operator application

- No identity-conditioned randomness

Metrics:

- Mutual information estimates

CHAPTER 42 — QUANTUM ATTACK CLASSES

42.1 Grover-Style Search

Attack:

- Quadratic speedup over completion candidates

Defense:

- Ambiguity width >> quadratic advantage
- Costly verification oracle
- Tunable entropy

Metrics:

- Expected time-to-collapse vs parameters

42.2 Quantum Linear Algebra (HHL-like)

Attack:

- Linearize constraints
- Solve for hidden structure

Defense:

- Non-linear operators
- No sparse linear system
- Non-abelian structure

Metrics:

- Conditioning numbers
- Solver convergence failure

42.3 Quantum Machine Learning

Attack:

- Train classifiers to predict completion identity

Defense:

- Label symmetry
- No supervised signal
- High-dimensional equivalence classes

Metrics:

- Classification accuracy vs chance

This class of attacks is **mandatory** to test.

CHAPTER 43 — SIDE-CHANNEL AND FAULT ANALYSIS

43.1 Timing and Power Leakage

Key observation:

- Θ-Crypt never branches on identity

Therefore:

- Timing leaks are limited to operator execution
- Identity-dependent leakage is structurally minimized

Mitigations:

- Constant-time operators
- Blinded evaluation order
- Fixed verification schedules

43.2 Fault Injection

Attack:

- Inject faults during verification or extraction

Defense:

- Redundant operators

- Transcript self-consistency checks
- Fail-closed behavior

Faults produce rejection, not information.

CHAPTER 44 — BENCHMARKING AND METRICS

44.1 What Must Be Measured

Every Θ-Crypt implementation must report:

1. Transcript size
2. Verification latency
3. Encapsulation/decapsulation time
4. Ambiguity width estimates
5. Attack advantage curves

44.2 Comparison with PQC Schemes

Θ-Crypt benchmarking is **not apples-to-apples** with Kyber or Dilithium.

Comparison axes:

- Security diversification
- Attack surface orthogonality
- Hardware profile
- Long-term confidence

Raw speed is secondary.

CHAPTER 45 — AUDIT ARTIFACTS AND INVESTOR READINESS

45.1 Mandatory Artifacts

A production-ready Θ-Crypt system must provide:

- Formal parameter spec
- Reference implementation
- Attack harness results
- Reproducible benchmarks
- Failure analysis logs

This is non-negotiable.

45.2 What Investors Should Ask For

Serious investors should demand:

- Evidence of ambiguity survival
- Independent cryptanalysis attempts
- Hardware feasibility studies
- Clear separation between research and product

Θ-Crypt is not a black box; it is a research platform.

END OF PART VI

What is now fixed:

- Parameter regimes
- Security engineering discipline
- Attack harness philosophy
- Benchmarking standards
- Audit artifacts

What remains:

- Software and hardware realizations
- Deployment architectures
- Governance, licensing, and roadmap

PART VII — IMPLEMENTATION ARCHITECTURES, HARDWARE PATHS, AND DEPLOYMENT MODELS

CHAPTER 46 — SOFTWARE ARCHITECTURE: FROM SPEC TO CODE

46.1 Architectural Principles

Θ-Crypt software is designed around three non-negotiable principles:

1. **Verifier-Centricity**
   Verification dominates computation. All paths are organized around deterministic operator execution.
2. **Transcript Immutability**
   Public transcripts are immutable once generated. All derived objects reference them by hash.
3. **Ambiguity Preservation**
   No software layer is allowed to infer, cache, or normalize identity.

These principles override conventional optimization instincts.

46.2 Reference Software Stack

A minimal Θ-Crypt software stack consists of:

- **Core Arithmetic Layer**
  - Finite-field arithmetic
  - Polynomial and convolution engines
  - Deterministic PRNG for operator seeds
- **Transcript Layer**
  - Transcript generation
  - Serialization/deserialization
  - Hash binding and integrity checks
- **Operator Engine**
  - Modular transport operators
  - Convolution/folding operators
  - Stress and consistency operators
- **Primitive Layer**
  - Θ-KEM
  - Θ-Sign
- **Harness Layer**
  - Attack harness
  - Benchmark runner
  - Ambiguity estimator

46.3 Language Choices

Recommended implementation languages:

- **Rust / C** for core arithmetic and verification
- **Python** for research harnesses and parameter exploration
- **Verilog / VHDL** for hardware kernels

The reference implementation must prioritize **auditability over concision**.

CHAPTER 47 — HARDWARE ARCHITECTURE: FPGA TO ASIC

47.1 Why Θ-Crypt Hardware Is Different

Θ-Crypt does not require:

- Large integer factorization
- Lattice basis reduction
- Elliptic curve scalar multiplication

Instead, it requires:

- Massive parallelism
- Deterministic operator pipelines
- High-throughput convolution

This produces a fundamentally different hardware profile.

47.2 FPGA Reference Architecture

An FPGA Θ-Crypt accelerator consists of:

- **Field Arithmetic Units**
  - Modular add/multiply
- **Convolution Engines**
  - NTT-like or custom folding logic
- **Operator Scheduler**
  - Fixed execution order
  - No data-dependent branching
- **Verification Controller**
  - Aggregates operator results
  - Enforces tolerance ε

FPGA prototypes are sufficient to validate:

- Throughput
- Power consumption
- Side-channel behavior

47.3 ASIC Path

ASIC implementation becomes attractive when:

- Verification dominates cost
- Operators stabilize
- Parameters are standardized

Projected ASIC advantages:

- 10–50× energy efficiency
- Predictable timing
- Reduced attack surface

Θ-Crypt ASICs resemble **signal-processing accelerators**, not crypto cores.

CHAPTER 48 — DEPLOYMENT MODELS

48.1 Library Integration

Θ-Crypt can be deployed as:

- A standalone cryptographic library
- A KEM/signature backend for existing protocols
- A hybrid scheme combined with lattice-based PQC

Library deployment is the lowest barrier to adoption.

48.2 Protocol Integration

Θ-KEM integrates naturally into:

- TLS-style handshakes
- VPN key negotiation
- Secure messaging systems

The handshake pattern is standard:

1. Exchange public transcripts
2. Encapsulate shared secret
3. Derive symmetric keys

No protocol-level exoticism is required.

48.3 Hybrid Deployment (Recommended)

For conservative environments, Θ-Crypt is best deployed as part of a **hybrid scheme**:

```
K = KDF(K_lattice || K_theta)
```

This provides:

- Defense in depth
- Smooth migration
- Risk diversification

Hybridization is not a concession; it is strategic.

CHAPTER 49 — FAILURE, RECOVERY, AND OPERATIONAL SAFETY

49.1 Failure Is Benign by Design

In Θ-Crypt:

- Verification failure ⇒ rejection
- Extraction failure ⇒ random output or ⊥
- No failure reveals identity

This sharply contrasts with schemes where failure leaks structure.

49.2 Key Rotation and Revocation

Key rotation consists of:

- Regenerating transcript + completion data
- Publishing new transcript hash
- Deprecating old transcripts

No incremental leakage accumulates across rotations.

49.3 Long-Term Storage

Θ-Crypt public keys can be stored indefinitely without degradation of security assumptions, because:

- No future algorithm retroactively reveals identity
- Ambiguity is structural, not computational

This is particularly attractive for archival encryption.

CHAPTER 50 — GOVERNANCE AND INSTITUTIONAL STRUCTURE

50.1 Separation of Concerns

Θ-Crypt requires strict institutional separation:

- **Foundational Authority**
  - Owns ontology and assumptions
  - Maintains mathematical definitions
- **Implementation Entities**
  - Build software and hardware
  - Compete on performance
- **Deployment Entities**
  - Integrate into products
  - Assume operational risk

This separation preserves credibility.

50.2 Open Research, Controlled Implementation

Recommended model:

- Publish:
  - Ontology
  - Security games
  - Transcript formats
  - Reference implementation
- License:
  - Optimized implementations
  - Hardware IP
  - Certified operator sets

This mirrors successful cryptographic ecosystems.

CHAPTER 51 — STANDARDIZATION AND EXTERNAL SCRUTINY

51.1 Path to Standardization

Θ-Crypt is not immediately suited for NIST-style bake-offs.

Instead, the path is:

1. Independent academic scrutiny
2. Open cryptanalysis challenges
3. Hybrid deployment experiments
4. Gradual formalization

Standardization follows evidence, not ambition.

51.2 What Scrutiny Should Focus On

External reviewers should focus on:

- Ambiguity collapse attempts
- Operator-induced leaks
- Discretization artifacts
- Side-channel behavior

Not on:

- Inversion hardness
- Classical reductions
- Algebraic elegance

CHAPTER 52 — ECONOMIC AND STRATEGIC POSITIONING

52.1 What Θ-Crypt Is Selling

Θ-Crypt does **not** sell speed.

It sells:

- Ontological diversification
- Structural robustness
- Long-term uncertainty resistance

This is a strategic asset, not a commodity.

52.2 Where Value Accrues

Value accrues in:

- Verified implementations
- Attack-resistant operator sets
- Hardware acceleration
- Institutional trust

Not in secrecy.

CHAPTER 53 — LIMITS, RISKS, AND OPEN QUESTIONS

53.1 Known Limits

- Verification is heavier than lattice schemes
- Conceptual novelty slows adoption
- Formal reductions are incomplete

These are acknowledged costs.

53.2 Open Research Questions

- Tight bounds on ambiguity width
- Optimal operator families
- Formal connections to complexity theory
- Long-term behavior under adaptive attacks

Θ-Crypt is a platform for research, not a finished doctrine.

CHAPTER 54 — FINAL SYNTHESIS

Θ-Crypt represents a deliberate break with a half-century of cryptographic habit.

It does not ask:

“How hard is it to compute the secret?”

It asks:

“Is the secret even determined by what you can see?”

By grounding cryptography in non-canonicity—an idea forced upon mathematics by Ramanujan, clarified by Zwegers, and articulated by Zagier—this architecture builds security not from bravado, but from epistemic humility.

The system is:

- Explicit in its assumptions
- Modest in its claims
- Aggressive in its testing posture

If it fails, it will fail visibly.
If it succeeds, it will succeed quietly—by making certain attacks meaningless rather than merely expensive.

I. APPENDICES (FORMAL + TECHNICAL)

APPENDIX A — FORMAL SECURITY GAMES (COMPLETE DEFINITIONS)

A.1 MMIP-ID (Identification Game)

Let Π be public parameters.

Setup

1. Challenger samples completion data $Y \leftarrow \text{KeySpace}(\Pi)$
2. Challenger derives transcript $T \leftarrow \text{Transcript}(Y)$
3. Challenger gives T to adversary $\mathcal{A}$

Adversary Access

- Evaluation oracle Eval_f
- Verification oracle $\text{Verify}(\Pi, T, \cdot)$
- Polynomial-time classical + quantum computation

Goal
$\mathcal{A}$ outputs $\hat{Y}$

Winning Condition
$\text{Adv} = \Pr[\hat{Y} = Y] - 1 / |\text{KeySpace}(\Pi)|$

Security requires $\text{Adv} \leq \text{negl}(\lambda)$

A.2 MMIP-D (Distinguishability Game)

Setup

1. Sample $Y_0, Y_1 \leftarrow \text{KeySpace}(\Pi)$, $Y_0 \neq Y_1$
2. Sample $b \leftarrow \{0,1\}$
3. Compute $T \leftarrow \text{Transcript}(Y\_b)$
4. Give T to $\mathcal{A}$

Goal
$\mathcal{A}$ outputs b'

Winning Condition
$\Pr[b' = b] \leq 1/2 + \text{negl}(\lambda)$

A.3 MMIP-F (Forgery Game)

Setup

1. Challenger samples $Y \leftarrow \text{KeySpace}(\Pi)$
2. Challenger gives $T \leftarrow \text{Transcript}(Y)$ to $\mathcal{A}$

Goal
$\mathcal{A}$ outputs $T' \neq T$

Winning Condition
$\text{Verify}(\Pi, T', \cdot) = \text{accept}$
AND
$\nexists\ Y'$ s.t. $\text{Transcript}(Y') = T'$

APPENDIX B — TRANSCRIPT FORMAL SPECIFICATION

```
Transcript T ::= {
    version_id,
    field_modulus p,
    truncation_degree N,
    weight k,
    operator_seeds [σ₁ … σ_L],
    coefficient_vector [a₀ … a_{N-1}],
    evaluation_points [(τ₁, v₁) … (τ_M, v_M)],
    tolerance ε,
    transcript_hash H(T)
}
```

Invariants

- Deterministic generation
- Collision-resistant hashing
- Operator-seed immutability

APPENDIX C — OPERATOR FAMILIES (MINIMAL SET)

C.1 Modular Transport Operator

```
Op_γ(T):
    for each (τ_i, v_i):
        check consistency of v_i with transformed τ_i
```

C.2 Folding / Convolution Operator

```
Op_fold(T):
    compute discrete convolution of coefficients
    enforce symmetry-breaking invariant
```

C.3 Cusp Stress Operator

```
Op_cusp(T):
    evaluate behavior near roots of unity
    enforce bounded growth window
```

All operators must:

- Be deterministic
- Be completion-blind
- Preserve ambiguity

APPENDIX D — FAILURE MODES AND SAFETY

Failure outcomes allowed:

- reject
- ⊥
- random key

Failure outcomes forbidden:

- partial identity leakage
- narrowing of ambiguity class
- oracle distinguishability

II. REFERENCE IMPLEMENTATION SKELETON

This is **engineer-ready**.

Language Choice: Rust (Core), Python (Harness)

Core Trait Definitions (Rust)

```
pub trait Transcript {
    fn verify(&self, candidate: &Candidate) -> bool;
    fn hash(&self) -> Hash;
}

pub trait CompletionData {
    fn extract(&self, transcript: &Transcript, ct: &Ciphertext) ->
Option<Selector>;
}

pub trait Operator {
    fn apply(&self, transcript: &Transcript, candidate: &Candidate) ->
bool;
}
```

Θ-KEM Skeleton

```
pub fn keygen(params: &Params) -> (PublicKey, SecretKey) {
    let completion = CompletionData::sample(params);
    let transcript = Transcript::from_completion(&completion, params);
    (PublicKey { transcript }, SecretKey { completion })
}

pub fn encaps(pk: &PublicKey) -> (Ciphertext, SharedKey) {
    let s = Selector::random();
    let ct = Ciphertext::embed(pk, &s);
    let k = KDF::derive(&s, &pk, &ct);
    (ct, k)
}

pub fn decaps(sk: &SecretKey, pk: &PublicKey, ct: &Ciphertext)
    -> Option<SharedKey>
{
    if !pk.transcript.verify(ct) {
        return None;
    }
    let s = sk.completion.extract(&pk.transcript, ct)?;
    Some(KDF::derive(&s, pk, ct))
}
```

Attack Harness (Python)

```
def ambiguity_estimator(transcript, verifier, attempts=10_000):
    survivors = []
    for _ in range(attempts):
        candidate = random_candidate()
        if verifier(transcript, candidate):
            survivors.append(candidate)
    return len(survivors)
```

This is **mandatory** tooling.

III. INVESTOR-FACING TECHNICAL MEMORANDUM

Executive Summary

This project introduces a **new cryptographic security regime** based on *controlled non-identifiability* rather than computational inversion hardness.

Unlike RSA, ECC, or lattice-based systems—where public data uniquely determines the secret—this architecture constructs public transcripts that are **mathematically compatible with exponentially many private identities**. Security arises from the impossibility of selecting the correct identity without private global data.

This approach is **orthogonal** to known quantum attacks (Shor, Grover), not by claiming immunity, but by **removing the structural preconditions those algorithms exploit**.

What Exists Today

- Fully specified cryptographic ontology
- Formal attacker models and security games
- Concrete KEM and signature constructions
- Discretized transcript and verifier architecture
- Software and hardware feasibility path

This is **not a whitepaper**. It is a technical system.

What the Investment Buys

- First-mover position in a **new cryptographic foundation**
- Licensable implementations (software + hardware)
- Strategic hedge against PQC monoculture risk
- Long-horizon IP grounded in pure mathematics

What This Is Not

- Not a replacement for PQC today
- Not a speed-optimized scheme
- Not a marketing-driven "quantum-proof" claim

It is **infrastructure-grade research with deployment trajectory**.

Risk Disclosure

- Novelty risk (real, acknowledged)
- Standardization timeline (long)
- Verification cost (higher than lattices)

These risks are **structural**, not executional.

IV. ACADEMIC MONOGRAPH VERSION

Title

Cryptography Beyond Canonicity:
Non-Identifiability as a Security Primitive

Abstract (Journal Length)

We propose a cryptographic architecture grounded in mathematical non-canonicity rather than computational inversion hardness. Drawing on the theory of mock modular forms, we construct public transcripts that are locally complete yet globally underdetermining, admitting exponentially many valid private completions. Security is formalized via identification and distinguishability games under a restricted observables model. We show how this epistemic asymmetry survives discretization, enabling concrete key exchange and signature schemes whose attack surface is orthogonal to

known quantum algorithms. This work introduces controlled non-identifiability as a third regime of cryptographic security, complementing information-theoretic secrecy and computational hardness.

Structure (Journal-Ready)

1. Introduction and Motivation
2. Non-Canonicity in Mock Modular Forms
3. Restricted Observables Model
4. Cryptographic Formalization
5. Discretization and Verification
6. Key Exchange and Signatures
7. Security Analysis
8. Discussion and Open Problems

About the Author

Marcos Elias is an engineer, mathematician, and systems architect whose work spans quantitative finance, cryptography, complex systems, and advanced computation. He holds multiple doctoral-level trainings in mathematics and computer science, with a research trajectory deeply rooted in probability theory, stochastic calculus, and the mathematical foundations of uncertainty.

His intellectual lineage traces back to the Russian school of probability and analysis, with a particular emphasis on the legacy of Andrei Kolmogorov and Kiyoshi Itō. Over the course of more than three decades, Elias has worked at the intersection of theory and implementation, designing large-scale quantitative systems for high-risk environments where traditional modeling assumptions fail.

Beyond academia, he is a serial entrepreneur and founder of multiple deep-technology initiatives spanning quantum algorithms, non-anthropocentric artificial intelligence, cryptographic systems, and advanced optimization frameworks. His work consistently focuses on regimes where standard notions of invertibility, optimization, and identifiability break down—whether in financial markets, complex adaptive systems, or information security.

The present work emerges from a long-standing dissatisfaction with the dominant epistemology of modern cryptography. Rather than treating security as a function of computational difficulty alone, Elias approaches it as a problem of *knowledge structure*: what can be observed, what can be inferred, and what remains fundamentally underdetermined. Drawing on the mathematics of mock modular and mock theta functions—objects that resist canonical identification despite complete local observability—he proposes a new cryptographic paradigm grounded in controlled non-canonicity.

This manuscript reflects both a foundational and a constructive ambition. It is not merely a theoretical exploration, but an attempt to translate deep mathematical asymmetries into implementable cryptographic architectures, complete with formal security models, engineering constraints, and deployment pathways. The work is intended for mathematicians, cryptographers, engineers, and institutional decision-makers who recognize that the post-quantum transition is not only a technological shift, but an ontological one.

Marcos Elias is the Founder and Chair of the Ramanujan Institute, where the foundational research underlying this work is anchored. Applied implementations and downstream systems are developed through affiliated research and engineering entities, preserving a strict separation between mathematical ontology and commercial deployment.