# EQSIGN: Practical Digital Signatures from the Non-Abelian Hidden Subgroup Problem and Information Theoretic Equivocation

Samuel Lavery

Trustless Privacy Inc.
sam@trustlessprivacy.com
Signal:samlavery.07

December 27, 2024

**Abstract**

We present a novel digital signature scheme grounded in non-commutative cryptography and implemented over a bilinear matrix group platform. At the core of our design is a unique equivocation function that obfuscates intermediate elements, effectively concealing outputs and minimizing observable information leakage. To the best of our knowledge, this is the first digital signature scheme to combine information-theoretic security with computational hardness, relying on a challenging instance of the Non-Abelian Hidden Subgroup Problem (NAHSP) and strengthened by practical guarantees. This dual-layered security approach ensures robustness against both classical and quantum adversaries while maintaining communication overheads competitive with RSA. Our work represents a significant advancement toward efficient, quantum-resilient digital signatures for real-world applications. This paper is an early pre-release intended to invite collaboration and feedback. The work is presented for research purposes only and is not intended for use in production systems.

# Contents

# 1 Introduction

## 1.1 The State of the Art

Modern quantum resilient cryptographic signature schemes are primarily based on structured lattice or hash based cryptography, each with a unique set of disadvantages. Lattice schemes, such as ML-DSA[2] leverage module lattices, which by virtue of their internal algebraic structure provide a potential avenue for quantum cryptanalysis, and variable sizes which are 5x the size of the signature primitives in use today. Hash based signature schemes such as SLH-DSA[1], to their credit, have virtually no exploitable algebraic structure and have small public keys, but produce relatively huge signatures requiring a substantial amount of computational resources to produce. The Falcon[11] signature algorithm, which has also been selected by NIST for standardization, is more communication efficient compared to ML-DSA, but relies on significantly more internal structure in the form of NTRU lattices, and remains challenging to implement in constant time due to reliance on floating point operations.

Signature solutions under consideration currently can generally be categorized as Lattice, Code based, Multivariate, Isogeny, and symmetric cryptography constructions with multiple variants remaining under consideration by NIST. Unfortunately, out of the 14 digital signature schemes selected for the NIST's second 'onramp' round of analysis, when considering communication cost alone, only SQISign[3] can be considered a reasonable replacement for current quantum weak signatures. Unfortunately, SQISign has the highest computational cost when compared to every other alternative, making widespread adoption fairly impractical. Additionally as isogeny based systems are fairly novel, and the predecessor SIDH was broken classically in a rather spectacular fashion[7], a bit of healthy skepticism is warranted. The other 13 candidates have similar or higher communications overhead compared to Falcon, and are based on largely untested computational hardness assumptions.

As the modern digital world has been built using digital signatures based on ECDSA and RSA, without a quantum resilient replacement with similar communications overhead, we are faced with significant challenges. We will have no choice but to redesign hardware, software, and protocols to accommodate these vastly increased communication costs, which will cost billions of dollars and take decades to migrate to. The current situation for digital signatures appears to be quite bleak. At this point, the probability that a provable quantum resilient digital signature solution that is both communication and computationally efficient will emerge is nearly inconceivable.

## 1.2 Secure and Efficient Signatures from the Non-Abelian Hidden Subgroup Problem and Information Theory

Non-commutative cryptography[10] has been an area of research since before the 1990s, and is still a credible basis for quantum resilient cryptographic systems. Previous attempts to construct secure non-commutative cryptographic systems were either communications inefficient compared to lattice schemes[8][4][6], or compromised generally by algebraic cryptanalysis. With most schemes having core algebraic and structural weaknesses[9], the bulk of research inertia has shifted to lattice or other well known problem groups with the perceived potential to achieve quantum resilience.

While there have been several attempts to leverage matrix groups for non-commutative cryptography[5], the chosen matrices were structured, lacked entropy, had exploitable subgroups, or exploitable linear relationships. These schemes relied on matrix equivalence or conjugacy problem hardness assumptions, which turned out to be less robust than anticipated. Our work uses a pair of dense, full rank random matrices, forming a bilinear group. By using non-correlated matrices, we mitigate attacks leveraging structural relationships. With half of the matrix group randomized each operation, observed correlation only applies to a single matrix pair. This acts to effectively minimize exploitable relationship information. This bilinear matrix group structure, combined with a novel equivocation function form the building blocks of our instance of the non-Abelian Hidden Subgroup Problem (NAHSP).

Non-commutative cryptographic schemes have received moderate attention within the cryptographic community, yet the Non-Abelian Hidden Subgroup Problem (NAHSP) remains comparatively under-explored. In contrast, the NAHSP is recognized as one of the most significant and challenging problems in quantum algorithmic research. For over two decades, intensive study of the NAHSP has not only failed to produce efficient quantum algorithms for general non-abelian groups, but has also yielded numerous negative results. Unlike problems that are merely conjectured to be quantum-resistant, the NAHSP has demonstrated resilience against decades of quantum attack attempts, establishing itself as a robust foundation for constructing quantum-resilient cryptographic systems.

In the context of this construction, the low dimensions $n = \{64, 128, 256\}$ and small modulus size $q = 257$ do not directly result in an intractable NAHSP instance. Normally, an adversary observing a small number of intermediate outputs could construct a solvable system of linear equations via modular arithmetic and Gaussian elimination. However, randomization acts to uniformly project intermediate output elements of matrix calculations across a large ambient module space $\mathbb{Z}_n^q$. We leverage information-theoretic principles[13], equivocation and mutual entropy, to prove practical information theoretic security guarantees. While we do not claim the perfect secrecy of the one-time pad[12], we use information theoretic arguments to quantify the computational infeasibility of reconstructing the problem. We employ a novel equivocation function to partition the ambient space into equivalence classes, dispersing secret entropy across a class of indistinguishable valid pre-images. If the original output is the needle, the adversary is given the haystack. This core disambiguation ensures that each equivalence class is populated with indistinguishable pre-images, obfuscating the relationship between inputs and outputs. This indistinguishability enables the transformation our computationally hard matrix problem into an information theoretic one. Thus, the challenge for an adversary shifts from using linear algebra to solve a well posed system of equations to identifying the correct indistinguishable elements needed to construct it.

This work employs a hybrid security model that amplifies computational complexity with information-theoretic security. By leveraging uniform distribution and indistinguishability, we transform the computational problem of solving the NAHSP into an information-theoretic challenge. The resilience of the NAHSP to quantum attack attempts underscores its suitability as a basis for this form of construction.

In practical terms, this work challenges the prevailing notion that achieving quantum resilience necessitates a significant increase in communication overhead. Our construction integrates the structural advantages of non-abelian matrix groups with information-theoretic principles, offering an alternative direction for designing cryptographic systems that balance quantum resistance with real-world usability. The short signatures and public keys comparable to ECDSA and RSA may be especially useful in highly constrained network environments, where more established lattice based schemes struggle to integrate.

This work is a preprint released to facilitate discussion and collaboration. Updates and refinements will follow as necessary.

# 2 High-Level Description

## 2.1 Core Representation

Our instance of the Non-Abelian Hidden Subgroup Problem (NAHSP) is defined within the context of a bilinear matrix group $G$. The group operation, matrix-vector multiplication, involves a public matrix $A$ and a secret vector $x$, while the hidden subgroup $N$ is implicitly defined through transformations induced by the secret matrix $U$. Specifically, the outputs $t \equiv Ax \mod q$ and $t' \equiv Ut \mod q$ are observed, with $U$ and $x$ remaining hidden. To amplify cryptographic hardness, the equivocation function $px$ maps these outputs into indistinguishable equivalence classes, obfuscating the relationship between $A$, $U$, $x$, and the observed results.

Our variant can be concisely represented for key generation as:

$$t \equiv A \cdot x \mod q,$$

$$t' \equiv U \cdot t \mod q,$$

$$pk = t'' \equiv px(t') \mod q,$$

For signing as:

$$t \equiv B \cdot x \mod q,$$

$$t' \equiv U \cdot t \mod q,$$

$$sig = t'' \equiv px(t' \circ J(C1)) \mod q,$$

For validation as:

$$LHS \equiv B \cdot (pk \circ J(C2)) \mod q,$$

$$LHS' \equiv px(LHS \circ J(C1)) \mod q,$$

$$RHS \equiv A \cdot (sig \circ J(C2)) \mod q,$$

$$RHS' \equiv px(RHS) \mod q,$$

$$LHS' \stackrel{?}{=} RHS'$$

where:

- $A$: Dense, full-rank, random public matrix used for key generation and verification.
- $B$: Dense, full-rank, random public matrix used for signing and verification.
- $U$: Dense, full-rank, random, uncorrelated private matrix.
- $x$: Secret vector uniformly sampled from $\mathbb{Z}^n_{1-256} = \{1, 2, \ldots, 256\}^n$, ensuring no zero entries.

$t$: Element in the right subgroup $H_{\text{right}} \subseteq G$, spanning the ambient space and serving as a hidden input to $H_{\text{left}}$.

$J()$: Secure hash function (e.g., SHA3/SHAKE) producing pseudorandom outputs in $\mathbb{Z}_q$.

$t'$: Element in the hidden left subgroup $N \subseteq G$.

$px()$: Mapping function projecting $t'$ into equivalence classes, ensuring computational infeasibility of $N$'s recovery.

$t''$: Obfuscated output element in the superset $N' \subseteq G$.

$q$: Prime modulus defining the finite field $\mathbb{Z}_q$.

$fs$: Fiat-Shamir heuristic in $\mathbb{Z}_q^n$, binding the public key, message, and randomness to the signature context.

$r$: Signature randomizer value in $\mathbb{Z}_q^n$, mitigating replay attacks and enhancing security against sEU-CMA.

$pk$: Public key element in $\mathbb{Z}_q^n$.

$C1$: Forgery constraint 1, an element in $\mathbb{Z}_q^n$, derived as a hash of intermediate context values related to $pk \cdot B$.

$C2$: Forgery constraint 2, an element in $\mathbb{Z}_q^n$, derived as a hash of $pk$, $sig$, and other context elements.

$\circ$: Element-wise multiplication operation.

$\cdot$: Matrix-vector product operation.

## 2.2 Construction

### 2.2.1 Definition of the Group $G$ and Subgroups $H_{\text{right}}$ and $H_{\text{left}}$

**Ambient Group $G_{\text{ambient}}$:**

The ambient modular group defines the space of possible elements as:

$$G_{\text{ambient}} = \mathbb{Z}_q^n,$$

representing the set of all $n$-dimensional vectors over $\mathbb{Z}_q$. The group operation is matrix-vector multiplication modulo $q$. This expansive space provides a form of limited one-time pad security, where the inherent size and approximate uniform partitioning of $G_{\text{ambient}}$ by $px(\cdot)$ ensure that individual elements are computationally indistinguishable without knowledge of the specific subgroup structure.

**Working Group $G$:**

The working group $G$ in the context of the NAHSP is defined as the group generated by the subgroups $H_{\text{right}}$ and $H_{\text{left}}$:

$$G = \langle H_{\text{right}}, H_{\text{left}} \rangle,$$

where each subgroup introduces hidden structure critical to cryptographic security.

**Subgroup $H_{\text{right}}$:**

The subgroup $H_{\text{right}}$ generates the space of input elements $t$ based on the public matrix $A$:

$$H_{\text{right}} = \{t \mid t = A \cdot x \mod q,\, x \in \mathbb{Z}_q^n\},$$

where:

- $A \in \mathbb{Z}_q^{n \times n}$: Dense, full-rank, random public matrix.
- $x \in \mathbb{Z}_q^n$: Secret vector sampled uniformly from $\{1, \ldots, q-1\}^n$.

**Subgroup $H_{\text{left}}$:**

The subgroup $H_{\text{left}}$ maps elements $t$ from $H_{\text{right}}$ through the private matrix $U$:

$$H_{\text{left}} = \{t' \mid t' = U \cdot t \mod q,\, t \in H_{\text{right}}\},$$

where:

- $U \in \mathbb{Z}_q^{n \times n}$: Hidden, dense, full-rank private matrix.
- $t \in H_{\text{right}}$: Hidden input generated by the public matrix $A$.

**Hidden Subgroup $N$:**

The hidden subgroup $N$ is a normal subgroup embedded within $H_{\text{left}}$:

$$N = \{t' \in H_{\text{left}} \mid t' \cdot t^{-1} \in H_{\text{left}},\, \forall t \in H_{\text{right}}\}.$$

The structure of $N$ ensures the following cryptographic properties:

- Normality and Algebraic Consistency: The subgroup $N$ maintains its normality within $H_{\text{left}}$, preserving the relationship $gN = Ng$ for all $g \in G$. This is essential for reductions to the Non-Abelian Hidden Subgroup Problem (NAHSP).
- Obfuscation by Subgroup Relationships: Elements of $N$ are indistinguishable from non-members without knowledge of the private transformations $U$ and $x$, as the coset structure of $N$ is obscured by the combined transformations within $H_{\text{left}}$.
- Cryptographic Hardness: Recovering $N$ requires solving the NAHSP, which is computationally infeasible under both classical and quantum adversarial models.

**Role of the Hidden Matrix $U$:**

The hidden matrix $U$ introduces non-commutative transformations that amplify cryptographic hardness by ensuring that the output space spans the full ambient group:

- Full-Rank Transformation: The full-rank nature of $U$ guarantees that the transformation $t' = U \cdot t \mod q$ spans the entire output space $\mathbb{Z}_q^n$. This ensures that the final output is not constrained by structural dependencies, maximizing the entropy of $t'$.

- Non-Commutative Action: The transformation $U \cdot t \mod q$ introduces non-commutative operations, breaking linear relationships and obfuscating the structure of $H_{\text{left}}$. This makes it computationally infeasible for an adversary to recover $U$ or $t$ without solving the underlying subgroup problem.

- Preservation of Subgroup Properties: The transformation maintains the algebraic properties of $H_{\text{left}}$ and preserves the coset structure of the hidden subgroup $N$ under conjugation:
$$gN = Ng, \quad \forall g \in G.$$
This preservation is critical for ensuring cryptographic consistency and enabling reductions to the NAHSP.

**Role of the Public Matrix $A$ and Secret Vector $x$:**

The public matrix $A$ and the secret vector $x$ jointly ensure that transformations are randomized and uniformly distributed across the input space, enabling the outputs to fully span the group $\mathbb{Z}_q^n$. Together, $A$ and $x$ mitigate adversarial attacks by introducing fresh randomness and structural complexity for every operation:

- Full-Rank Input Transformation: The full-rank nature of $A$ ensures that $t = A \cdot x \mod q$ spans the entire space of valid inputs, $\mathbb{Z}_q^n$. This guarantees that every operation begins with a uniformly distributed element, eliminating structural biases.

- Per-Operation Randomization: The randomized public matrix $A$, combined with the secret vector $x$, ensures that transformations differ across operations. This randomness prevents correlation attacks and ensures that adversaries cannot infer relationships between consecutive operations.

- Obfuscation of Intermediate Values: The secret vector $x$ acts as a one-time secret for each transformation, ensuring that the intermediate value $t$ remains private and uncorrelated with $U$.

- Resistance to Chosen Message Attacks: The per-operation uniqueness of $A$ and $x$ ensures that information observed from one operation cannot be reused or leveraged to attack subsequent operations. This protects the scheme from signature re-use attacks.

**Security Mechanisms:**

The security of the scheme is rooted in the computational hardness of solving the Non-Abelian Hidden Subgroup Problem (NAHSP):

- Computational Hardness: Recovering the hidden subgroup $N$ or reconstructing $U$ and $x$ from observed outputs requires solving the NAHSP, a problem resistant to both classical and quantum adversaries.

- Structural Complexity: The layered transformations through $A$ and $U$, combined with the subgroup relationships between $H_{\text{right}}$, $H_{\text{left}}$, and $N$, amplify the inherent difficulty of the problem.

**Adversary's Computational Task:**

The adversary's goal is to recover $N$ and reconstruct $U$ and $x$. This task is rendered cryptographically non-trivial due to:

- Full-Rank Transformations: The full-rank nature of $A$ and $U$ ensures that observed outputs span the entire group $\mathbb{Z}_q^n$, preventing structural shortcuts or biases.
- Non-Commutative Action: The non-commutative nature of $U \cdot t \mod q$ disrupts linear relationships, obfuscating the subgroup structure within $H_{\text{left}}$ and making it computationally infeasible to infer $U$ or $x$ directly.
- Combinatorial Complexity: The adversary must distinguish subgroup elements in $G_{\text{ambient}}$, which involves an exponentially large search space without access to the private transformations.

**Adversary's Information-Theoretic Challenges:** In addition to the computational hardness posed by NAHSP, the scheme introduces inherent information-theoretic barriers that further obfuscate the relationships between inputs, outputs, and subgroup membership:

- Uniformly Distributed Outputs: The transformations $t = A \cdot x \mod q$ and $t' = U \cdot t \mod q$ ensure that the outputs are uniformly distributed over $\mathbb{Z}_q^n$. This uniformity prevents adversaries from inferring structural dependencies or narrowing the search space.
- Ambiguity of Subgroup Membership: Without access to the private matrix $U$ or secret vector $x$, adversaries cannot distinguish elements of the hidden subgroup $N$ from non-members within $H_{\text{left}}$, as the coset relationships are fully obscured.
- Exponential Pre-Image Set Sizes: The adversary must contend with an exponentially large set of potential pre-images for any observed output, making it infeasible to isolate the correct subgroup elements even under exhaustive search.

**Dual-Layered Security:**

The security of the scheme combines:

- Computational Hardness of NAHSP: The cryptographic guarantees are fundamentally tied to the infeasibility of solving NAHSP, a problem resistant to both classical and quantum adversaries.
- Information-Theoretic Obfuscation: The transformations induced by $A$, $U$, and $x$ ensure that observed outputs retain high entropy, preserving indistinguishability across the full ambient group $G_{\text{ambient}}$.

# 3 Problem Statement

The objective of this cryptographic scheme is to secure the hidden subgroup $N$, ensuring its structure remains concealed from adversaries. This is achieved by obfuscating the relationships between the public basis $A$, the hidden matrix $U$, and the secret vector $x$, while leveraging a lossy mapping function $px(\cdot)$ to induce equivalence classes. The scheme employs dual-layered security mechanisms: computational hardness from the Non-Abelian Hidden Subgroup Problem (NAHSP) and information-theoretic obfuscation from $px(\cdot)$.

## Adversary's Knowledge

The adversary has access to:

- The public matrix $A$, which spans the ambient group $G_{\text{ambient}}$ and defines $H_{\text{right}}$,
- The obfuscated output $t''$, resulting from applying the lossy mapping $px(\cdot)$ to elements of $H_{\text{left}}$.

## Adversary's Limitations

The adversary does not have access to:

- The secret vector $x$, used in the transformation $t = A \cdot x \mod q$,
- The intermediate vector $t$, which resides within $H_{\text{right}}$,
- The private matrix $U$, responsible for mapping elements from $H_{\text{right}}$ to $H_{\text{left}}$ and defining the hidden subgroup $N$.

## Equivocation of the Mapping Function $px(\cdot)$

The mapping function $px(\cdot)$ is a lossy transformation that projects elements of the ambient group $G_{\text{ambient}}$ into equivalence classes. This mapping introduces significant obfuscation, ensuring that subgroup membership cannot be determined feasibly without knowledge of the private components.

**Properties of $px(\cdot)$:**

- $px : G_{\text{ambient}} \to$ Equivalence Classes, where each equivalence class contains indistinguishable pre-images.
- The function disrupts linear and algebraic relationships within the group, rendering coset structures unobservable.
- Without access to $x$ or $U$, distinguishing subgroup members from non-members within equivalence classes is infeasible.

**Impact of Equivocation:** The lossy nature of $px(\cdot)$ exponentially increases the adversary's search space by creating a many-to-one mapping:

- Pre-images of $px(\cdot)$ form equivalence classes that mask coset relationships within $H_{\text{left}}$,
- The adversary must contend with an exponentially large number of indistinguishable elements, effectively reducing any observed output to noise.

- By the *Data Processing Inequality (DPI)*, a result derived from Shannon's foundational work in information theory (Theorem of Noisy Channels), the mutual entropy between the input and output of $px(\cdot)$ is provably reduced through this lossy mapping. The surjectivity of $px(\cdot)$ increases the likelihood that the output's entropy is maximized relative to the adversary's view, rendering it statistically indistinguishable from random noise and enhancing equivocation.

## Chaining Mechanism

The chaining mechanism enhances security by linking independent problem instances through intermediate outputs. Each stage introduces unique secrets and transformations, ensuring that the overall system is resilient against adversarial attacks. To give an example in the signature context with six chained instances $k = 6$:

For $k = 0$:

$$t_0 \equiv B_0 \cdot x_0 \mod q,$$
$$t'_0 \equiv U_0 \cdot t_0 \mod q,$$
$$t''_0 \equiv px(t'_0 \circ J(C1_0)) \mod q,$$

For $k = 1$ to 5:

$$t_k \equiv B_k \cdot (x_k \circ t''_{k-1}) \mod q,$$
$$t'_k \equiv U_k \cdot t_k \mod q,$$
$$t''_k \equiv px(t'_k \circ J(C1_k)) \mod q,$$

Where the final output $sig = t''_5$.

**Mechanism:**

- Intermediate outputs $t''_k$ from one stage are passed as hidden inputs to the next stage,
- Each stage employs independent secrets $x_k$, matrices $U_k$, and public matrices $A_k$, ensuring randomness and unlinkability.

**Security Benefits:**

- Error Propagation: Any errors or approximations in recovering one stage amplify across subsequent stages, compounding the adversary's difficulty.
- Independence of Stages: Knowledge of secrets from one stage does not simplify reconstruction of secrets from subsequent stages due to the introduction of fresh randomness and transformations.
- Unlinkability: Intermediate values $t_k$ and $t'_k$ remain hidden, ensuring that adversaries cannot correlate outputs across stages.
- Hardness Amplification: Solving one instance of the chained system yields no useful information for subsequent stages. The adversary must solve all instances simultaneously, which exponentially increases the overall complexity of the problem.

# Verification and Forgery Mitigation

The verification process ensures the integrity of the transformations applied during signing and key generation, validating that the observed signature $\sigma$ corresponds to the public key $pk$ and the private components $x$ and $U$, without revealing these private components. By leveraging the mapping function $px(\cdot)$, contextual hash constraints, and entropy checks, the scheme mitigates forgery attempts while maintaining soundness and completeness.

**Verification Equation:** The verification process compares two transformed outputs derived from the public key $pk$ and the signature $\sigma$, iteratively refining them under contextual constraints $C1$ and $C2$:

- $C1_k$: Represents the intermediate value derived during signing and verification, computed as $J(pk \cdot B_k)^3$, ensuring consistency between signing and verification.
- $C2_k$: A hash of $pk$, $\sigma$, $FS$ (Fiat-Shamir Heuristic), and $r$ (message randomizer), binding the signature to its context and mitigating signature malleability.

The verification equation is computed as follows:

$$LHS_0 \leftarrow pk, \quad RHS_0 \leftarrow \sigma$$

For $k = 0$ to $k - 2$:

$$LHS_{k+1} = px(B_k \cdot (LHS_k \circ J(C2_k)) \mod q), \quad RHS_{k+1} = px(A_k \cdot (RHS_k \circ J(C1_k)) \mod q).$$

For the final stage ($k = k - 1$):

$$LHS_k = B_k \cdot (LHS_{k-1} \circ J(C2_k)) \mod q, \quad RHS_k = A_k \cdot (RHS_{k-1} \circ J(C1_k)) \mod q.$$

A signature $\sigma$ is valid if and only if:

$$LHS_k \stackrel{?}{=} RHS_k.$$

**Key Components:**

- Public Matrices $A$ and $B$: Define the observable transformations applied to the public key and signature during verification.
- Secure Hash Function $J()$: Produces contextual constraints $C1$ and $C2$, binding the signature and public key to the specific signing context.
- Mapping Function $px(\cdot)$: Masks equivalence classes to ensure subgroup membership remains indistinguishable, preventing adversarial reconstruction of $x$ or $U$.
- Observed Entropy Constraint: During both signature generation and validation, the observed entropy and randomness of signature candidates are checked to ensure they statistically represent approximately $1/10$ of the combinatorial possibilities. This constraint aligns with information-theoretic principles, ensuring that signatures exhibit near-randomness and resist predictability.

**Forgery Mitigation:** Forgery is mitigated through the interaction of several mechanisms:

- Contextual Hash Constraints: The hash constraints $C1$ and $C2$ bind the signature and public key to specific contextual values, ensuring that signatures cannot be reused or manipulated across different contexts.

- Lossy Mapping Function and Probability of Forgery: The lossy mapping function $px(\cdot)$ obfuscates subgroup membership, making it computationally infeasible for adversaries to generate valid signatures without access to the private keys. The probability of a successful forgery at each level $k$ is determined by the ratio of equivalence class members $m_k$ to the total number of equivalence classes $n_k$. For each level, the adversary must generate a value that maps to the correct equivalence class under $px(\cdot)$, resulting in a success probability of approximately $\frac{m_k}{n_k}$. Across $K$ levels, the cumulative probability of forging a valid signature is given by:

$$P_{\text{forgery}} \sim \prod_{k=0}^{K-1} \frac{m_k}{n_k}.$$

  This product reflects the compounding difficulty of forging a signature, as the adversary must satisfy all constraints simultaneously. The carefully chosen ratio of equivalence class members to equivalence class numbers ensures that the probability of a successful forgery remains negligibly small.

- Fixed and Randomized Public Matrices: The public matrix $A$ is fixed and defines the ambient group structure, while the signing matrix $B$ is randomized and tied to the message. This dynamic prevents adversaries from correlating multiple signatures to infer structural relationships or exploit reuse.

- Observed Entropy Constraint: The observed entropy constraint ensures that edge-case scenarios are effectively mitigated. Signature candidates are required to statistically adhere to near-randomness, aligning with approximately $1/10$ of the combinatorial possibilities. This increases the difficulty of identifying predictable or exploitable patterns, enhancing resilience to forgery.

**Soundness and Completeness:**

- Completeness: Any valid signature $\sigma$, generated using the correct private components $x$ and $U$, satisfies the verification equation.

- Soundness: Any invalid signature $\sigma'$, generated without access to the private components, fails verification. This failure arises because $\sigma'$ maps to incorrect equivalence classes under $px(\cdot)$, and fails entropy checks for statistical validity.

*Related proofs of soundness, completeness, equivocation, and equivalence class ratio impact will be presented as part of the formal proof of existential unforgeability under chosen message attacks (EUF-CMA) in a subsequent section.*

## Adversary's Tasks: Key Recovery vs. Forgery

The adversary's objectives can be categorized as follows:

- Key Recovery: Reconstructing the hidden subgroup $N$ by recovering $U$ and $x$:
  - This requires solving the NAHSP, an infeasible task due to the equivocation induced by $px(\cdot)$ and the computational hardness of the problem.
- Forgery: Generating a valid signature $\sigma'$ without access to the private keys:
  - This requires reversing branching layers of $px(\cdot)$ to identify valid subgroup elements, which is infeasible due to the lossy nature of the mapping and the randomness introduced at each stage.

## Summary

The scheme achieves robust security by:

- Obfuscating subgroup structure through the lossy mapping $px(\cdot)$,
- Amplifying adversarial difficulty with the chaining mechanism, ensuring that errors propagate across stages,
- Maintaining soundness and completeness in the verification process, ensuring only valid signatures satisfy the verification equation,
- Combining computational hardness from the NAHSP with information-theoretic obfuscation from $px(\cdot)$, ensuring resilience against both classical and quantum adversaries.

# 4 Preliminary Results and Contributions

This work introduces a novel digital signature scheme that incorporates both **information-theoretic security** and **computational hardness**, explicitly tied to the Non-Abelian Hidden Subgroup Problem (NAHSP). While the results presented are preliminary, they suggest a promising approach to balancing efficiency, security, and practicality in post-quantum cryptography. The key contributions of this work include:

1. Fiat-Shamir Transformation with Contextual Binding: Leveraging the Fiat-Shamir transformation to securely bind the public key, message, and randomness, generating a challenge seed that derives a unique set of challenge bases per signature. This approach reinforces security and ensures contextual linkage between the signature and the corresponding public key.
2. Chaining Mechanism for Amplified Hardness: Introducing a chaining mechanism that combines independent instances of the matrix-based NAHSP problem. Each stage introduces fresh randomness and transformations, compounding adversarial complexity and amplifying computational difficulty with every additional stage.

3. Verification through Structured Basis Transformations: Designing signature verification as a proof of consistency through structured basis transformations. This approach ensures that transformations applied to the public key and signature align under obfuscated subgroup relationships, preserving algebraic correctness while preventing exploitation of subgroup structures.

4. Information-Theoretic Security via High-Entropy Mapping Functions: Introducing a non-linear, many-to-one mapping function $px(\cdot)$ that compresses the ambient space $G_{\text{ambient}}$ into equivalence classes. The inherent high entropy of $px(\cdot)$'s outputs enforces:

    - Computational indistinguishability of equivalence class elements without access to private keys $x_k$ and $U_k$,
    - Explicit rejection of low-entropy forgeries during verification, adding an entropy-based security layer that complements computational hardness.

5. Novel Matrix-Based Cryptographic Framework: Developing a cryptographic platform based on unrelated public/private matrix groups. The independence of public matrices $A$, $B$, and private matrices $U$, $x$ prevents structural exploitation, supporting efficient signature generation and verification.

6. First Practical Hybrid Information Theoretic and NAHSP-Based Construction: Constructing what we believe to be the first practical digital signature scheme explicitly based on the NAHSP, using non-commutative matrix groups and leveraging both information theoretic functions and chaining mechanisms to ensure robust security.

**Context and Preliminary Results:** The proposed scheme, though unrefined, demonstrates the potential of non-commutative cryptography to address critical challenges in quantum resilience. While further validation, cryptanalysis, and exploration of parameter optimization are necessary, the approach offers:

- Communication Efficiency: Preliminary parameters suggest competitive communication costs compared to RSA signatures and a significant reduction compared to most lattice-based schemes.

- Dual-Layered Security: By combining information-theoretic indistinguishability with computational hardness rooted in the NAHSP, the scheme provides a layered defense against both classical and quantum adversaries.

- Feasibility and Scalability: The use of independent, unrelated public/private matrix groups provides a scalable and tunable platform for balancing security and efficiency, with conservative parameter choices supporting incremental improvements over time.

**Careful Optimism:** While matrix group-based schemes have been explored in the past, this work introduces novel techniques that warrant renewed investigation of non-commutative cryptography. The preliminary results presented here are encouraging but must be rigorously validated by the broader cryptographic community. Future work will focus on parameter tuning, formal proof refinement, performance improvements, and independent verification to solidify the scheme's practical viability and theoretical soundness.

**Complexity Analysis** The mapping function $px(\cdot)$ partitions the group $G$ into equivalence classes, obfuscating the subgroup structure of $N$ and increasing the adversary's difficulty in distinguishing elements. Unlike a group homomorphism, $px(\cdot)$ does not preserve group operations but ensures computational indistinguishability of elements within the same equivalence class. This indistinguishability amplifies the complexity of solving the problem by significantly increasing the effective solution space.

**Impact of $px(\cdot)$:** The hardness of the problem is tied directly to the pre-image count of $px(\cdot)$, which determines the size of equivalence classes and the adversarial search space. Specifically:

- The adversary must navigate all elements in $px^{-1}(g')$ for a given equivalence class $g'$, where $px^{-1}(g')$ contains all pre-images indistinguishable under $px(\cdot)$.
- The size of $px^{-1}(g')$ is determined by the partitioning of the ambient group $G_{\text{ambient}}$ into equivalence classes via the mapping $px(\cdot)$. The hidden subgroup $N$ and the transformations induced by $x$ and $U$ influence the structure of these partitions, but the pre-image membership size fundamentally scales with the size of $G_{\text{ambient}}$ and configuration of $px(\cdot)$.
- The indistinguishability within equivalence classes ensures that structural relationships between elements of $N$ and $G$ are practically obscured, limiting adversarial insights.

**Chaining Mechanism and Amplified Complexity:** The chaining mechanism compounds complexity by propagating errors and introducing additional randomness at each stage, requiring the adversary to solve multiple independent instances of the obfuscated problem. In a single instance, the complexity of solving the problem scales with the size of equivalence classes induced by $px(\cdot)$. For $k$ chained instances, the total complexity is amplified as:

$$O(|px^{-1}(g')|^k),$$

where $|px^{-1}(g')|$ is the size of the pre-image set for a single equivalence class. This reflects:

- The exponential growth of the adversarial search space due to chained instances, requiring reconstruction of intermediate outputs to solve subsequent stages.
- The cascading effect of errors, where small inaccuracies in earlier stages propagate, significantly increasing the difficulty of reconstructing the entire system.

**Security and Complexity Relationship:** The indistinguishability introduced by $px(\cdot)$ ensures that adversaries cannot efficiently distinguish elements of $N$ within equivalence classes or between stages of the chaining mechanism. By explicitly tying the complexity to the pre-image count $|px^{-1}(g')|$, the scheme achieves:

- Scalable Hardness: The size of equivalence classes and the number of chained instances can be tuned to balance efficiency and security.
- Resistance to Structural Attacks: The obfuscation introduced by $px(\cdot)$ disrupts structural relationships, ensuring that subgroup recovery requires infeasible computational resources.

- Cascading Complexity: The chaining mechanism amplifies the adversarial challenge, requiring reconstruction of intermediate outputs across multiple stages, with errors compounding exponentially.

**Practical Observations:** This work does not claim proven hardness for the NAHSP in the general case but leverages its empirical resistance to both classical and quantum attacks. The inclusion of $px(\cdot)$ and chaining mechanisms provides additional layers of security, making the scheme robust under practical cryptographic assumptions while maintaining tunable efficiency for real-world applications.

**Communication Cost:** Perhaps the most relevant result of this work is leveraging information theoretic security to achieve practical signature and public key sizes.

| **Level** | $n$ | **PK (bytes)** | **Sig (bytes)** | $k$ |
|---|---|---|---|---|
| I | 64 | 80 | 96 | 8 |
| III | 128 | 152 | 176 | 6 |
| V | 256 | 288 | 320 | 4 |

Table 1: Public Key, Signature Sizes, and Chain Instances Across Levels

## 4.1 Structure of Remainder of Paper

- Formal mapping of our construct to the Non-Abelian Hidden Subgroup Problem
- Analysis of the information theoretic properties of $px(\cdot)$ in relation to NAHSP
- Proof of Verification Constancy
- Security Notes and Attack Models
- Proof of IND-CPA hardness
- Proof of sEU-CMA security
- Algorithms and Implementation
- Performance Comparison
- Future Work and Conclusion

# 5 Notation and Definitions

Throughout this paper, we give both abstract parameters and concrete example formulas. If specific values are used, they are based on the level III instance, as thus far it has received the bulk of analysis.

# 6 Formal Reduction to the Non-Abelian Hidden Subgroup Problem (NAHSP)

## 6.1 Group Structure and Properties

**Ambient Group $G_{\text{ambient}}$:** The ambient group $G_{\text{ambient}}$ is defined as:

$$G_{\text{ambient}} = \mathrm{GL}(n, \mathbb{Z}_q),$$

the group of invertible $n \times n$ matrices over $\mathbb{Z}_q$, where the group operation is matrix multiplication modulo $q$. This group is non-abelian and serves as the foundational structure for constructing $G$.

**Working Group $G$:** The working group $G$ is constructed as a semidirect product:

$$G = H_{\text{left}} \rtimes H_{\text{right}},$$

where:

- $H_{\text{left}} = \langle U \rangle$, the cyclic subgroup generated by the matrix $U$,
- $H_{\text{right}} = \langle A \rangle$, the cyclic subgroup generated by the matrix $A$.

The automorphism action of $H_{\text{right}}$ on $H_{\text{left}}$ ensures that $G$ is non-abelian:

$$h_R \cdot h_L \cdot h_R^{-1} = \phi_{h_R}(h_L), \quad h_R \in H_{\text{right}},\ h_L \in H_{\text{left}},$$

where $\phi_{h_R}$ is an automorphism of $H_{\text{left}}$.

**Hidden Subgroup $N$:** The hidden subgroup $N$ is defined as $N = H_{\text{left}}$. As established in Lemma 2, $N$ is a normal subgroup of $G$, ensuring that:

$$gNg^{-1} \subseteq N, \quad \forall g \in G.$$

This normality guarantees that $G$ can be partitioned into disjoint cosets of $N$:

$$G = \bigcup_i g_i N, \quad g_i N \cap g_j N = \emptyset \text{ for } i \neq j.$$

## 6.2 Formal Definition of NAHSP

**Definition 1** (Non-Abelian Hidden Subgroup Problem (NAHSP))**.** *The **Non-Abelian Hidden Subgroup Problem** (NAHSP) is defined as follows:*

- ***Input:*** *A finite non-abelian group $G$ and a function $f : G \to S$, where $S$) is the set of left cosets of a hidden subgroup $H \subseteq G$. The function $f$ satisfies:*

$$f(g) = f(g') \iff gH = g'H.$$

- ***Output:*** *Determine the hidden subgroup $H$.*

## 6.3 Oracle Construction and Reduction to NAHSP

**Oracle Function $f$:** Define the oracle function $f : G \to S$ as:

$$f(g) = gN,$$

where $gN$ is the left coset of $N$ containing $g$. The function $f$ satisfies the equivalence relation:

$$f(g) = f(g') \iff gN = g'N.$$

Thus, $f$ is constant on cosets of $N$ and distinct across cosets, fulfilling the requirements of the NAHSP.

**Correspondence with NAHSP:**

- **Group $G$:** The group $G = H_{\text{left}} \rtimes H_{\text{right}}$ serves as the non-abelian group in the NAHSP framework.
- **Hidden Subgroup $H$:** The hidden subgroup $H$ in NAHSP corresponds to $N = H_{\text{left}}$ in our construction.
- **Oracle Function $f$:** The oracle function $f(g) = gN$ encodes the coset structure of $N$, aligning with the oracle requirements of NAHSP.

## 6.4 Hardness of Subgroup Recovery

- **Non-Abelian Structure:** The semidirect product $G = H_{\text{left}} \rtimes H_{\text{right}}$ is inherently non-abelian due to the automorphism action of $H_{\text{right}}$ on $H_{\text{left}}$. This non-abelian nature prohibits the direct application of abelian techniques, such as Fourier analysis, which are pivotal in efficiently solving the Hidden Subgroup Problem (HSP) in abelian groups.
- **Classical Complexity:** Classical algorithms lack the necessary tools to exploit the group structure effectively. They would be compelled to perform exhaustive brute-force enumeration over the cosets of $N$, a task rendered computationally infeasible by the exponential size of $G$. Moreover, the intertwined structures of $H_{\text{left}}$ and $H_{\text{right}}$ offer no combinatorial shortcuts for efficient subgroup identification.
- **Quantum Complexity:** Quantum algorithms, particularly those utilizing the Quantum Fourier Transform (QFT), falter in non-abelian settings like $G$. The automorphism action of $H_{\text{right}}$ on $H_{\text{left}}$ disrupts the coherence and periodicity necessary for QFT-based techniques to identify subgroup structures efficiently. Consequently, these quantum approaches do not yield a polynomial-time solution for NAHSP in such non-abelian groups.

# Conclusion

Recovering the hidden subgroup $N = H_{\text{left}}$ in the group $G = H_{\text{left}} \rtimes H_{\text{right}}$ satisfies the definition of the Non-Abelian Hidden Subgroup Problem (NAHSP). The non-abelian structure of $G$, combined with the automorphism action of $H_{\text{right}}$ on $H_{\text{left}}$, ensures that this problem is computationally infeasible under both classical and quantum adversarial models. Thus, the cryptographic hardness of the NAHSP is directly inherited by the problem of recovering $N$ in $G$. □

## 6.5 Reduction

**Adversarial Setup.** Let $\mathcal{A}$ be an adversary attempting to recover the hidden subgroup $N = H_{\text{left}}$ from the group $G = H_{\text{left}} \rtimes H_{\text{right}}$. The adversary interacts with an oracle function $f : G \to S$, where $S$) is the set of left cosets of $N$ in $G$. The function $f$ is defined as:

$$f(g) = gN,$$

where $gN$ is the coset of $N$ containing $g$. The function $f$ satisfies the equivalence relation:

$$f(g) = f(g') \iff gN = g'N.$$

The adversary's goal is to identify $N$ given oracle access to $f$.

**Definition of Security.** The adversary's advantage $\text{Adv}_{\mathcal{A}}$ in recovering $N$ is defined as:

$$\text{Adv}_{\mathcal{A}} = \Pr[\mathcal{A}(f) = N] - \Pr[\mathcal{A}_{\text{random}}(f) = N],$$

where $\mathcal{A}_{\text{random}}$ is a baseline adversary that outputs a random subgroup $N'$ chosen uniformly at random from the set of all possible subgroups of $G$. The probabilities are taken over the random choice of $N$ and any randomness inherent in the adversaries.

**Reduction to the Non-Abelian Hidden Subgroup Problem.** Assume $\mathcal{A}$ is an adversary that can recover the hidden subgroup $N = H_{\text{left}}$ with advantage $\epsilon$. We construct a reduction $\mathcal{R}$ that uses $\mathcal{A}$ to solve the Non-Abelian Hidden Subgroup Problem (NAHSP) as follows:

1. **Input to $\mathcal{R}$:** The group $G = H_{\text{left}} \rtimes H_{\text{right}}$ and oracle function $f : G \to S$ defined by $f(g) = gN$, where $N = H_{\text{left}}$.
2. **Reduction Steps:**
   (a) $\mathcal{R}$ provides $\mathcal{A}$ with oracle access to $f$.
   (b) $\mathcal{A}$ outputs a candidate subgroup $N'$.
   (c) $\mathcal{R}$ verifies whether $N'$ is a valid hidden subgroup by checking:
   $$\forall g, g' \in G, \quad g^{-1}g' \in N' \iff f(g) = f(g').$$
   This ensures that $N'$ correctly defines the coset structure as per the oracle $f$.
3. **Output of $\mathcal{R}$:** If verification succeeds, $\mathcal{R}$ outputs $N'$ as the solution to NAHSP. Otherwise, $\mathcal{R}$ outputs failure.

**Analysis of the Reduction.**

- **Correctness:** If $\mathcal{A}$ successfully identifies $N$, then $\mathcal{R}$ correctly solves NAHSP by outputting $N' = N$. The verification step ensures that $N'$ uniquely satisfies the coset equivalence relation defined by $f$, thereby guaranteeing the correctness of the solution.

- **Efficiency:** The reduction $\mathcal{R}$ invokes $\mathcal{A}$ once and performs polynomial-time group operations for verification. Therefore, the computational overhead of $\mathcal{R}$ is polynomial in the size of $G$ and bounded by the runtime of $\mathcal{A}$.
- **Adversarial Advantage:** Suppose $\mathcal{A}$ has a non-negligible advantage $\epsilon$ in recovering $N$. Then, $\mathcal{R}$ achieves the same advantage in solving NAHSP:

$$\mathrm{Adv}_{\mathcal{R}} = \mathrm{Adv}_{\mathcal{A}} = \epsilon.$$

  This implies that any adversary $\mathcal{A}$ capable of recovering $N$ with advantage $\epsilon$ enables $\mathcal{R}$ to solve NAHSP with the same advantage.

**Hardness of Subgroup Recovery.**

- **Classical Adversaries:** Classical algorithms would need to enumerate cosets of $N$, which is computationally infeasible due to the exponential size of $G$. Additionally, the non-abelian structure of $G$ lacks the necessary algebraic properties that allow for efficient subgroup identification, preventing the use of techniques such as brute-force search or combinatorial optimizations.
- **Quantum Adversaries:** Quantum algorithms, including those leveraging the Quantum Fourier Transform (QFT), struggle with $G$'s non-abelian structure. The automorphism action of $H_{\text{right}}$ on $H_{\text{left}}$ disrupts the periodicity and coherence essential for QFT-based subgroup recovery. As a result, these quantum techniques fail to efficiently exploit the hidden subgroup structure in $G$, ensuring resistance against known quantum attacks.

**Conclusion.** This reduction demonstrates that recovering the hidden subgroup $N = H_{\text{left}}$ in $G = H_{\text{left}} \rtimes H_{\text{right}}$ is at least as hard as solving the Non-Abelian Hidden Subgroup Problem (NAHSP). The intractability of NAHSP under both classical and quantum adversarial models ensures the cryptographic security of the proposed system. □

# 7 Equivocation and Indistinguishability from $px(\cdot)$

## 7.1 Equivocation Function $px(\cdot)$

The equivocation function $px(\cdot)$ is a deterministic mapping that compresses an input element into an equivalence class represented by the output. It is a lossy, surjective, many-to-one compression function that reduces real entropy while maintaining high observed entropy in the output. The mutual entropy between the input and output is distributed across indistinguishable equivalence classes, ensuring computational impracticality in enumerating all potential valid inputs from a given output. While the correct input is guaranteed to exist within the equivalence class, no heuristic information can differentiate it from other valid pre-images. Note that we use level III parameters for this section in general.

The $px(\cdot)$ function operates as follows:

**1. Inverse NTT Transform:** The input element, initially represented in the NTT domain over a field $q = 257$, is transformed back to the input domain via the appropriate inverse NTT, based on $n$.

**2. Forward NTT Transform to $q = 1283$:** A forward NTT is performed using parameters $q = 1283, \omega = 3$. Note that only addition is performed using this field, so using $\omega = 3$ only impacts computational performance. This step expands the coefficient range by approximately a factor of 5, mapping them to values between 0 and 1282.

**3. Scaling and Ambiguity Introduction:** The element is added to itself element-wise and reduced modulo 1283. This scaling operation is repeated four times, introducing additional ambiguity at each step. For each coefficient, there are two possible pre-image states—either it rolled over or it did not. This process creates an internal diffusion factor of $2^{4n}$.

**4. Uniform Input Distribution:** The input element is derived from:

- A uniformly random, full-rank hidden matrix $U$,
- A uniquely randomized, full-rank public matrix $B$,
- A uniformly random secret element $x$.

These components ensure the input uniformly spans the ambient modular space $\mathbb{Z}_q^n$, directly supporting the uniformity produced by the diffusion factor $2^{4n}$. **5. Inverse and Forward NTT Transforms:** The diffused element undergoes:

- An inverse NTT transform with parameters $q = 1283, \omega = 3$,
- A forward NTT transform with parameters $q = 257, \omega = \{81, 9, 3\}$, for on $n = 64, n = 128, n = 256$ respectively.

This step maps the element back to a smaller field while preserving ambiguity.

Each coefficient in the $q = 1283$ field has approximately 5 pre-image coefficients in the $q = 257$ field. With $n$ coefficients, the total number of pre-images is:

$$5^n.$$

For $n = 128$, this results in $2^{297}$ pre-images distributed across $2^{727}$ equivalence classes (assuming uniform partitioning). Each equivalence class contains $2^{297}$ elements, making it computationally infeasible for an adversary to enumerate all valid pre-images for a given output. The core of our information-theoretic security lies in the impracticality of inverting $px(\cdot)$. This security is based on the following principles:

### Information-Theoretic Security

The core of our information-theoretic security lies in the impracticality of inverting $px(\cdot)$. This security is based on the following principles:

- Diffusion and Avalanche Effect: The diffusion factor $2^{4n}$ ensures that small changes in the input lead to significant and widespread changes in the output, making it difficult to trace back to the original input.
- Pre-Image Resistance: Mapping to approximately $5^n$ pre-images distributed across $2^{727}$ equivalence classes for $n = 128$ ensures that each output corresponds to a large and computationally infeasible set of inputs.

- NTT Transformation Security: The use of NTTs with carefully chosen parameters introduces additional complexity, leveraging the hardness of problems related to discrete transforms over finite fields.

- Assumptions on Computational Resources: The security guarantees assume that adversaries do not possess exponential computational resources to perform exhaustive searches within the equivalence classes.

In summary, while the $px(\cdot)$ function allows enumeration of all pre-images theoretically, the computational and combinatorial requirements make this infeasible in practice. This foundational design ensures the cryptographic strength of the equivocation function in supporting secure operations.

## 7.2 Analysis of $px()$

This section rigorously establishes the security properties of the mapping function $px(\cdot)$, a core cryptographic primitive in this system. Through formal proofs, we demonstrate its resilience against adversarial attacks, its statistical uniformity, highlighting its robustness in both classical and quantum computational models.

It should be noted that selecting specific parameters for our $px()$ function requires consideration of modular overlap. The foundation of our function is based on additive diffusion of projected elements from $q_0 = 257$ to a larger prime field, such as $q_1 = 1283$. As it is mathematically impossible for two prime numbers to divide cleanly, we seek to get as close as possible. For example, a $q = 1285 == 5 * 257$, leaving no modular remainder during mapping, but 1285 is not a prime number. Our choice of $q_1 = 1283$ in this case leaves us 1.17% overlapping values during inversion. Restated, out of 257 elements 254 will have 5 possible pre-image coefficients that map back, with the remainder having 4. The implication being that in reality, not all equivalence classes hold exactly the same number of pre-images and there is a slight deviation related to the 1.17% modular overlap. For simplicity, as the majority in this case have 5 valid pre-image coefficients, we will use 5, vs a more accurate $\approx 4.99 - 5.01$ value.

## 7.3 Probability of Random Forgery

The information-theoretic barrier we create is not infinite, but it presents an intractably large solution space for an adversary, as we will show below.

Table 2: Preimage and Equivalence Class Analysis for $px()$

| **Dimension ($n$)** | **Ambient Space ($2^n$)** | **Preimages ($2^b$)** | **Equivalence Classes ($2^e$)** |
|---|---|---|---|
| 64 | $2^{512}$ | $2^{149}$ | $2^{363}$ |
| 128 | $2^{1024}$ | $2^{297}$ | $2^{727}$ |
| 256 | $2^{2048}$ | $2^{640}$ | $2^{1408}$ |

These numbers yield the following probability ratios:

$$\begin{aligned}
\text{For } n = 64: &\quad 1.844674407370955 \times 10^{-45},\\
\text{For } n = 128: &\quad 3.402823669209385 \times 10^{-90},\\
\text{For } n = 256: &\quad 1.157920892373162 \times 10^{-179}.
\end{aligned}$$

This represents the probability for each $n$, that a randomly selected pre-image will map to a selected equivalence class. These numbers are useful for proving probability of random signature forgery, but to put them into context we will relate them to winning the PowerBall lottery which has a probability of 1 in 292,201,338. The only constraint is that you can only buy one ticket per jackpot.

| **n** | Probability | Powerball Wins | Probability (Chaining) | Powerball Wins (Chaining) |
|---|---|---|---|---|
| 64 | $1.84 \times 10^{-45}$ | 13 | $1.34 \times 10^{-358}$ | 350 |
| 128 | $3.40 \times 10^{-90}$ | 26 | $1.55 \times 10^{-537}$ | 529 |
| 256 | $1.15 \times 10^{-180}$ | 53 | $1.80 \times 10^{-720}$ | 711 |

Assuming these calculations are correct, without chaining the odds of guessing a valid signature for $n = 64$ are $\approx$ the same as winning PowerBall 13 times in a row. As level I has $n = 64$ and 8 chained instances, the success probability, based on buying one ticket per jackpot, jumps to winning PowerBall 350 times in a row. Guessing a valid signature at level V where $n = 256$ with 4 chained instances is as likely as willing 711 consecutive jackpots. As this is technically not impossible, similar to how the proposed scheme doesn't guarantee perfect secrecy, it could in theory happen. It's just unlikely.

### 7.3.1 Brute Force Storage Requirements

In the context of attempting to mount a brute force attack, we consider the minimum storage requirements to guarantee a successful outcome. Each element requires $\{64, 128, 256\}$ bytes of storage and each observed output $t''$ maps to a large number of valid preimages. There also exists a lower bound of observations required to guarantee enough correct $t'$ outputs such that the correct set of equations that can be defined and solved to recover both $U$ and $x$. The storage required for each security level is listed below, in brontobytes:

Table 3: Preimage Storage and Observation Scaling for $px()$

| $n$ | **Preimages ($2^b$)** | **1 Observation (BB)** | **Min Observations (BB)** |
|---|---|---|---|
| 64 | $2^{149}$ | $2^{65}$ | $2^{72}$ |
| 128 | $2^{297}$ | $2^{214}$ | $2^{221}$ |
| 256 | $2^{640}$ | $2^{648}$ | $2^{656}$ |

### Brontobytes and Exabytes: Units for Massive Data Volumes

The **brontobyte** (BB) is a theoretical unit of data storage equivalent to:

$$1\,\text{BB} = 2^{90}\,\text{bytes} = 1{,}024\,\text{yottabytes}.$$

It represents a continuation of the binary progression of data storage units. To provide perspective:

- $1\,\text{BB} = 1{,}024\,\text{YB}$ (yottabytes),
- $1\,\text{YB} = 1{,}024\,\text{ZB}$ (zettabytes),
- $1\,\text{ZB} = 1{,}024\,\text{EB}$ (exabytes).

By comparison:

- $1\,\text{EB} = 2^{60}$ bytes, or roughly 1 billion gigabytes.
- A single brontobyte ($2^{90}$) is $1,099,511,627,776\,\text{EB}$, far exceeding the total amount of data stored globally.

**Relevance to Information-Theoretic Security**

In this context, brontobyte-scale data is required for analyzing the feasibility of brute-force attacks and quantifying the strength of information-theoretic guarantees. For example:

- Storing all preimages for even a single observation exceeds brontobyte (BB) levels with increasing dimensionality.
- As of 2024, the global installed base of data storage capacity is projected to be approximately 11.23 zettabytes (ZB), equivalent to 11.23 trillion terabytes (TB) or 0.00001071 brontobytes (BB).
- For minimum required observations ($n = 64$, 65 observations), this storage requirement guarantees an adversary faces infeasible data handling and computational costs.
- Brute force storage requirements are infeasible, and the probability of simply guessing a valid answer is negligible, exactly what one would expect from practical information theoretic security guarantees.

## 7.4 Equivalence Class Invariance of $px(\cdot)$

**Lemma 1** (Equivalence Class Invariance of $px(\cdot)$)**.** *The mapping function $px : G \to \mathcal{Y}$ partitions the group $G$ into equivalence classes defined by the subgroup $N$. Specifically:*

$$px(g_1) = px(g_2) \iff g_1 \sim g_2 \quad \forall g_1, g_2 \in G,$$

*where $\sim$ is the equivalence relation:*

$$g_1 \sim g_2 \iff g_1 \cdot n = g_2 \text{ for some } n \in N.$$

*Proof.* **Objective:** To show that $px(\cdot)$ is invariant over equivalence classes induced by $N$.

**Definition of Equivalence Classes:** The equivalence class of an element $g \in G$ with respect to $N$ is:

$$[g]_N = \{g \cdot n \mid n \in N\}.$$

These classes partition $G$ into disjoint subsets such that:

$$G = \bigcup_i [g_i]_N, \quad [g_i]_N \cap [g_j]_N = \emptyset \text{ for } i \neq j.$$

**Invariance of $px(\cdot)$:** The mapping function $px(\cdot)$ satisfies the following invariance property:

$$px(g \cdot n) = px(g) \quad \forall g \in G,\, n \in N.$$

This property implies that $px(\cdot)$ maps all elements of an equivalence class $[g]_N$ to the same value in $\mathcal{Y}$.

**Proof of Equivalence:** To prove $px(g_1) = px(g_2) \iff g_1 \sim g_2$, we consider both directions:

- **Forward Direction ($\Rightarrow$):** Assume $px(g_1) = px(g_2)$. By the invariance property of $px(\cdot)$, this implies:

  $$px(g_1 \cdot n_1) = px(g_2 \cdot n_2) \quad \text{for some } n_1, n_2 \in N.$$

  Since $px(\cdot)$ is consistent across equivalence classes, it follows that:

  $$g_1 \cdot n_1 = g_2 \cdot n_2 \quad \text{for some } n_1, n_2 \in N,$$

  which implies $g_1 \sim g_2$.

- **Backward Direction ($\Leftarrow$)**: Assume $g_1 \sim g_2$, i.e., $g_1 \cdot n = g_2$ for some $n \in N$. By the invariance property of $px(\cdot)$:

  $$px(g_2) = px(g_1 \cdot n) = px(g_1).$$

  Therefore, $px(g_1) = px(g_2)$.

**Conclusion:** The function $px(\cdot)$ is invariant across equivalence classes induced by $N$. This ensures that elements within the same equivalence class are indistinguishable under $px(\cdot)$. □

## 7.5 Enhanced Hiding of $px(\cdot)$

**Lemma 2** (Ambiguity and Obfuscation in $px(\cdot)$)**.** *The mapping function $px(\cdot)$ is a **many-to-one** function that increases the entropy of its outputs and introduces obfuscation through the hidden matrix $U$. Specifically, $px(\cdot)$ creates an ambiguous search space where recovering the original inputs is computationally infeasible.*

*Proof.* **Objective:** To show that $px(\cdot)$ creates ambiguity by mapping multiple distinct inputs to the same output and obfuscates structural relationships through the hidden matrix $U$.

**Many-to-One Mapping:** The mapping $px(\cdot)$ compresses the input space, assigning multiple distinct inputs to the same output:

$$|px^{-1}(y)| \geq 2^k, \quad \text{where } k \text{ depends on system parameters.}$$

This many-to-one nature ensures that adversaries cannot uniquely identify an input from an output.

**Obfuscation via $U$:** The hidden matrix $U$ transforms the input as:

$$t' = U \cdot t \mod q.$$

Without knowledge of $U$, inverting this transformation is computationally infeasible. The randomness of $U$ ensures that no exploitable dependencies exist between the input $t$ and its transformation $t'$.

**Combined Effect:** The combined properties of $px(\cdot)$ and $U$ ensure:

- **Ambiguity:** Each output corresponds to a large equivalence class of indistinguishable inputs, creating an inflated search space.
- **Obfuscation:** Structural relationships between inputs are disrupted, preventing adversaries from reconstructing $t$ without explicit knowledge of $U$.

**Security Implications:** Recovering the original input $t$ for a given output $px(t)$ requires exhaustive search through all possible pre-images. The exponential size of the search space and the disruption of algebraic structures ensure that this task is computationally infeasible.

**Conclusion:** The mapping $px(\cdot)$, combined with the obfuscation introduced by $U$, creates a high-entropy, ambiguous output space. These properties ensure the security of the mapping against adversarial recovery of original inputs. □

## 7.6 Indistinguishability and Statistical Uniformity of $px(\cdot)$ Outputs

Note that we use Level III parameters for this section, $n = 128, q = 257$.

**Lemma 3** (Indistinguishability of $px(\cdot)$ Outputs)**.** *The outputs of the mapping function $px(\cdot)\colon \mathbb{Z}_{257}^{128} \to \mathbb{Z}_{257}^{128}$ are computationally indistinguishable from uniformly random vectors in $\mathbb{Z}_{257}^{128}$, given no access to the secret input $x$, intermediate values $t'$, or the hidden transformation parameters.*

*Proof.* **Objective:** To prove that the outputs $px(t')$ are computationally indistinguishable from uniformly random vectors, assuming adversaries lack access to the secret input $x$ or intermediate values.

**Step 1: High Entropy and Uniform Mixing** The mapping $px(\cdot)$ introduces high entropy and uniform mixing through sequential transformations:

- **Initial Transformation (NTT):** The input vector $t$ is transformed into $t' = \mathrm{NTT}_{1283}(t)$, dispersing coefficients of $t$ across the frequency domain. This ensures pseudorandom spreading of $t'$ over $\mathbb{Z}_{1283}^{128}$.
- **Additive Mixing:** The operation $t'' = t' + 2 \cdot t'$ introduces further uniformity, erasing any residual structure in $t'$.
- **Inverse Transformation:** The inverse NTT, $z = \mathrm{INV\_NTT}_{1283}(t'')$, preserves high entropy and disperses any remaining correlations across coefficients in $\mathbb{Z}_{1283}^{128}$.
- **Final Projection:** The mapping $px(t') = \mathrm{NTT}_{257}(z)$ reduces $z$ modulo 257, ensuring outputs are uniformly distributed in $\mathbb{Z}_{257}^{128}$ and removing residual patterns.

**Step 2: Compression and Ambiguity** The function $px(\cdot)$ compresses $\mathbb{Z}_{1283}^{128}$ into $\mathbb{Z}_{257}^{128}$, introducing a many-to-one mapping. Each output $y \in \mathbb{Z}_{257}^{128}$ corresponds to approximately $2^{297}$ indistinguishable pre-images (for level III):

$$|px^{-1}(y)| \approx 2^{297}.$$

This compression ensures that adversaries observing $px(t')$ cannot deduce a unique input $t$, significantly inflating the effective search space.

**Step 3: Statistical Uniformity** The modular reductions and transformations in $px(\cdot)$ ensure that outputs pass standard randomness tests:

- **Entropy Preservation:** High entropy at intermediate states $t'$ and $z$ ensures no statistical patterns remain.
- **Empirical Validation:** Statistical tests (e.g., NIST randomness suite) confirm that $px(t')$ outputs are indistinguishable from uniformly random elements in $\mathbb{Z}_{257}^{128}$.

**Step 4: Entropy and Adversarial Uncertainty** The lossy nature of $px(\cdot)$ guarantees high apparent entropy for adversaries:

- **Min-Entropy ($H_\infty$):** Assuming uniform distribution over $\mathbb{Z}_{257}^{128}$, the min-entropy of $px(t')$ is:
$$H_\infty(px(t')) = \log_2(|\mathbb{Z}_{257}^{128}|) = 727\,\text{bits}.$$
- **Conditional Entropy ($H(t' \mid px(t'))$):** Given $px(t')$, the adversary faces residual uncertainty about $t'$:
$$H(t' \mid px(t')) = 1024 - 727 = 297\,\text{bits}.$$
This indicates that each output corresponds to $2^{297}$ indistinguishable pre-images, obfuscating input-output relationships.
- **False Entropy Perception:** From the adversary's perspective, $px(t')$ appears to have full entropy $H(px(t'))$, as outputs are indistinguishable from uniform distributions.

**Step 5: Computational Indistinguishability** For any efficient adversary $\mathcal{A}$, distinguishing $px(t')$ from a uniformly random vector $r \in \mathbb{Z}_{257}^{128}$ is computationally infeasible:

$$|\Pr[\mathcal{A}(px(t')) = 1] - \Pr[\mathcal{A}(r) = 1]| \leq \text{negl}(n),$$

where $\text{negl}(n)$ is a negligible function of the security parameter $n$. The modular reductions and many-to-one compression ensure that adversaries cannot exploit patterns to distinguish $px(t')$ from random vectors.

**Conclusion** The mapping $px(\cdot)$ ensures high entropy, statistical uniformity, and computational indistinguishability. These properties collectively enhance its cryptographic strength, making the outputs indistinguishable from uniformly random vectors and robust against adversarial analysis. $\square$

## 7.7 Collision and Pre-Image Resistance of $px(\cdot)$

**Lemma 4** (Collision and Pre-Image Resistance of $px(\cdot)$)**.** *The mapping function* $px(\cdot)\colon \mathbb{Z}_{257}^{128} \to \mathbb{Z}_{257}^{128}$ *satisfies:*

1. ***Collision Resistance:*** *Finding distinct inputs* $t_1, t_2 \in \mathbb{Z}_{257}^{128}$ *such that* $px(t_1) = px(t_2)$ *is computationally infeasible.*
2. ***Pre-Image Resistance:*** *Given* $y \in \mathbb{Z}_{257}^{128}$*, finding any* $t \in \mathbb{Z}_{257}^{128}$ *such that* $px(t) = y$ *is computationally infeasible.*

*These properties hold under standard cryptographic assumptions.*

*Proof.* **Objective:** To demonstrate that $px(\cdot)$ is resistant to both collision and pre-image attacks by analyzing its structure, randomness, and computational complexity.

**Structure of $px(\cdot)$** The mapping $px(\cdot)$ consists of the following steps:

1. $t' = \mathrm{NTT}_{1283}(t)$, where $t = A \cdot x \mod 257$ and $A$ is a random public matrix.
2. $t'' = t' + 3 \cdot t'$, introducing additive mixing in $\mathbb{Z}_{1283}^{128}$.
3. $z = \mathrm{INV\_NTT}_{1283}(t'')$, returning to the time domain modulo 1283.
4. $px(t) = \mathrm{NTT}_{257}(z)$, projecting the result into $\mathbb{Z}_{257}^{128}$.

These transformations ensure randomness, mixing, and compression, making $px(\cdot)$ resistant to both collision and inversion attempts.

**Collision Resistance Analysis** For $px(t_1) = px(t_2)$ to hold, it must be true that $z_1 = z_2$, since $\mathrm{NTT}_{257}$ is invertible. This implies:

$$\mathrm{INV\_NTT}_{1283}(t_1'') = \mathrm{INV\_NTT}_{1283}(t_2'').$$

Given that:

$$t_1'' = t_1' + t_1', \quad t_2'' = t_2' + t_2',$$

distinct $t_1' \neq t_2'$ result in distinct $t_1'' \neq t_2''$ due to additive mixing. A collision would require:

$$\mathrm{NTT}_{1283}(t_1) = \mathrm{NTT}_{1283}(t_2),$$

which is unlikely given the pseudorandom nature of $A$. The probability of such random collisions is bounded by:

$$P_{\text{collision}} \leq \frac{q^2}{2 \cdot 257^{128}},$$

where $q$ is the number of adversarial queries. Since $257^{128}$ is astronomically large, $px(\cdot)$ is collision-resistant.

**Pre-Image Resistance Analysis** To invert $px(t) = y$, an adversary must reverse multiple transformations:

- Recover $z$ from $y = \text{NTT}_{257}(z)$, which is infeasible without knowledge of $t$ or intermediate states.
- Reverse $z = \text{INV\_NTT}_{1283}(t'')$ to find $t''$, where $t'' = t' + 3 \cdot t'$. Additive mixing obscures linear relationships in $t'$.
- Solve $t = A \cdot x \mod 257$ from $t' = \text{NTT}_{1283}(t)$. Without knowledge of $x$, this is computationally infeasible due to the pseudorandomness of $A$.

Additionally, the lossy nature of $px(\cdot)$ ensures:

$$|px^{-1}(y)| \approx 2^{297},$$

making it computationally infeasible to identify a unique pre-image among $2^{297}$ candidates.

**Conclusion** The structural properties of $px(\cdot)$, including pseudorandom transformations, additive mixing, and modular reductions, ensure resistance to both collision and pre-image attacks. These properties make $px(\cdot)$ secure under standard cryptographic assumptions against both classical and quantum adversaries. □

## 7.8 Avalanche Effect in $px(\cdot)$

**Lemma 5** (Avalanche Effect of $px(\cdot)$)**.** *For any inputs $t_1, t_2 \in \mathbb{Z}_{257}^{128}$ differing by a single bit, the outputs $px(t_1)$ and $px(t_2)$ are computationally indistinguishable from independent, uniformly random vectors in $\mathbb{Z}_{257}^{128}$.*

*Proof.* **Objective:** To show that a small change in $t$ propagates unpredictably through $px(\cdot)$, ensuring significant and uncorrelated differences in the outputs.

**Step 1: Propagation Through NTT Transformations** The input $t$ undergoes the transformation $t' = \text{NTT}_{1283}(t)$. Due to the properties of the NTT:

- Each coefficient of $t'$ depends on all coefficients of $t$.
- A single-bit change in $t$ affects every coefficient of $t'$ due to the frequency-domain dispersion.

This ensures that the effect of a single-bit change is amplified across the intermediate state $t'$.

**Step 2: Additive Mixing** The operation $t'' = t' + 2 \cdot t'$ introduces further mixing:

- The additive mixing operation is performed modulo 1283, ensuring that changes propagate unpredictably due to modular wraparound.
- Any change in $t'$ affects all coefficients of $t''$.

**Step 3: Inverse Transformation** The inverse NTT $z = \text{INV\_NTT}_{1283}(t'')$ maps the mixed state back to the time domain. This operation:

- Preserves the non-linear dependencies introduced by additive mixing.
- Further disperses the effects of the initial change across all coefficients of $z$.

**Step 4: Final Projection** The projection to $\mathbb{Z}_{257}^{128}$ via $\text{NTT}_{257}$ ensures that the output $px(t)$ reflects the amplified changes from earlier stages. Specifically:

- Modular reduction ensures that even small differences in $z$ produce large, unpredictable differences in $px(t)$.
- The NTT modulo 257 further spreads any changes across all coefficients.

**Step 5: Statistical Indistinguishability** For any $t_1, t_2$ differing by a single bit, the outputs $px(t_1)$ and $px(t_2)$ satisfy:

$$\Pr[\mathcal{A}(px(t_1)) = 1] - \Pr[\mathcal{A}(px(t_2)) = 1] \leq \text{negl}(n),$$

where $\mathcal{A}$ is any efficient adversary and $\text{negl}(n)$ is a negligible function of the security parameter $n$.

**Conclusion** The transformations within $px(\cdot)$ amplify any small changes in $t$, ensuring that $px(t_1)$ and $px(t_2)$ are computationally indistinguishable from independent, uniformly random vectors. This establishes the avalanche effect for $px(\cdot)$. $\square$

## 7.9 Adaptive Security of $px(\cdot)$

**Lemma 6** (Adaptive Security of $px(\cdot)$)**.** *For any adversary $\mathcal{A}$ making up to $q$ adaptive queries to $px(\cdot)$, the outputs of $px(\cdot)$ remain indistinguishable from independent, uniformly random vectors in $\mathbb{Z}_{257}^{128}$, given the randomized matrix $A$ is independently regenerated for each operation.*

*Proof.* **Objective:** To prove that $px(\cdot)$ maintains its security properties against adversaries making multiple adaptive queries.

**Step 1: Randomization of $A$** The matrix $A$ is independently regenerated for each invocation of $px(\cdot)$. This ensures that:

- Outputs $px(t)$ from different invocations are uncorrelated.
- An adversary cannot infer patterns or dependencies between outputs from different queries.

**Step 2: Independence of Transformations** Each invocation of $px(\cdot)$ is independent due to the randomized $A$. Specifically:

- The NTT transformations $\text{NTT}_{1283}$ and $\text{NTT}_{257}$ depend on $A$, ensuring fresh randomness for each query.
- Additive mixing and modular reductions are independent for each invocation, further decoupling the outputs.

**Step 3: Indistinguishability Under Adaptive Queries** For any $q$ adaptive queries $t_1, t_2, \ldots, t_q$, the corresponding outputs $px(t_1), px(t_2), \ldots, px(t_q)$ are indistinguishable from independent, uniformly random vectors. Formally:

$$\Delta = |\Pr[\mathcal{A}(px(t_1), \ldots, px(t_q)) = 1] - \Pr[\mathcal{A}(r_1, \ldots, r_q) = 1]| \leq \text{negl}(n),$$

where $r_1, \ldots, r_q$ are independent, uniformly random vectors in $\mathbb{Z}_{257}^{128}$.

**Step 4: Resilience to Query Correlations** Even if $\mathcal{A}$ chooses $t_1, t_2, \ldots, t_q$ adaptively, the randomized $A$ ensures that:

- Outputs $px(t_i)$ are uncorrelated.
- Knowledge of $px(t_i)$ does not provide any advantage in predicting $px(t_{i+1})$.

**Conclusion** The independence of $A$ across queries ensures that $px(\cdot)$ is secure against adaptive adversaries, maintaining indistinguishability and unpredictability under multiple queries. □

## 7.10 Adversarial Complexity and Relation to NAHSP

**Lemma 7** (Adversarial Complexity of Pre-Image Recovery)**.** *Recovering the valid pre-image of $px(t')$ requires brute-forcing all $2^{297}$ indistinguishable pre-images and testing each against the cryptographic construction. This task is computationally infeasible under both classical and quantum adversarial models, as it reduces to solving a combinatorial subgroup recovery problem tied to NAHSP.*

*Proof.* **Objective:** To show that recovering the valid pre-image of $px(t')$ is computationally infeasible due to the obfuscation introduced by $px(\cdot)$ and its connection to the Non-Abelian Hidden Subgroup Problem (NAHSP).

**Step 1: Compression and Pre-Image Ambiguity** The mapping function $px(\cdot)$ transforms inputs $t \in \mathbb{Z}_{1283}^{128}$ to outputs $px(t') \in \mathbb{Z}_{257}^{128}$, with a compression ratio of approximately $2^{297}$-to-1:

$$R_{\text{compression}} = \frac{|\mathbb{Z}_{1283}^{128}|}{|\mathbb{Z}_{257}^{128}|} = 2^{297}.$$

For a given output $px(t') = y$, the adversary faces approximately $2^{297}$ indistinguishable pre-images $t_1, t_2, \ldots, t_{2^{297}}$. Among these, only one pre-image corresponds to the correct subgroup $N$.

**Step 2: Valid Pre-Image and Subgroup Recovery** The valid pre-image satisfies the transformation:

$$t' = U \cdot t \mod q, \quad t = A \cdot x \mod q,$$

where:

- $A \in \mathbb{Z}_{257}^{128\times128}$ is a public, randomized, full-rank matrix.
- $U \in \mathbb{Z}_{257}^{128\times128}$ is a secret, dense, full-rank matrix defining the subgroup $N$.

Recovering this valid pre-image is equivalent to solving the subgroup recovery problem for $N$ in $G = H_{\text{left}} \rtimes H_{\text{right}}$, where $G$ is the semidirect product of $H_{\text{left}} = \langle U \rangle$ and $H_{\text{right}} = \langle A \rangle$.

**Step 3: Combinatorial Search Space** Without knowledge of $x$ or $U$, the adversary must:

- Enumerate all $2^{297}$ indistinguishable pre-images $t_i$ for the given output $px(t') = y$.
- Test each pre-image against the cryptographic construction to determine whether it satisfies the subgroup structure defined by $A$ and $U$.

This brute-force search involves solving a system of obfuscated equations for each candidate $t_i$, including:

- Modular reductions in $\mathbb{Z}_{1283}$ and $\mathbb{Z}_{257}$,
- Non-linear dependencies introduced by NTT transformations and additive mixing.

The total complexity scales as:

$$O(2^{297}),$$

since each pre-image requires testing against the subgroup structure.

**Step 4: Reduction to NAHSP** The task of identifying the valid pre-image reduces to solving the Non-Abelian Hidden Subgroup Problem (NAHSP) for $G$:

- The hidden subgroup $N$ is defined by $H_{\text{left}} = \langle U \rangle$.
- The cosets of $N$ in $G$ correspond to equivalence classes of inputs under $px(\cdot)$.

Solving the NAHSP involves identifying the subgroup $N$ from its coset structure, which is computationally hard for non-abelian groups like $G$. The obfuscation introduced by $px(\cdot)$ ensures that:

$$\Pr[\text{Adversary recovers } N] \leq \frac{1}{|px^{-1}(y)|} = \frac{1}{2^{297}}.$$

**Step 5: Resistance to Quantum Speedup** Quantum algorithms like Grover's provide no advantage because:

- The search space is structured around the subgroup recovery problem for $N$, which involves combinatorial dependencies between pre-images.
- NAHSP inherently disrupts the coherence and periodicity necessary for quantum algorithms to achieve efficient speedups.
- The adversary must brute-force permutations of obfuscated equations, which cannot be accelerated by Grover's algorithm.

**Step 6: Formal Complexity Analysis** The total complexity of recovering the valid pre-image can be summarized as:

- **Classical Complexity:** $O(2^{297})$, due to the need to brute-force all indistinguishable pre-images.
- **Quantum Complexity:** $O(2^{297})$, as quantum algorithms provide no advantage for structured subgroup recovery problems.

**Conclusion** Recovering the valid pre-image of $px(t')$ reduces to solving the NAHSP for $G = H_{\text{left}} \rtimes H_{\text{right}}$. The compression introduced by $px(\cdot)$, combined with the obfuscation from modular reductions, NTTs, and subgroup structures, ensures that this task is computationally infeasible for both classical and quantum adversaries. □

### 7.11 Quantum Resistance of $px(\cdot)$

The mapping $px(\cdot)$ achieves quantum resistance by introducing high entropy, compression, and structural obfuscation, effectively neutralizing known quantum algorithmic advantages. Key disruptions include:

- **Quantum Fourier Transform (QFT):**
  - $px(\cdot)$ collapses cosets of $G$ into indistinguishable equivalence classes, removing the periodic eigenstate structures required for QFT-based solvers.
  - The non-abelian properties of $G$ disrupt coherence and prevent the exploitation of group symmetries, negating QFT efficiency.
- **Grover's Search:**
  - Compression and indistinguishability inflate the effective search space, counteracting Grover's quadratic speedup by increasing the adversary's uncertainty over $2^{297}$ indistinguishable pre-images.
  - Structural dependencies introduced by $px(\cdot)$ further impede the isolation of marked states necessary for Grover's algorithm.
- **Error Amplification and Post-Processing Complexity:**
  - Outputs of $px(\cdot)$ exhibit exponential entropy, requiring $O(2^{297})$ operations to correlate cosets with subgroup elements.
  - Modular reductions and non-linear transformations propagate noise in quantum superpositions, amplifying errors and degrading adversarial coherence.
- **Theoretical Structural Attacks:**
  - While this form of attack is hypothetical, we anticipate a variety of advances in topological quantum computing, enabling the next generation of quantum algorithms based on braids, toroids, hypercubes, and other topological structures to exploit periodicity in ways we have yet to consider.
  - The uniformly unstructured nature of this construction, combined with constant randomization of half of the matrix group makes this form of attack less likely to succeed in the future.

By obfuscating structural relationships and enforcing exponential search complexity, $px(\cdot)$ ensures that recovering the hidden subgroup $N$ remains computationally infeasible under both classical and quantum adversarial models. These properties align quantum complexity with classical bounds, establishing $px(\cdot)$ as a robust cryptographic primitive.

# 8 Information-Theoretic Security of $px(t')$

**Theorem.** Let $px : G \to Y$ be a mapping function with equivalence classes of size $|px^{-1}(y)| \geq 2^k$ for all $y \in Y$. Then:

1. The adversary's mutual information $I(t'; px(t'))$ is negligible, bounded by $\varepsilon$, where $\varepsilon$ is a function of the compression ratio $|G|/|Y|$.

2. The adversary's probability of recovering $t'$ from $px(t')$ is negligible, bounded by $\frac{1}{|px^{-1}(px(t'))|}$.

**Proof.**

*Definitions and Setup.* Let $t'$ represent the hidden group elements, and $Y = px(t')$ the observed outputs. The mapping $px$ compresses $G$ into $Y$, such that each $y \in Y$ corresponds to an equivalence class of size $|px^{-1}(y)|$.

*Mutual Information Bound.* Mutual information is defined as:

$$I(t'; Y) = H(t') - H(t' \mid Y),$$

where:

- $H(t') = \log_2(|G|)$, the entropy of $t'$,
- $H(t' \mid Y) = \log_2(|px^{-1}(y)|)$, the conditional entropy of $t'$ given $Y$.

Substituting, we have:

$$I(t'; Y) = \log_2(|G|) - \log_2(|px^{-1}(y)|).$$

Rewriting in terms of the compression ratio $|G|/|Y|$, the leakage is:

$$I(t'; Y) \leq \log_2\left(\frac{|G|}{|Y|}\right).$$

To ensure negligible leakage, the compression ratio $|G|/|Y|$ must satisfy:

$$\log_2\left(\frac{|G|}{|Y|}\right) \leq \varepsilon,$$

where $\varepsilon$ is a negligible function of the security parameter $n$.

*Adversarial Success Probability.* The adversary's probability of recovering $t'$ given $Y$ is:

$$\Pr[\text{Recover } t'] = \frac{1}{|px^{-1}(px(t'))|}.$$

Since $|px^{-1}(px(t'))| \geq 2^k$, this probability is:

$$\Pr[\text{Recover } t'] \leq 2^{-k}.$$

For sufficiently large $k$, this probability is negligible:

$$\Pr[\text{Recover } t'] \leq \text{negl}(n).$$

*Subgroup Recovery.* Recovering the hidden subgroup $N$ requires solving the Non-Abelian Hidden Subgroup Problem (NAHSP). The adversary cannot distinguish elements in $px(t')$ without solving NAHSP, ensuring that $N$ remains hidden.

*Conclusion.*

1. The mutual information $I(t'; px(t'))$ is bounded by $\log_2(|G|/|Y|)$, which can be made negligible by choosing sufficiently large parameters $|G|$ and $|px^{-1}(y)|$.
2. The adversary's probability of recovering $t'$ or $N$ is negligible, ensuring that the system achieves practical information-theoretic security.

□

# 9 Proof of Consistency as Verification Under Homomorphic Transformations

**Lemma 8.** *The verification equation $LHS' = RHS'$ holds if and only if the signature $\sigma$ is generated using the corresponding private keys and the specified public key $pk \| fs$, with high probability.*

*Proof.* Let the key generation, signing, and verification functions be defined as follows:

**1. Key Generation**

$$t = A \cdot x \mod q, \quad t' = U \cdot t \mod q, \quad pk = px(t') \mod q,$$

where:

- $A$ is a public matrix,
- $x$ is the secret key,
- $U$ is a private matrix,
- $px$ is the mapping function.

**2. Signature Generation**

$$\sigma = px\big(U \cdot (J(C1) \circ t)\big) \mod q,$$

where:

- $J$ is a hash function (e.g., SHAKE),
- $C1$ is constraint1, derived from $pk \cdot B$ intermediates after cubing and hashing.
- $\circ$ denotes a Hadamard product.

**3. Verification Function**

$$\begin{aligned}
\text{LHS} &= B \cdot (pk \circ J(C2) \mod q, \\
\text{LHS}' &= px\big(\text{LHS} \circ J(C1)\big) \mod q, \\
\text{RHS} &= A \cdot (\sigma \circ J(C2)) \mod q, \\
\text{RHS}' &= px(\text{RHS}) \mod q.
\end{aligned}$$

**Step 1: Valid Signature Consistency**

- Substitute the signature generation equation into RHS:

$$\text{RHS} = A \cdot \big(px(U \cdot (J(C1) \circ t)) \circ J(C2)\big) \mod q.$$

- Using the properties of $px$ and $t' = U \cdot t$, it follows that:

$$px\big(U \cdot (J(C1) \circ t)\big) = px(t') \mod q,$$

where $t'$ satisfies the public key equation $pk = px(t') \mod q$. - Therefore, the transformations applied during signing and verification align, yielding:

$$\text{RHS}' = px(\text{RHS}) = pk \mod q.$$

**Step 2: Validating LHS$'$**

- Substitute $pk$ into LHS:

$$\text{LHS} = B \cdot (pk \circ J(C2) \mod q.$$

- Apply the transformation $px$:

$$\text{LHS}' = px(\text{LHS} \circ J(C1) \mod q.$$

- Since the signature $\sigma$ was generated using the correct private key, the transformations $J(C2)$ compensate for modular inconsistencies, ensuring:

$$\text{LHS}' = pk \mod q.$$

**Step 3: Equivalence of LHS$'$ and RHS$'$**

- Both LHS$'$ and RHS$'$ reduce to $pk \mod q$, implying:

$$\text{LHS}' = \text{RHS}' \iff \sigma \text{ was generated using the correct private key.}$$

**Step 4: Probabilistic Argument for Invalid Signature**

- For an invalid $\sigma$, the transformations in LHS and RHS will not align. To quantify this: - The output of $J(pk\|fs)$ is uniformly distributed over its range. - Each $\sigma$ candidate not generated with the correct private key maps to a random equivalence class under $px$, with negligible probability of aligning with LHS. - The adversary must guess both:

- $\sigma$, which depends on the secret key $x$ and the private matrix $U$,
- The hash $J(pk\|fs)$, which is computationally infeasible due to the pre-image resistance of $J$.

- The success probability of forging $\sigma$ without knowledge of $x$ is bounded by:

$$P_{\text{success}} \leq \frac{1}{q^n},$$

where $q^n$ is the size of the search space for $\sigma$. This represents an information-theoretic lower bound on the success probability.

**Step 5: Contrapositive**

- For an invalid $\sigma$, the mismatch between $\mathrm{LHS}'$ and $\mathrm{RHS}'$ occurs due to inconsistencies in equivalence class mapping, leading to:

$$\mathrm{LHS}' \neq \mathrm{RHS}'.$$

**Conclusion**

The verification equation $\mathrm{LHS}' = \mathrm{RHS}'$ holds if and only if the signature $\sigma$ is generated using the valid private key $x$, the private matrix $U$, and the specified public key and basis $B$ constraint $C1$. The probabilistic argument establishes that forging a valid $\sigma$ without knowledge of the private key is computationally infeasible with high probability. $\square$

# 10 Implementation Details

## 10.1 Matrix Generation Using Diverse Cryptographically Secure PRNGs

To ensure cryptographic security and reproducibility, the public and private matrices in our construction should be generated deterministically using distinct cryptographically secure pseudorandom number generators (CSPRNGs). These are recommendations for high security, and certain implementations may prefer alternate functions.

### 10.1.1 Public Matrix Generation

The public matrices $A$ used to generate the subgroup $H_{\text{right}}$ are derived using AES-DRBG, per NIST-approved DRBG specifications. Each matrix $A \in \mathbb{Z}_q^{n \times n}$ is constructed as follows:

1. Input: A 256-bit public seed $\mathrm{Seed}_A$, which may be application-specific or predefined.
2. Generation: Use AES-DRBG in CTR mode to generate $n^2$ entries.
3. Mapping: Map each entry modulo $q$ to produce a dense, full-rank matrix $A$.
4. Validation: Optionally verify $A$'s rank to ensure it is full rank.

This deterministic process is efficient, ensures reproducibility, and eliminates reliance on weak randomness.

### 10.1.2 Private Matrix Generation

The private matrices $U$, which define the subgroup $H_{\text{left}}$, are generated using SHA-512, SHA3-512, or SHAKE-256:

1. Pre-Input: Optionally use a private 256-bit (or larger) value to key the hash function.
2. Input: A 256-bit private seed $\mathrm{Seed\_}U$, derived from an entropy source or securely exchanged during key generation.

3. Hashing: Apply the chosen hash function to Seed_$U$ to produce $n^2$ pseudorandom outputs.
4. Mapping: Map these outputs modulo $q$ to construct $U$, ensuring full rank and density.
5. Validation: Optionally verify $U$'s rank to confirm full rank.

### 10.1.3 Security Implications

Using AES-DRBG for public matrices and SHA-512/SHA3/SHAKE for private matrices ensures high entropy, cryptographic security, compliance with NIST standards, and diversity in matrix generation. These methods eliminate correlations between $A$ and $U$, ensuring the subgroup structures $H_{\text{right}}$ and $H_{\text{left}}$ align with the theoretical reductions to NAHSP. Deterministic generation guarantees that the matrices are free from vulnerabilities introduced by weak or biased randomness. Furthermore, ensuring full rank for both matrices preserves the cryptographic strength of the construction.

## 10.2 Algorithm Details

## 10.3 Utility Algorithms

Note that we don't specify the specific pseudo-random algorithm used to expand the *seed* value, as this function is designed to be modular. In our reference instance we use AES256-DRBG, but other PRNG constructions are certainly supported.

**Algorithm 1 Sample**(*seed*)

**Generates a matrix for the with non-zero elements.**

**Require:** Dimensions $K, N$, prime modulus $Q1$, root $R1$, and seed `seed`.

```
Initialize A[K][N][N] as an empty matrix.
for mat = 0 to K − 1 do
    rows_written ← 0.
    while rows_written < N do
        Generate pseudo-random buffer buff using seed.
        for y = 0 to N − 1 do
            Extract trial_vec from buff.
            Apply transformation NTT(trial_vec, Q1, R1).
            if trial_vec contains no zero elements then
                Store trial_vec in A[mat][rows_written].
                rows_written ← rows_written + 1.
                if rows_written = N then
                    break inner loop.
                end if
            end if
        end for
    end while
end for
return A.
```

**Algorithm 2** **genC2**($elm1, elm2, m, SIG\_fs, SIG\_r$)

**Generates a constraint element $v$ by hashing inputs and reducing modulo $Q1$.**

**Require:** Elements elm1, elm2 of size $N$, public variables $m$, SIG_fs, SIG_r of size SEED_SIZE, and prime modulus $Q1$.

**Ensure:** Element $v[N]$ with non-zero elements.

```
Initialize SHAKE256 context: mdctx.
if mdctx initialization fails then
    Throw error and terminate.
end if
Begin SHAKE256 hashing process.
Update hash with elm1, elm2, m, SIG_fs, and SIG_r.
Finalize hash to produce hash_output[N × sizeof(int32_t)].
for i = 0 to N − 1 do
    Extract val from hash_output[i].
    Compute v[i] ← abs(val) mod Q1.
    if v[i] = 0 then
        Set v[i] ← 1 to ensure non-zero component.
    end if
end for
Free SHAKE256 context: mdctx.
return v.
```

**Algorithm 3 genC1**($pk, SIG\_MATRIX$)

**Generates constraining element set $C1$ by computing a cubed and hashed version of $pk \cdot B$.**

**Require:** Element $pk[N]$, signature matrix SIG_MATRIX$[K][N][N]$, and prime modulus $Q1$.

**Ensure:** Matrix $C1[K][N]$ with processed values.

Initialize vector $LHS \leftarrow pk$.
Initialize $result[N] \leftarrow 0$, $LHS[N] \leftarrow pk$.
**for** mat $= 0$ to $K-1$ **do**
  Reset $result[N] \leftarrow 0$.
  result $\leftarrow$ `MatrixVectorProduct`(SIG_MATRIX[mat], LHS, $Q1$).
  LHS $\leftarrow$ result.
  Compute $LHS \leftarrow LHS^3 \mod Q1$.
  Initialize SHAKE256 context: mdctx.
  **if** mdctx initialization fails **then**
    Throw error and terminate.
  **end if**
  Begin SHAKE256 hashing process.
  Update hash with LHS, result.
  Finalize hash to produce hash_output$[N]$].
  **for** $i = 0$ to $N-1$ **do**
    Extract $val$ from hash_output$[i]$.
    Compute $v[i] \leftarrow \text{abs}(val) \mod Q1$.
    **if** $v[i] = 0$ **then**
      Set $v[i] \leftarrow 1$ to ensure non-zero component.
    **end if**
  **end for**
  Free SHAKE256 context: mdctx.
  Store $LHS$ in $C1$[lat].
**end for**
**return** $C1$.

Table 4: Parameter Values for Levels 1, 3, and 5

| **Level** | **Dimension** ($N$) | $\omega$ ($R1$) | **Chain** ($K$) | ($Q1$) | ($SEED\_SIZE$) |
|---|---|---|---|---|---|
| 1 | 64 | 81 | 8 | 257 | 16 |
| 3 | 128 | 9 | 6 | 257 | 24 |
| 5 | 256 | 3 | 4 | 257 | 32 |

## Key Generation Algorithm

**Algorithm 4 KeyGen()**

**Generates public and private keys.**

**Require:** Prime modulus $Q1$, root $R1$, dimension $N$, number of chains $K$, seed size SEED_SIZE.

Initialize secretKeys$[K][N]$ uniformly random $x$ in the range $[1, 255]$.
Initialize matrix: MATRIX_A$[K][N][N]$.
Initialize matrix: MATRIX_U$[K][N][N]$.
Generate random seed: `PK_SEED_A` of length SEED_SIZE.
Generate random seed: `SK_SEED_U` of length SEED_SIZE.
Sample `MATRIX_A` using `PK_SEED_A`.
Sample `MATRIX_U` using `SK_SEED_U`.
Initialize current_pk $\leftarrow$ secretKeys[0].
current_pk $\leftarrow$ `NTT`(current_pk, $Q1$, $R1$).
**for** I = 0 to $K-1$ **do** ▷ Iterate through the chain of transformations.
    Initialize result$[N] \leftarrow 0$.
    **if** I $> 0$ **then**
        Update skey $\leftarrow$ secretKeys[I] and compute skey $\leftarrow$ `NTT`(skey, $Q1$, $R1$).
        Element-wise multiplication: current_pk $\leftarrow$ skey $\circ$ current_pk $\mod Q1$.
    **end if**
    result $\leftarrow$ `MatrixVectorProduct`(MATRIX_A[I], current_pk, $Q1$).
    current_pk $\leftarrow$ result.
    result$[N] \leftarrow 0$.
    result $\leftarrow$ `MatrixVectorProduct`(MATRIX_U[I], current_pk, $Q1$).
    Apply hiding function: current_pk $\leftarrow px$(result).
**end for**
Ensure non-zero condition: nonzero_count(current_pk) $\geq N$.
**if** Condition fails **then**
    Retry key generation.
**end if**
**return** current_pk, PK_SEED_A, SK_SEED_U, secretKeys[K]

## 10.4 Signature Generation

**Algorithm 5 Sign**($m, secretKeys[K], PK\_SEED\_A, pk\_elem, SK\_SEED\_U$)

**Generates a signature for a message.**

**Require:** Prime modulus $Q1$, root $R1$, dimension $N$, chain count $K$, seed size $SEED\_SIZE$, message $m$, , secret keys $secretKeys[K][N]$, public seed $PK\_SEED\_A$, public key $pk\_elem$, secret seed $SK\_SEED\_U$.

Initialize sig$[N] \leftarrow 0$.
Set SIG_COMPLETED $\leftarrow 0$.
Generate random bytes: rand_A, rand_B of size $SEED\_SIZE$.
Compute FS using `shake256`(rand_A, PK_SEED_A, pk_element).
Compute SIG_SEED_B using `shake256`(FS, $m$, rand_B, pk_element).
Sample matrix: MATRIX_B$[K][N][N]$ using SIG_SEED_B.
Sample matrix: MATRIX_U$[K][N][N]$ using SK_SEED_U.
Initialize C1$[K][N]$ via `genC1`($pk$, MATRIX_B).
Set sig $\leftarrow$ secretKeys[0] and apply `forward_ntt`(sig, $Q1, R1$).
**for** I $= 0$ to $K - 1$ **do**
    **if** I $> 0$ **then**
        skey $\leftarrow$ `forward_ntt`(secretKeys$[I]$, $Q1, R1$).
        sig $\leftarrow$ skey $\circ$ sig $\mod Q1$.
    **end if**
    result$[N] \leftarrow$ `MatrixVectorProduct`(MATRIX_B$[I]$, sig, $Q1$).
    sig $\leftarrow$ result
    result$[N] \leftarrow 0$.
    result$[N] \leftarrow$ `MatrixVectorProduct`(MATRIX_U$[I]$, sig, $Q1$).
    sig $\leftarrow$ sig $\circ$ C1[I] $\mod Q1$.
    Apply Hiding Function sig $\leftarrow px$(result)).
**end for**
Apply `inverse_ntt`(sig, $Q1, R1$).
Validate Entropy C3_CHECK $\leftarrow$ Verify_entropy($sig$).
C3 Check Retry after clearing buffers if C3_CHECK $= 1$.
Count non-zero elements in sig.
**if** nonzero_count(sig) $\geq$ N **then**
    Output sig, FS, rand_B.
**else**
    Retry after clearing buffers.
**end if**

## 10.5 Signature Verification

This function uses the Fiat-Shamir heuristic to reconstruct the signature basis seed that was used during the signing process. Together with the public key matrix seed, both public and signature matrices are sampled and 'swapped', such that sig signature element is transformed by the public basis and the public key element is transformed by the signature basis. The public key is also isomorphically transformed by the masking value that was used at each layer during signing. This mask is derived from the interaction between the public key and the signature basis, effectively binding them together.

Additionally as valid signatures are information theoretically guaranteed to have observational entropy at or near maximum, we leverage this to detect potential forgeries.

**Algorithm 6 Verify**$(m, PK\_SEED\_A, pk\_elem, sig, FS, rand\_B)$

**Verifies the signature of a message.**

**Require:** Message $m$, public seed $PK\_SEED\_A$, public key $pk\_elem$, signature $sig$, Fiat-Shamir heuristic $FS$ and randomizer $rand\_B$.

Initialize SIG_SEED_B[$SEED\_SIZE$], C2[$N$], $C3\_CHECK \leftarrow 0$.
Validate Entropy C3_CHECK $\leftarrow$ Verify_entropy($sig$).
C3 Check **return** 0 if C3_CHECK = 1.
Compute C2 using `genC2`$(sig, pk\_elem, m, FS, rand\_B)$.
Apply `forward_ntt`$(C2, Q1, R1)$ and `forward_ntt`$(sig, Q1, R1)$.
Construct temp_fs by concatenating FS, $m$, rand_B, and $pk\_elem$.
Compute SIG_SEED_B using `shake256`(temp_fs).
Sample matrix: PK_MATRIX[$K$][$N$][$N$] using PK_SEED_A
Sample matrix: SIG_MATRIX[$K$][$N$][$N$] using SIG_SEED_B.
Initialize LHS[$N$] $\leftarrow pk\_elem$, RHS[$N$] $\leftarrow sig$.
Compute C1[$K$][$N$] using `genC1`$(pk, SIG\_MATRIX)$.
**for** I = 0 to $K-1$ **do**
    LHS $\leftarrow$ LHS $\circ$ C2 $\mod Q1$.
    LHS $\leftarrow$ `MatrixVectorProduct`(SIG_MATRIX[I], LHS).
    LHS $\leftarrow$ LHS $\circ$ C1[I] $\mod Q1$.
    **if** $I \neq K-1$ **then**
        Apply Equivocation function: LHS $\leftarrow px$(LHS).
    **end if**
**end for**
**for** I = 0 to $K-1$ **do**
    RHS $\leftarrow$ RHS $\circ$ C2 $\mod Q1$.
    RHS $\leftarrow$ `MatrixVectorProduct`(PK_MATRIX[I], RHS).
    **if** $I \neq K-1$ **then**
        Apply Equivocation function: RHS $\leftarrow px$(RHS).
    **end if**
**end for**
Compare RHS and LHS.
**return** 0 if equal, otherwise 1.

### Observed Entropy Rejection Sampling

As part of what we are calling the third constraint, we want to rejection sample based on the observed randomness of the $\sigma$ input. We are considering elements of length of $n = \{64, 128, 256\}$ and have implemented constraints thus far on two probabilistic features, bit level and byte level randomness. In the byte probability case, we are measuring each component byte value, with the ideal $\sigma$ having zero colliding component values. However, we find that given the small sets of we are considering, we see multiple values appear two or three times, despite being valid.

To increase the granularity of randomness checking, we measure the raw ratio of 0 and 1 value bits across the array. For $n = 128$, where an ideally random signature would have

512 value 0 bits and 512 value 1 bits. To constrain input signatures to approximately 1/10 of the total possible modular space, we set the threshold ratio to 0.991. But, a "downshifted" element where components were in the range of $\{0, \ldots, 255\}$ entirely consisting of value 128 would pass with 512 0 bits and 512 1 bits. To correctly reject invalid signatures, we measure both bit level entropy and byte level entropy.

Table 5: Observed Entropy Thresholds for Different Values of using Byte probabilities $N$

| $N$ | **H_THRESHOLD** |
|---|---|
| 64 | 5.8 |
| 128 | 6.8 |
| 256 | 7.8 |

Table 6: Observed Entropy Thresholds for Different Values of using Bit probabilities $N$

| $N$ | **HB_THRESHOLD** |
|---|---|
| 64 | 0.991 |
| 128 | 0.991 |
| 256 | 0.991 |

**Algorithm 7 Verify_Entropy**($sig$)

**Validates the entropy of a signature.**

**Require:** Signature $sig$.

Extract entropy-relevant components from $sig$: SIG_VALUES $\leftarrow$ `extract`($sig$).
Compute the empirical distribution DIST of SIG_VALUES over the modular domain $[0, Q)$.
Calculate the Shannon entropy H_SIG:

$$\text{H\_SIG} \leftarrow -\sum_{x \in \text{DIST}} p(x) \log p(x),$$

where $p(x)$ is the probability of $x$ in DIST.
Extract count of 0 bits and 1 bits from $sig$ as COUNT0 and COUNT1.
Compute Ratio as HB_SIG.
**if** H_SIG $<$ H_THRESHOLD and HB_SIG $<$ HB_THRESHOLD **then**
    **return** 1 ▷ Entropy too low, validation fails.
**else**
    **return** 0 ▷ Entropy validation succeeds.
**end if**

While the underlying concept of rejection sampling based on entropy should be clear, the exact implementation is to be refined in subsequent revisions of this preprint. The achievable constraint is that the adversary cannot forge $\sigma$ using the entire ambient modular space, but only a specific fraction of it. This will lead to more refined and accurate probability.

## 10.6 Hiding Function

Table 7: Hiding Function NTT Parameters - Prime Fields and $\omega$ Roots of Unity

| Level | Q1 | R1 | Q2 | R2 | R3 |
|---|---|---|---|---|---|
| I | 257 | 81 | 1283 | 3 | 3 |
| III | 257 | 9 | 1283 | 3 | 3 |
| V | 257 | 3 | 1283 | 3 | 3 |

Each round of modular addition causes approximately half of the coefficients to wrap around the modulus, creating information loss and diffusion. Mapping to a higher modulus, combined with scaling and permutation is responsible for the bulk of pre-images introduced by the hiding function. This function can be modified based on a predetermined optimal mapping ratio of output to potential valid pre-image inputs. As an initial setting, for level III, we target a compression ratio of $2^{293}$ to 1 out of $q^n$ possibilities, creating a computationally sufficient number of indistinguishable elements per coset, invariant across both the hidden and ambient group elements.

**Algorithm 8 Hiding Function** $px(\text{in\_elem})$

**Transforms an input, a coset of group $N$, altering the structure using a series of modular operations creating a many-to-one mapping.**

**Require:** Input vector in_elem[$N$], prime moduli $Q1$, $Q2$, roots $R1$, $R2$.
**Ensure:** Output vector out_elem[$N$].

1: Allocate temporary arrays: vecsq[$N$],vec[$n$].
2: Copy vecsq ← in_elem.
3: vecsq ← `inverse_ntt`(vecsq, $Q1$, $R1$).
4: vecsq ← `forward_ntt`(vecsq, $Q2$, $R2$).
5: Copy vec ← vecsq.
6: vecsq ← `pointwise_addition`(vecsq, vecsq, $Q2$).
7: vecsq ← `pointwise_addition`(vecsq, vec, $Q2$).
8: vecsq ← `pointwise_addition`(vecsq, vec, $Q2$).
9: vecsq ← `pointwise_addition`(vecsq, vec, $Q2$).
10: vecsq ← `inverse_ntt`(vecsq, $Q2$, $R2$).
11: vecsq ← `forward_ntt`(vecsq, $Q1$, $R1$).
12: out_vec ← vecsq
13: **return** out_vec.

Note that step 11 performs an NTT forward transformation in the $Q1$ domain. After step 10, as a result of the internal diffusion and inverse NTT from $Q2$, we will have an element, (for level III) will look like this:

320, 925, 1060, 280, 475, 730, 1065, 20, 1120, 845, 1110, 125, 255, 430, 1245, 230, 165, 300, 695, 565,
980, 1005, 175, 860, 1135, 785, 610, 765, 760, 855, 1000, 175, 485, 310, 635, 710, 1015, 1110, 240, 1155, 40,
960, 1225, 840, 545, 220, 1005, 390, 940, 765, 1245, 40, 840, 320, 750, 1170, 120, 410, 480, 1270, 530, 470,
380, 530, 290, 25, 350, 325, 1265, 1005, 1275, 0, 975, 1055, 315, 1005, 915, 985, 240, 545, 455, 730, 570,
875, 320, 60, 200, 835, 880, 1205, 685, 1190, 1200, 495, 260, 245, 300, 370, 120, 700, 795, 330, 295, 705,

$$660, 695, 320, 455, 905, 1095, 105, 300, 30, 145, 1095, 900, 285, 1010, 395, 650, 695, 465, 1195, 545, 1185,$$
$$305, 255, 1175, 128$$

This is a result of using an NTT $\omega$ that isn't technically 'valid' for this field and array size. This is fine and intentional. We aren't performing any convolutional operations with this field (so the $\omega$ value isn't a factor) and it gives us a uniformly distributed set of components across 0 to 1282. We perform step 11 to explicitly alias the array back to the same reside class as the $q = 257$ NTT domain. After step 11, for level III, the same array becomes:

$$253, 85, 34, 171, 146, 12, 128, 73, 193, 243, 20, 119, 243, 158, 19, 237, 198, 185, 149, 15, 0, 0, 88, 112, 0,$$
$$193, 0, 201, 181, 68, 131, 0, 216, 14, 96, 130, 206, 220, 168, 153, 251, 200, 194, 154, 116, 151, 59, 163, 152,$$
$$193, 228, 198, 107, 170, 243, 111, 117, 209, 215, 230, 145, 165, 29, 252, 200, 82, 52, 115, 241, 55, 221,$$
$$135, 137, 243, 119, 197, 169, 216, 162, 70, 196, 234, 236, 178, 201, 66, 114, 101, 249, 26, 4, 184, 30, 76,$$
$$218, 201, 97, 240, 186, 182, 83, 248, 184, 7, 162, 178, 90, 94, 205, 43, 69, 213, 234, 55, 131, 253, 49,$$
$$208, 167, 216, 130, 108, 18, 142, 155, 205, 137, 217, 128$$

This aliasing of components from a field $\approx 5$ times larger down to $q = 257$ is how we generate multiple pre-images per output, and compress inputs to a specific equivalence class.

Empirical testing demonstrates that outputs from the mapping function $px(\cdot)$ appear statistically uniform and indistinguishable from random values. This uniformity is achieved through the interplay of NTT-based projections, modular scaling, and iterative mixing operations applied over a larger finite field.

The vector $t$ spans the full ambient group $G$ as matrix $A$ is full rank and changes with each operation and secret $x$ is uniformly random. combined This projection $t' \equiv U \cdot t$ also spans the entire ambient space. This process effectively blurs the cosets of $N$, distributing them uniformly over $Z_n^q$ and mapped to a specific equivalence classes created by $px(\cdot)$.

As a result, recovering all valid indistinguishable pre-images for a given output element of $px(\cdot)$ is insufficient to reconstruct $N$. The pre-images include a superset of elements comprising valid members of unrelated ambient group elements, with probabilistically only one valid element. This amalgamation obscures the boundaries of the 'true' $N$, making it computationally infeasible to distinguish subgroup membership based solely on inversion of $px(\cdot)$.

# 11 Attack Models

In this section, we rigorously analyze the security of the proposed cryptographic scheme against both classical and quantum adversaries. We focus on proving that the scheme achieves IND-CPA (Indistinguishability under Chosen Plaintext Attack) security by demonstrating the computational infeasibility of recovering the hidden subgroup $N$ or distinguishing ciphertexts under the specified attack models.

## 11.1 Classical Adversaries

Classical adversaries are limited to polynomial-time algorithms and lack quantum computational capabilities. We will show that, under standard cryptographic assumptions, such adversaries cannot feasibly recover the private keys or forge valid signatures.

### 11.1.1 Preliminaries

Let us recall the key components:

- The public key $\mathrm{pk} = t'' = px(t')$, where $t' = U \cdot t \mod q$ and $t = A \cdot x \mod q$.
- The mapping function $px(\cdot)$ is a lossy, many-to-one function inducing high ambiguity.
- The hidden subgroup $N$ is embedded in the non-abelian group $G = H_{\text{left}} \ltimes H_{\text{right}}$.

### 11.1.2 Proof of Security Against Classical Adversaries

**Lemma 1 (Computational Indistinguishability).** *Under the assumption that $px(\cdot)$ is a pseudorandom function and that the underlying group operations are secure, any polynomial-time classical adversary has a negligible advantage in distinguishing between valid signatures and random elements, or in recovering the private key $x$ or the matrix $U$.*

**Proof.** To prove this lemma, we proceed by contradiction. Assume there exists a polynomial-time classical adversary $\mathcal{A}$ that can distinguish valid signatures or recover $x$ or $U$ with non-negligible probability.

**Step 1: Reduction to the Hardness of NAHSP.**

Recall that recovering $x$ or $U$ is equivalent to solving the Non-Abelian Hidden Subgroup Problem (NAHSP) in the group $G$.

- The adversary's task reduces to finding $N$ given oracle access to $f(g) = px(U \cdot A \cdot x)$.

- As established in Section 3, solving NAHSP in this group is computationally infeasible for classical adversaries.

**Step 2: Indistinguishability of $px(\cdot)$ Outputs.**

- The function $px(\cdot)$ introduces high ambiguity, mapping exponentially many inputs to the same output.

- From Lemma 3 in Section 4.3, we know that the outputs of $px(\cdot)$ are computationally indistinguishable from uniform random elements in $\mathbb{Z}_q^n$.

**Step 3: Adversary's Advantage is Negligible.**

- The adversary $\mathcal{A}$ cannot distinguish between $px(U \cdot A \cdot x)$ and a random element without solving NAHSP.

- The probability that $\mathcal{A}$ successfully recovers $x$ or $U$ is bounded by $\varepsilon = \frac{1}{2^\lambda}$, where $\lambda$ is the security parameter (e.g., $\lambda = 297$ as per the preimage count).

- Since $\varepsilon$ is negligible, $\mathcal{A}$ cannot succeed with non-negligible probability.

**Conclusion.** Therefore, under standard cryptographic assumptions, no polynomial-time classical adversary can break the scheme, ensuring IND-CPA security against such adversaries.

## 11.2 Quantum Adversaries

Quantum adversaries have access to quantum computational resources, including algorithms like the Quantum Fourier Transform (QFT) and Grover's algorithm. We will demonstrate that even with these capabilities, adversaries cannot feasibly compromise the scheme.

### 11.2.1 Proof of Security Against Quantum Adversaries

**Lemma 2 (Resistance to Quantum Attacks).** *Under the assumption that the NAHSP is hard for quantum computers in non-abelian groups, and given the properties of the mapping function $px(\cdot)$, any polynomial-time quantum adversary has a negligible advantage in breaking the scheme.*

**Proof. Step 1: Non-Abelian Structure Prevents Efficient QFT-Based Attacks.**

- Quantum algorithms like Shor's algorithm rely on the ability to perform efficient QFT over abelian groups.
- The group $G = H_{\text{left}} \ltimes H_{\text{right}}$ is non-abelian, as shown in Section 3.1.
- As a result, the standard QFT does not provide a means to solve the hidden subgroup problem efficiently in $G$.

**Step 2: Ambiguity Introduced by $px(\cdot)$.**

- The mapping function $px(\cdot)$ further complicates any attempt to extract information via quantum algorithms.
- From Lemma 7 in Section 4.7, even if a quantum adversary could invert $px(\cdot)$, they would face an exponentially large preimage space, with $2^{297}$ indistinguishable candidates.

**Step 3: Grover's Algorithm is Ineffective Due to Exponential Search Space.**

- Grover's algorithm provides a quadratic speedup for unstructured search problems.
- Grover's algorithm generally is easily applied to chained systems with multiple secrets.

**Step 4: No Known Quantum Algorithm Solves NAHSP in Non-Abelian Groups Efficiently.**

- Despite extensive research, no quantum algorithm has been found that solves the NAHSP efficiently in general non-abelian groups.
- The hardness of NAHSP in such groups is a widely accepted assumption in quantum cryptography.

**Conclusion.** Given the non-abelian structure of the group and the properties of $px(\cdot)$, quantum adversaries cannot break the scheme with non-negligible probability. Therefore, the scheme achieves IND-CPA security even in the presence of quantum adversaries.

## 11.3 IND-CPA Security Proof

We now provide a formal proof that the scheme is IND-CPA secure.

**Theorem 1 (IND-CPA Security).** *Under the assumption that the NAHSP is hard for both classical and quantum adversaries, and that $px(\cdot)$ behaves as a pseudorandom function, the proposed digital signature scheme is IND-CPA secure.*

**Proof. Definition of IND-CPA Security.**

A digital signature scheme is IND-CPA secure if no polynomial-time adversary can distinguish between signatures of chosen messages, even when given access to a signing oracle.

**Game-Based Proof Structure.**

We consider the standard IND-CPA security game between a challenger and an adversary $\mathcal{A}$:

1. **Setup:** The challenger generates a public-private key pair $(pk, sk)$ and provides $pk$ to $\mathcal{A}$.

2. **Query Phase:** $\mathcal{A}$ may request signatures on messages of its choice.

3. **Challenge Phase:** $\mathcal{A}$ selects two messages $m_0$ and $m_1$. The challenger randomly selects $b \in \{0, 1\}$ and returns $\sigma = \text{Sign}(m_b, sk)$.

4. **Guess Phase:** $\mathcal{A}$ outputs a guess $b'$. The adversary wins if $b' = b$.

Our goal is to show that $\Pr[b' = b] \leq \frac{1}{2} + \varepsilon$, where $\varepsilon$ is negligible.

**Analysis.**

Assume, for contradiction, that $\mathcal{A}$ can win the game with a non-negligible advantage $\delta$.

**Step 1: Construction of a Simulator to Solve NAHSP.**

We construct a simulator $\mathcal{S}$ that uses $\mathcal{A}$ to solve the NAHSP:

- $\mathcal{S}$ receives an instance of NAHSP in $G$ and needs to find the hidden subgroup $N$.

- $\mathcal{S}$ simulates the challenger for $\mathcal{A}$, using the NAHSP instance to generate public keys and signatures.

- When $\mathcal{A}$ outputs $b'$, $\mathcal{S}$ uses this information to extract information about $N$.

**Step 2: Contradiction with the Hardness of NAHSP.**

- If $\mathcal{S}$ can solve NAHSP using $\mathcal{A}$'s advantage $\delta$, then the hardness assumption of NAHSP is violated.

- Therefore, $\delta$ must be negligible.

**Step 3: Security Reduction via Hybrid Arguments.**

- We can define a sequence of hybrid experiments transitioning from the real scheme to an ideal scheme where signatures are replaced with random values.

- The indistinguishability of outputs from $px(\cdot)$ ensures that $\mathcal{A}$ cannot distinguish between hybrids with non-negligible advantage.

**Conclusion.**

Since any non-negligible advantage $\delta$ leads to a contradiction with the hardness of NAHSP, we conclude that $\mathcal{A}$ cannot win the IND-CPA game with more than negligible advantage. Therefore, the scheme is IND-CPA secure.

## 11.4 Resistance to Forgery Under Chosen Message Attacks

**Theorem 2 (Unforgeability under Chosen Message Attack).** *Assuming the hardness of NAHSP and the collision resistance of the hash function $J(\cdot)$, the proposed scheme is existentially unforgeable under chosen message attacks (EUF-CMA).*

**Proof. Step 1: Assumptions and Definitions.** - Let $\mathcal{A}$ be an adversary attempting to forge a valid signature $\sigma^*$ for a message $m^*$ that has not been queried during the signing oracle phase. - The scheme uses the Fiat-Shamir heuristic to bind the signature to the message, the public key, and a random nonce. - A valid forgery requires $\mathcal{A}$ to produce $\sigma^*$ such that:

$$\text{Verify}(m^*, \sigma^*, pk) = \text{true}.$$

**Step 2: Connection to NAHSP and $px(\cdot)$.** - To forge $\sigma^*$, $\mathcal{A}$ must either: 1. Recover the private key $x$ or the hidden matrix $U$, allowing the computation of valid transformations. This is equivalent to solving the Non-Abelian Hidden Subgroup Problem (NAHSP), which is assumed to be hard. 2. Generate a valid preimage under $px(\cdot)$ without access to the private key or matrix. The lossy, many-to-one nature of $px(\cdot)$ ensures that the adversary cannot distinguish valid preimages from an exponentially large indistinguishable set.

**Step 3: Resistance to Hash Function Collisions.** - The Fiat-Shamir heuristic involves the hash function $J(\cdot)$, which produces a binding challenge for the signature. For $\mathcal{A}$ to forge $\sigma^*$, it must either: 1. Find a collision $J(pk \parallel m^* \parallel r) = J(pk \parallel m' \parallel r')$, which is infeasible due to the assumed collision resistance of $J(\cdot)$. 2. Guess the challenge generated by $J(\cdot)$ and align it with a valid subgroup element. The probability of such a guess is negligible due to the high entropy of the output space of $J(\cdot)$.

**Step 4: Reduction to a Hard Problem.** - Assume $\mathcal{A}$ successfully forges $\sigma^*$ with non-negligible probability. We construct a simulator $\mathcal{S}$ that uses $\mathcal{A}$ to solve NAHSP or find a collision in $J(\cdot)$: 1. $\mathcal{S}$ simulates the signing oracle for $\mathcal{A}$, generating signatures using a secret key $x$ and the private matrix $U$. 2. If $\mathcal{A}$ outputs a valid forgery $\sigma^*$, $\mathcal{S}$ uses $\sigma^*$ to extract information about the hidden subgroup $N$ or to find a collision in $J(\cdot)$. 3. Since both outcomes contradict the hardness of NAHSP or the collision resistance of $J(\cdot)$, $\mathcal{A}$'s success probability must be negligible.

**Step 5: Conclusion.** - The adversary $\mathcal{A}$ cannot forge a valid signature $\sigma^*$ on a message $m^*$ without solving NAHSP, inverting $px(\cdot)$, or finding a collision in $J(\cdot)$, all of which are computationally infeasible. - Therefore, the proposed scheme is existentially unforgeable under chosen message attacks (EUF-CMA).

## 11.5 Inapplicability of CCA2 Security

It is important to note that oue proposed digital signature scheme does not incorporate a decryption oracle, as it is not designed to handle encrypted messages or ciphertext directly. The absence of such an oracle renders the chosen ciphertext attack (CCA2) model irrelevant for this construction.

Instead, the security of the scheme is analyzed under the IND-CPA (Indistinguishability under Chosen Plaintext Attack) and EUF-CMA (Existential Unforgeability under Chosen Message Attack) models, which are sufficient and appropriate given the nature of the signature application.

By excluding a decryption oracle, the scheme eliminates a common attack vector associated with adaptive adversaries in CCA2 scenarios, further solidifying its robustness in practical cryptographic deployments.

### 11.5.1 Brute Force Key Recovery

Brute force recovery of secrets is implementation dependent. The scheme covered in this document leverages a single private matrix seed for each instance in the chain, and unique password $x$ elements per instance. Based on size and security concerns, if private key size was critical, the private key, in theory, could be shrunk to a single secret seed that expanded to provide every hidden matrix and $x$ element. If absolute security were paramount, each hidden matrix could be derived from its own seed, or stored fully instantiated. In this brief analysis, we will simply derive the costs of brute forcing the scheme as described. Each level has $k$ chains, with one $x$ secret element, effectively $n$ bytes long. Additionally, each level has one hidden matrix seed, of $SEED\_SIZE$ length, in bytes. Relative sizes are listed below:

Table 8: Brute Force Secret Byte Analysis

| n | k | Single x | All x | Hidden Seed | Total Secret | Complexity |
|---|---|---|---|---|---|---|
| 64 | 8 | 64 | 512 | 16 | 528 | $2^{4224}$ |
| 128 | 6 | 128 | 768 | 24 | 792 | $2^{6336}$ |
| 256 | 4 | 256 | 1024 | 32 | 1056 | $2^{8448}$ |

As even level I requires $2^{4224}$ classical operations, brute force attacks do not appear to be a practical concern with the variant as described in this paper. Note that this could change depending on various implementation optimizations.

### 11.6 Conclusion on Attack Models

Through rigorous proofs, we have established that the proposed cryptographic scheme is secure against both classical and quantum adversaries. The security relies on:

- The computational hardness of the NAHSP in non-abelian groups.
- The pseudorandomness and computational indistinguishability introduced by the mapping function $px(\cdot)$.
- The collision resistance of the hash function used in the Fiat-Shamir heuristic.

These properties collectively ensure that adversaries cannot feasibly recover private keys, forge signatures, or distinguish ciphertexts, thereby achieving IND-CPA security and resisting forgery under chosen message attacks.

## 12 Implementation and Efficiency

### 12.1 Performance Evaluation

Table 9: Compute Cycles (in Megacycles) for Key Generation, Signing, and Verification

| **Level** | **Key Generation (Mc)** | **Signature (Mc)** | **Verification (Mc)** |
|---|---|---|---|
| I | .49 | .38 | .44 |
| III | 1.03 | .958 | 1.10 |
| V | 9.46 | 3.25 | 3.48 |

Platform: Apple MacBook M2 MAX with 32 GB RAM.

| Level | $n$ | PK (bytes) | Sig (bytes) | $k$ |
|---|---|---|---|---|
| I | 64 | 80 | 96 | 8 |
| III | 128 | 152 | 176 | 6 |
| V | 256 | 288 | 320 | 4 |

Table 10: Public Key, Signature Sizes, and Chain Instances Across Levels

## 12.2 Comparison with Other Schemes

Table 11: Performance Metrics of Alternative Signature Algorithms (in Bytes and Megacycles)

| Algorithm | PK+SIG (Bytes) | Sign (Mcycles) | Verify (Mcycles) |
|---|---|---|---|
| Dilithium2 | 3,732 | 0.333 | 0.118 |
| Dilithium3 | 5,261 | 0.529 | 0.179 |
| Dilithium5 | 7,219 | 0.642 | 0.280 |
| MAYO1 | 1,489 | .461 | .175 |
| MAYO3 | 3,233 | 1.664 | .610 |
| MAYO5 | 5,846 | 4.150 | 1.186 |
| HAWK-512 | 1,573 | .085 | .148 |
| HAWK-1024 | 3,661 | .180 | .303 |
| Falcon-512 | 1,563 | 1.010 | .081 |
| Falcon-1024 | 3,073 | 2.053 | .161 |
| SLH-DSA-128f | 17,120 | 239.794 | 12.910 |
| SLH-DSA-192f | 35,712 | 386.862 | 19.877 |
| SLH-DSA-256f | 49,920 | 763.942 | 19.886 |
| SQIsign-1 | 241 | 5,669.00 | 108.00 |
| SQIsign-3 | 359 | 43,760.00 | 654.00 |
| SQIsign-5 | 463 | 158,544.00 | 2,177.00 |
| EQISIGN-I | 176 | .38 | .44 |
| EQISIGN-III | 328 | .958 | 1.10 |
| EQISIGN-V | 608 | 3.25 | 3.45 |

The above data was been gathered from the PQShield Post-Quantum signatures zoo, we have not verified it and the most recent developments may not be reflected, but we feel the source is accurate for this early comparison. To date, our priority has been theoretical security and communication efficiency, with little attention paid to performance optimization. Over time, it is almost certain our performance numbers will improve beyond our reference instance. We are using portable ANSI C, openSSL/TSL, our portion of the code does not leverage intrinsics, is single threaded, and the majority of time is currently spent expanding matrices from seed. With dedicated effort we feel performance and optimization will yield significant improvements. That said, we do not expect to outperform ML-DSA in terms of compute, nor do we expect computational performance to be a barrier for adoption.

[0]Data source: `https://pqshield.github.io/nist-sigs-zoo/c`

### 12.3 Rejection Sampling of Zero Coefficients in Output Variables

The elimination of zero value elements is a strong method to create resilience against heuristic cryptanalysis, both quantum and classical. Due to the zero product property of finite fields, allowing the value zero tends to cause accumulation in output variables (keys, signatures) opening a window for exploitation. Additionally, while not covered elsewhere, in practical implementations, combined with the working modulus of $q = 257$, we are able to leverage the possible component range to increase communications efficiency. Under normal circumstances, operating modulo 257 results in elements that range from $\{0, ..., 256\}$, or 257 unique values, requiring 9 bits to accurately represent. The elimination of zero via rejection sampling allows each value to map to 256 elements which can be represented using 8 bits.

Simply, when serializing variables for transmission, before transmission we simply subtract one from each value. Upon receiving keys or signatures we 'reconstitute' them by incrementing each by one, mapping back to the appropriate original values. This is an easy optimization to align with normal machine word boundaries.

## 13 Open Research Questions

1. For function px(), the optimal ratio of class size to class number in relation to group sizes $|G|$ and $|N|$ is an open question.

2. Correct formal complexity classification of NAHSP under extreme equivocation. The NAHSP is conjectured to be in the EXP complexity class, with no known efficient quantum algorithm. These conjectures should be proven if possible, but this work is outside the scope of this paper.

## 14 Applications of Matrix based NAHSP-Based Cryptography

The NAHSP-based cryptographic scheme offers a versatile, lightweight, and quantum-resistant framework for diverse applications. Its compact signatures, efficient communication requirements, and ability to deploy through software patches without new hardware set it apart from traditional lattice-based approaches such as Dilithium. These features enable its use in domains ranging from terrestrial networks to undersea and RF-limited environments, making it uniquely suited for next-generation cryptographic needs.

### 14.1 Core Cryptographic Capabilities

An NAHSP-based scheme based on bilinear matrices can be extended beyond foundational primitives to more advanced cryptographic constructions:

- Digital Signatures: Compact and efficient signatures ensure secure authentication, document signing, and certificate management, with communication sizes significantly smaller than lattice-based systems like Dilithium.

- Public Key Agreement: Enables fast, quantum-resilient key exchanges with minimal communication overhead, ideal for constrained networks.
- Identity-Based and Attribute-Based Cryptography: Supports fine-grained access control, allowing secure communication based on user identities or attributes without requiring heavy key distribution infrastructure.
- Zero-Knowledge Proofs (ZKPs): Facilitates privacy-preserving verification of statements without exposing underlying secrets, essential for regulatory compliance and secure interactions.

## 14.2 Efficient Communication and Deployability

**Compact Communication Sizes:**

- NAHSP-based cryptography achieves extremely small communication footprints, with signatures and keys often requiring a fraction of the size used by lattice-based systems. This efficiency is critical in bandwidth-limited environments such as undersea and RF networks.
- Example: For a security level equivalent to Dilithium-III, the NAHSP-based scheme offers signatures of 80 bytes and public keys under 100 bytes, compared to the hundreds or thousands of bytes required by Dilithium.

**Software-Only Deployment:**

- Unlike lattice-based systems, which often require specialized hardware for efficient operation, NAHSP-based cryptography can be implemented as a drop-in replacement via software patches.
- This allows immediate deployment in existing infrastructures, including mobile devices, routers, and IoT systems, without the need for hardware upgrades.
- Rapid updates ensure forward compatibility with evolving security standards while minimizing deployment costs.

## 14.3 Secure Communication Across Diverse Environments

NAHSP-based cryptography's compact and efficient design enables secure communication in challenging and bandwidth-constrained domains:

- Cellular Networks: Ensures efficient and secure handshakes, even in low-latency 5G environments, where minimal communication overhead is critical.
- Radio Frequency (RF) Systems: Compact key sizes and signatures reduce transmission time in RF-constrained settings, such as military radios and satellite uplinks.
- Undersea Acoustics: Low-bandwidth undersea acoustic networks benefit from NAHSP's compact communication, enabling secure exchanges where data rates are severely limited.

## 14.4 Comparison with Dilithium and Other Systems

A matrix based NAHSP scheme addresses several limitations of Dilithium and similar lattice-based approaches:

- Smaller Communication Sizes: Signatures and keys are significantly more compact, reducing storage and bandwidth requirements.
- Flexibility Across Environments: Performs robustly in environments where lattice-based schemes face challenges, such as RF and undersea communication.
- Advanced Constructions: Offers natural support for identity-based encryption and zero-knowledge proofs, features that are generally not feasible to implement over schemes with noise.

## 14.5 Real-World Applications

**Critical Infrastructure and Defense:**

- Secures command and control systems in military and intelligence operations, ensuring quantum resilience and adaptability across RF and satellite links.
- Protects SCADA systems in critical infrastructure, such as energy grids and transportation networks, with lightweight and efficient cryptographic primitives.

**IoT and Edge Devices:**

- Provides secure authentication for IoT devices with constrained processing power, mitigating risks of botnet attacks and data breaches.
- Ensures efficient encryption and signing for edge devices in industrial and healthcare settings.

**Blockchain and Distributed Systems:**

- Enhances consensus mechanisms with compact, quantum-resistant signatures, reducing energy consumption and improving scalability.
- Secures smart contracts and cryptographic tokens with lightweight, efficient constructions.

**Telecommunications and Financial Systems:**

- Enables secure mobile payment systems and digital banking with minimal communication overhead, ensuring transaction authenticity and integrity.
- Modernizes public key infrastructure (PKI) for quantum resilience while minimizing deployment costs.

## 14.6 Conclusion

While the primary embodiment of this invention is a digital signature scheme leveraging the Non-Abelian Hidden Subgroup Problem (NAHSP) and equivocation via the px() mapping function, however the core mechanism is broadly applicable to other cryptographic primitives. These include, but are not limited to, public key exchange, encryption schemes (such as identity-based and attribute-based encryption), and zero-knowledge proofs. The underlying NAHSP-based obfuscation provides a foundation for secure and efficient cryptographic systems across various applications.

# 15 Future Work and Concluding Remarks

- While theoretically robust, the construction requires careful selection of optimal variables and implementation of the underlying mathematics to run in constant time. Achieving constant time execution mitigates side-channel attack and aligns with best practices for any algebraic cryptographic scheme.

- Like all novel forms of cryptography, extensive adversarial cryptanalysis is required. We welcome experienced cryptanalysis focused collaboration.

- Optimization of functions to leverage platform-specific SIMD instructions can significantly accelerate operations while maintaining constant runtime guarantees. This will enhance the scheme's practical viability across diverse hardware platforms.

- The scheme employs rejection sampling on intermediates with zero-value coefficients to prevent degeneracy. A thorough adversarial analysis is needed to evaluate whether this rejection sampling introduces potential vulnerabilities that could aid cryptanalysis. If identified, appropriate mitigations must be developed.

- Additionally, the rejection sampling method of zero components is fairly simple and is currently a major performance cost. By implementing a more optimal mechanism, these costs can be minimized.

Integrating quantum-resilient cryptographic systems into existing infrastructures remains a complex challenge, particularly for hardware-constrained environments or legacy systems reliant on established PKI frameworks. This NAHSP-based system offers a promising approach by providing compact signatures and practical efficiency suitable for retrofitting into current infrastructures, including standard-sized X.509 certificates. These properties make it a strong candidate for addressing the scalability and trust requirements needed in the transition to post-quantum security.

While this work may be among the first cryptographic systems to aim for practical information-theoretic security guarantees by design, its formal proofs and construction serve as a foundational step toward bridging the gap between theoretical resilience and real-world application. This approach diversifies the cryptographic landscape, complementing existing quantum-resilient efforts and enhancing robustness against diverse attack vectors.

Beyond its immediate applications, this cryptosystem opens new avenues of research across multiple fields:

- Cryptography: The NAHSP framework invites exploration into additional constructions such as group-based encryption, secure multi-party computation, and advanced privacy-preserving protocols.

- Complexity Theory: By leveraging non-abelian group properties, the system provides fertile ground for studying alternate hardness assumptions and their implications for classical and quantum computational limits.

- Quantum Algorithm Design: The inherent resilience of the scheme challenges researchers to explore novel quantum algorithms capable of addressing non-abelian group problems, advancing our understanding of quantum computational power.

- Systems Security: With its adaptability to constrained environments such as IoT, RF, and undersea acoustics, this system sets the stage for breakthroughs in secure communication under extreme conditions.

**Closing Statement:** This cryptographic scheme offers a significant contribution to the evolving landscape of post-quantum security. By providing practical information-theoretic guarantees and addressing key implementation challenges, it has the potential to transform how secure systems are designed and deployed. While further research and optimization remain, this work lays a strong foundation for future innovations in cryptography, complexity theory, and quantum algorithm design, positioning it as a critical component in the journey toward resilient and scalable global security systems.

# 16 Acknowledgements

This work is provisionally patented, USPTO 63/726,609.

# References

[1] Daniel J. Bernstein et al. "The SPHINCS+ Signature Framework". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. CCS '19. London, United Kingdom: Association for Computing Machinery, 2019, pp. 2129–2146. ISBN: 9781450367479. DOI: 10.1145/3319535.3363229. URL: https://doi.org/10.1145/3319535.3363229.

[2] Léo Ducas et al. "Dilithium: A high-speed lattice-based digital signature scheme". In: *CCS 2019*. 2019, pp. 897–918.

[3] Luca De Feo et al. *SQISign: compact post-quantum signatures from quaternions and isogenies*. Cryptology ePrint Archive, Paper 2020/1240. 2020. URL: https://eprint.iacr.org/2020/1240.

[4] David Garber. "Braid group cryptography". In: *Braids: Introductory lectures on braids, configurations and their applications*. World Scientific, 2010, pp. 329–403.

[5] Dimitri Grigoriev and Ilia Ponomarenko. *Constructions in public-key cryptography over matrix groups*. 2005. arXiv: math/0506180 [math.GR]. URL: https://arxiv.org/abs/math/0506180.

[6] Ki Hyoung Ko et al. "New public-key cryptosystem using braid groups". In: *Advances in Cryptology—CRYPTO 2000: 20th Annual International Cryptology Conference Santa Barbara, California, USA, August 20–24, 2000 Proceedings 20*. Springer. 2000, pp. 166–183.

[7] Patrick Longa, Wen Wang, and Jakub Szefer. *The Cost to Break SIKE: A Comparative Hardware-Based Analysis with AES and SHA-3*. Cryptology ePrint Archive, Paper 2020/1457. 2020. URL: `https://eprint.iacr.org/2020/1457`.

[8] Karl Mahlburg. "An overview of braid group cryptography". In: *preprint* (2004).

[9] Alexei Myasnikov, Vladimir Shpilrain, and Alexander Ushakov. "A practical attack on a braid group based cryptographic protocol". In: *Advances in Cryptology–CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005. Proceedings 25*. Springer. 2005, pp. 86–96.

[10] Alexei G Myasnikov, Vladimir Shpilrain, and Alexander Ushakov. *Non-commutative cryptography and complexity of group-theoretic problems*. 177. American Mathematical Soc., 2011.

[11] Michael Schmid et al. *Falcon Takes Off - A Hardware Implementation of the Falcon Signature Scheme*. Cryptology ePrint Archive, Paper 2023/1885. 2023. URL: `https://eprint.iacr.org/2023/1885`.

[12] Claude E. Shannon. "Communication theory of secrecy systems". In: *Bell Syst. Tech. J.* 28.4 (1949), pp. 656–715. DOI: `10.1002/J.1538-7305.1949.TB00928.X`. URL: `https://doi.org/10.1002/j.1538-7305.1949.tb00928.x`.

[13] Claude Elwood Shannon. "A Mathematical Theory of Communication". In: *The Bell System Technical Journal* 27 (1948), pp. 379–423. URL: `http://plan9.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf` (visited on 04/22/2003).