POLITECNICO DI TORINO

Master's Degree
in Mathematical Engineering

Master Thesis

Design Choices and Methodological Trade-offs in Lattice-Based Anonymous Credentials

Supervisors
prof. Danilo Bazzanella
Bianca Rampazzo (LINKS Foundation)
Andrea Vesco(LINKS Foundation)

Candidate
Flavio Gasparini

Academic Year 2025-2026

Acknowledgements

First and foremost, I would like to thank Prof. Bazzanella for guiding me through my Master's degree journey and for creating this study track about Cryptography; without him, none of this would have been possible. I also wish to thank Andrea Vesco for the opportunity to carry out my thesis work at Links, as well as my company supervisor, Bianca Rampazzo, for her invaluable support and patience over the past few months. Finally, my deepest thanks go to my family and friends for always standing by my side over the years. Without your unwavering support, this path would have been far more difficult and challenging.

Contents

1 Anonymous Credentials 1

1.1 The Credential System . . . 1
1.2 Anonymous Credentials . . . 2
1.2.1 Cryptographic Model and Required Properties . . . 2
1.3 State of the Art and Post-Quantum Cryptography . . . 3

2 Mathematical Preliminaries 5

2.1 Notation . . . 5
2.2 Coefficient Representation and Multiplication Matrices . . . 6
2.3 Lattices . . . 6
2.4 Module-SIS and Module-LWE Problems . . . 7
2.5 Cryptographic Primitives . . . 8
2.5.1 Cryptographic Hash Functions . . . 8
2.5.2 Commitment Schemes . . . 8
2.5.3 Modeling Interactive Protocols . . . 9
2.6 Probability Distributions . . . 9

3 Framework of an ℓ_2-Norm-Based ZKP 10

3.1 State of the Art . . . 11
3.2 Committing to the Module-SIS Secret . . . 12
3.2.1 Ajtai Commitment Scheme . . . 12
3.2.2 BDLOP Commitment Scheme . . . 13
3.2.3 The ABDLOP Hybrid Commitment Scheme . . . 14
3.3 Building an Interactive Protocol for the ABDLOP Commitment . . . 15
3.3.1 Validation of the ABDLOP Commitment . . . 16
3.3.2 Rejection Sampling . . . 17
3.3.3 Verifying the Norm of the Masked Secret . . . 21
3.4 Evaluating the ℓ_2 Norm via a Quadratic Equation . . . 22
3.4.1 Masking the Polynomial and Isolating the Constant Term . . . 23
3.4.2 Zero-Knowledge Proof of the Polynomial Relation . . . 25
3.5 ZKP for Solutions of Quadratic Equations . . . 25

3.5.1 Integrating the Quadratic Relation with the Commitment Scheme . . 27
3.5.2 Verifying Multiple Quadratic Relations 28
3.6 Checking the ℓ_2 Norm Bound 30
3.6.1 Transforming the Inequality into an Exact Equation 30
3.6.2 Preventing Modular Wrap-Around 31
3.7 Summary of the ℓ_2 Norm Verification Protocol 33
3.8 Parameter Instantiation 34
3.8.1 Challenge Space $\mathcal{C}^*$ 34
3.8.2 Standard Deviation for Rejection Sampling with Polynomial Challenges 38
3.8.3 Standard Deviation for Rejection Sampling to Prevent Modular Wrap-Around 38
3.9 Formal Security Guarantees and Parameter Constraints 39
3.10 Non-Interactive Zero-Knowledge via Fiat-Shamir 42

4 Construction of an Anonymous Credential System 45
4.1 A Hard Linear Problem: The ISIS_f Assumption 46
4.1.1 Choosing the Function f 47
4.1.2 The Interactive ISIS_f Assumption 48
4.1.3 Trapdoor Generation 48
4.2 Issuing a Credential 49
4.2.1 Implementing Selective Disclosure 50
4.2.2 Security in Concurrent Executions 51
4.2.3 The Modulus Lifting Technique: Two Approaches 51
4.2.4 The Full Protocol 53
4.3 Verification of the Credential 53
4.4 Security Parameters and Instantiation 60
4.5 Differences Between the LNP Framework and the BLNS Instantiation 62
4.6 Optimization of the Protocol by LINKS Foundation 62

5 Parameter and Algorithm Comparison 65
5.1 Managing the Transformation from Inequality to Equality 65
5.1.1 A Novel Greedy Approach for Inequality Management 67
5.2 Comparison of Rejection Algorithms 71
5.2.1 Comparison Between the Rej_2 and Rej_1 Approaches 72
5.2.2 Comparison Between the Rej_0 and Rej_1 Approaches 72
5.2.3 Summary of Rejection Algorithm Selection 74
5.3 Validation of the LINKS Foundation Parameter Instantiation for Rejection Sampling 75

6 Conclusion 79

Bibliography 81

Abstract

The advent of quantum computers is driving the cryptographic community to renew its underlying primitives to ensure security in a post-quantum world. Within this context, the development of new post-quantum anonymous credential models represents a crucial challenge: it is vital to ensure security and anonymity while preserving high performance, both in terms of memory overhead and computational efficiency for the prover and the verifier This thesis focuses on the analysis of the zero-knowledge proof (ZKP) framework presented by Lyubashevsky, Nguyen, and Plançon in 2022 [LNP22], and its subsequent application in the construction of an anonymous credential system proposed in the paper by Bootle, Nguyen, Lyubashevsky, and Sorniotti [BNLS23]. The security of the ZKP framework relies on well-known problems in lattice-based cryptography, namely MSIS and MLWE. This framework leverages the computation of the l_2 norm through a highly compact ZKP for linear and quadratic equations. To adapt this framework into an anonymous credential system, the $ISIS_f$ with a binary encoding function is utilized. This assumption rests on the computational difficulty of finding a short vector s and a binary preimage u such that $As = Bu$. The analysis conducted in this thesis highlights the motivations driving specific protocol design choices and the parametric conditions required to guarantee the model's security. Finally, the thesis analyzes the differences between the framework paper and the anonymous credentials paper, evaluating the pros and cons of each design choice. Specifically, it examines the various rejection sampling techniques employed and the methods used to transform the protocol from norm equality to inequality, proposing a novel approach specifically for this transformation.

Chapter 1

Anonymous Credentials

1.1 The Credential System

In the landscape of modern digital interactions, a fundamental requirement is the establishment of protocols where a user (the holder) demonstrates to a service provider (the verifier) the possession of specific access rights or attributes. This authorization is guaranteed through a credential, a digital artifact used to verify specific claims or permissions.

Credentials are typically emitted by an issuer, an entity, usually a trusted third party or the service provider itself, responsible for checking if the prover has the right to own a credential (see Figure 1.1 [VC]). The credentials rarely possess an infinite lifespan. Robust

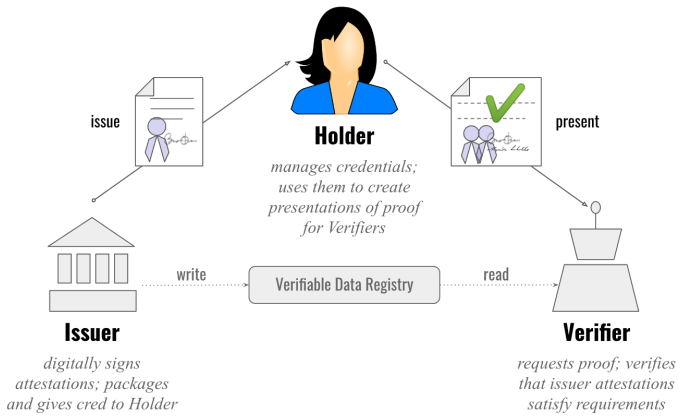

Figure 1.1. Credential Scheme

systems must therefore incorporate mechanisms for one-time-use or explicitly revocable credentials, ensuring that access rights can either expire automatically after a predefined duration or be actively revoked by the service provider.

These standard credential systems, however, introduce significant privacy risks. To grant a credential, the issuer has access to all the information about the holder, even if such

data is not strictly required for the specific service. Furthermore, when a prover presents a credential to a verifier, they typically reveal their full identity or a persistent unique identifier. This enables service providers, or observing third party trackers, to aggregate distinct access events into a comprehensive behavioral profile, effectively mapping digital activities to a specific user.

To solve this problem, a naïve countermeasure might involve issuing an identical credential to all holders, making it impossible to distinguish individual access events. However, this approach introduces severe operational and security vulnerabilities. Unlike physical tokens, digital assets are inherently non-scarce and can be duplicated indefinitely, which could trivially enable Denial of Service (DoS) attacks. Consequently, systems must retain the capacity to identify a malicious credential holder and revoke their privileges.

1.2 Anonymous Credentials

An effective solution to these privacy and security problems is the use of anonymous credentials. This approach adds structure to the credential, dividing the overall claim into individual attributes. The issuer then signs these attributes using specific cryptographic signatures.

Anonymous credential systems aim to guarantee the following core properties:

- **Unlinkability**: This ensures that when a user presents the same credential multiple times, to different services or at different times, neither the verifiers nor the issuer can link those actions together.

- **Selective Disclosure**: This allows the holder to choose and reveal only specific attributes from their credential (for example, proving an attribute that explicitly states you are over 18, hiding your exact date of birth).

1.2.1 Cryptographic Model and Required Properties

Designing a protocol for managing anonymous credentials requires formalizing two distinct processes: the *issuance* of the credential (between the issuer and the holder) and the *verification* of the credential (between the holder and the verifier) (see Figure 1.1). The goal is to satisfy the privacy properties outlined in Section 1.2, while simultaneously guaranteeing the standard security properties typical of cryptographic protocols. A primary security requirement is *unforgeability*: an attacker lacking the appropriate rights must have only a negligible probability of either tricking an issuer into granting a valid credential or fabricating a fake ZKP that a verifier will accept. To achieve this, the credential is intrinsically tied not only to its attributes but also to a cryptographic secret (a secret key) known exclusively to the holder. This prevents a malicious actor from using an intercepted credential, as they would not know the underlying secret.

To enforce this binding, anonymous credential systems rely on Zero-Knowledge Proofs (ZKPs). Formally, a ZKP is a cryptographic protocol between a prover (who holds a secret witness for a claim) and a verifier. This protocol must satisfy three fundamental properties:

- **Completeness:** If the statement is true and both parties behave honestly, the verifier will accept the proof with overwhelming probability (typically probability 1).
- **Soundness:** If the statement is false, a malicious prover cannot convince an honest verifier to accept the proof, except with a negligible probability. In the context of anonymous credentials, this is often extended to Knowledge Soundness (making it a Proof of Knowledge), which guarantees that a successful prover must actually know the underlying witness, rather than just the fact that the statement is true.
- **Zero-Knowledge:** The execution of the protocol leaks absolutely no information about the prover's secret witness to the verifier. Formally, this is guaranteed by the existence of an expected probabilistic polynomial-time simulator. This simulator can generate a valid proof transcript without any access to the witness, and this simulated proof is computationally indistinguishable from an interaction with a real prover.

ZKPs and related primitives are utilized to secure both the issuance and verification phases. During issuance, a prominent cryptographic technique is the *blind signature*. This allows an issuer to digitally sign a holder's credential by verifying the public attributes required to grant it, without learning the hidden attributes or the user's secret.

During the verification phase, the holder employs a ZKP to demonstrate possession of the credential bound to the secret and its corresponding valid signature. This provides the verifier with cryptographic assurance that the signature is authentic and securely bound to the undisclosed credential, enabling authorization without exposing the underlying attributes.

Finally, to address the revocation problem mentioned earlier without breaking anonymity, these systems often integrate tools like *cryptographic accumulators* or zero-knowledge revocation registries. This allows the holder to prove via a ZKP that their credential has not been revoked by the issuer, maintaining privacy while preventing DoS or Sybil attacks.

1.3 State of the Art and Post-Quantum Cryptography

In the landscape of classical cryptography, the state of the art for anonymous credentials relies on highly efficient signature schemes. The most prominent models are BBS+ [ASM06], based on the q-Strong Diffie-Hellman (q-SDH) assumption; Pointcheval-Sanders (PS) [PS16], secure under a variant of the Diffie-Hellman assumption; and Camenisch-Lysyanskaya (CL) [CL02], which relies on the Strong RSA assumption. BBS+ and PS are particularly widely adopted due to their efficient algebraic structure, which enables the generation of short and highly compact proofs.

However, the security of these classical assumptions is fundamentally tied to the hardness of the Integer Factorization and Discrete Logarithm problems. Mathematically, these reduce to the task of finding the order of an element in a finite group (an instance of the Abelian Hidden Subgroup Problem). While strictly intractable for classical computing architectures, this barrier falls entirely with the advent of quantum computers; in fact, Shor's algorithm, introduced in 1994, can solve these exact problems in polynomial time. Consequently, once large-scale quantum computers become practically viable, these classical cryptographic protocols will be completely compromised.

To mitigate this existential threat, the cryptographic community is actively transitioning towards Post-Quantum Cryptography, exploring new mathematical foundations that resist quantum attacks. The most promising techniques rely on Coding Theory and lattice-based problems. This thesis will explore state-of-the-art methodologies for constructing ZKP protocols using post-quantum lattice techniques, ultimately demonstrating how these can be applied to design a secure, post-quantum Anonymous Credential protocol.

Thesis Outline.

The remainder of this thesis is organized as follows:

- **Chapter 2** introduces the mathematical preliminaries, notations, and core cryptographic assumptions that form the foundation of the protocols presented.
- **Chapter 3** establishes the theoretical framework of an ℓ_2-norm-based Zero-Knowledge Proof [LNP22]. It systematically explores the ABDLOP hybrid commitment scheme, the interactive mechanics for proving linear and quadratic relations, and the Fiat-Shamir transformation required to compile the system into a Non-Interactive Zero-Knowledge (NIZK) architecture.
- **Chapter 4** bridges the gap between theory and application by detailing the construction of a post-quantum anonymous credential system [BNLS23]. It defines the underlying ISIS_f assumptions, formalizes the cryptographic mechanics of the credential Issuance and Verification phases.
- **Chapter 5** provides a comprehensive comparative analysis and algorithmic optimization of the implemented protocols. It introduces a novel, highly efficient Greedy decomposition algorithm to manage norm inequalities, evaluates the mathematical trade-offs between different rejection sampling strategies, and rigorously validates the theoretical bounds against the concrete parameters instantiated by the LINKS Foundation.
- **Chapter 6** concludes the thesis by briefly summarizing the main results and the practical efficiency gained through the proposed optimizations.

Chapter 2

Mathematical Preliminaries

This section introduces the notation used throughout the thesis and formalizes the fundamental mathematical concepts and computational assumptions that form the basis of the cryptographic protocols presented.

2.1 Notation

Let λ be the security parameter, provided in unary as 1^λ to all involved algorithms. Let $\mathbb{Z}_p$ denote the ring of integers modulo p. Let $q = q_1 \cdot \ldots \cdot q_n$ be a product of n odd primes where $q_1 < q_2 < \cdots < q_n$ (usually $n = 1$ or $n = 2$).

To distinguish between different algebraic structures, arrow notation (e.g., $\vec{v} \in \mathbb{Z}_q^m$) is exclusively used to denote column vectors whose elements are scalars (integers or integers modulo q). Matrices over $\mathbb{Z}_q$ are correspondingly denoted by regular capital letters (e.g., A). By default, all vectors are column vectors. The concatenation of two scalar vectors $\vec{v}$ and $\vec{w}$ is indicated by $\vec{v} \parallel \vec{w}$. For $\vec{v}, \vec{w} \in \mathbb{Z}_q^k$, the operation $\vec{v} \circ \vec{w}$ represents the usual component-wise (Hadamard) multiplication. For simplicity, the notation $\vec{u}^2 = \vec{u} \circ \vec{u}$ is adopted.

The notation $x \leftarrow S$ indicates that $x \in S$ is sampled uniformly at random from a finite set S, and similarly $x \leftarrow D$ signifies that x is sampled according to a distribution D. For $n \in \mathbb{N}$, let $[n] := \{1, \ldots, n\}$.

For a power of two d and a positive integer p, the polynomial quotient rings $\mathbb{Z}[X]/(X^d+1)$ and $\mathbb{Z}_p[X]/(X^d+1)$ are denoted by $\mathcal{R}$ and $\mathcal{R}_p$, respectively. Lowercase letters denote single polynomial elements in $\mathcal{R}$ or $\mathcal{R}_p$. Bold lowercase (resp. uppercase) letters, such as $\mathbf{x}$ (resp. $\mathbf{A}$), are strictly reserved to represent column vectors (resp. matrices) containing polynomial elements over $\mathcal{R}$ or $\mathcal{R}_p$.

To explicitly denote the scalar coefficient vector of a polynomial $f \in \mathcal{R}_p$, we naturally apply the arrow notation, writing $\vec{f} \in \mathbb{Z}_p^d$. This convention identically extends to polynomial vectors: the integer coefficient expansion of a polynomial vector $\mathbf{x} \in \mathcal{R}_p^l$ is denoted as $\vec{x} \in \mathbb{Z}_p^{ld}$. By default, the i-th coefficient of a polynomial is written using the corresponding

regular font letter with a subscript i; e.g., $f_{d/2} \in \mathbb{Z}_p$ is the coefficient corresponding to $X^{d/2}$ of $f \in \mathcal{R}_p$. For the constant coefficient, however, the notation $\tilde{f} := f_0 \in \mathbb{Z}_p$ is uniquely reserved.

2.2 Coefficient Representation and Multiplication Matrices

For a polynomial vector $\mathbf{x} \in \mathcal{R}^l$, $\mathsf{Coeffs}(\mathbf{x}) \in \mathbb{Z}^{ld}$ is defined as the coefficient vector of $\mathbf{x}$. Similarly, $\mathsf{Coeffs}^{-1}(\vec{x}) \in \mathcal{R}^l$ represents the polynomial vector with coefficients $\vec{x}$.

For a polynomial $f = f_0 + f_1 X + \cdots + f_{d-1} X^{d-1} \in \mathcal{R}$, the *multiplication matrix* $\mathsf{rot}(f) \in \mathbb{Z}^{d \times d}$ is defined as the anti-circulant matrix:

$$\mathsf{rot}(f) = \begin{bmatrix} f_0 & -f_{d-1} & \cdots & -f_1 \\ f_1 & f_0 & \cdots & -f_2 \\ \vdots & \vdots & \ddots & \vdots \\ f_{d-1} & f_{d-2} & \cdots & f_0 \end{bmatrix} .$$

In particular, it holds that for $f, g \in \mathcal{R}$, $\mathsf{rot}(fg) = \mathsf{rot}(f)\mathsf{rot}(g)$. This definition is naturally extended to matrices over $\mathcal{R}$. Namely, for a matrix $\mathbf{F} = (f_{i,j}) \in \mathcal{R}_q^{n \times m}$, the matrix $\mathsf{rot}(\mathbf{F})$ is defined as a block matrix:

$$\mathsf{rot}(\mathbf{F}) = \begin{bmatrix} \mathsf{rot}(f_{1,1}) & \mathsf{rot}(f_{1,2}) & \cdots & \mathsf{rot}(f_{1,m}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{rot}(f_{n,1}) & \mathsf{rot}(f_{n,2}) & \cdots & \mathsf{rot}(f_{n,m}) \end{bmatrix} \in \mathbb{Z}^{nd \times md} .$$

Then, for any polynomial vector $\mathbf{x} \in \mathcal{R}_q^m$, the following property holds over $\mathbb{Z}_q$:

$$\mathsf{Coeffs}(\mathbf{F}\mathbf{x}) = \mathsf{rot}(\mathbf{F})\mathsf{Coeffs}(\mathbf{x}) .$$

2.3 Lattices

A lattice Λ is a discrete subgroup of $\mathbb{R}^n$. Given a set of linearly independent vectors $B = \{\vec{b}_1, \ldots, \vec{b}_n\} \subseteq \mathbb{R}^n$, the n-dimensional full-rank lattice generated by B is defined as follows:

$$\Lambda = \Lambda(B) := \left\{ \sum_{i=1}^{n} c_i \vec{b}_i : c_1, \ldots, c_n \in \mathbb{Z} \right\} . \tag{2.1}$$

An ideal lattice is isomorphic to an ideal within a polynomial quotient ring under the coefficient embedding. While $\mathcal{R}$ naturally embeds into a lattice, the quotient ring $\mathcal{R}_q$ is a finite ring and thus not a lattice itself. However, cryptographic constructions often operate

over *q-ary lattices*, which are defined as the full-rank lattices obtained by taking the pre-image in $\mathcal{R}$ of modules defined over $\mathcal{R}_q$. Within these mathematical structures exist well-known computationally hard problems that can be exploited to build secure cryptographic protocols, most notably the Module-SIS (M-SIS) and Module-LWE (M-LWE) problems.

2.4 Module-SIS and Module-LWE Problems

The post-quantum security of the proposed schemes relies on the well-known computational lattice problems Module-LWE (MLWE) and Module-SIS (MSIS) [LS15]. Both problems are typically defined over the ring $\mathcal{R}_q$. When bounding the norm of elements in $\mathcal{R}_q$, it is implicitly assumed that their coefficients are uniquely lifted to the representative interval $(-q/2, q/2] \subset \mathbb{Z}$, and the norm is evaluated on the resulting integer vector.

Definition 2.4.1 ($\mathsf{MSIS}_{n,m,\mathcal{B},q}$)**.** Given a uniformly random matrix $\mathbf{A} \leftarrow \mathcal{R}_q^{n\times m}$, the Module-SIS problem with parameters $m > n > 0$ and a norm bound $0 < \mathcal{B} < q$ asks to find a non-zero vector $\mathbf{z} \in \mathcal{R}_q^m$ such that $\mathbf{Az} = \mathbf{0}$ over $\mathcal{R}_q$ and $0 < \|\mathbf{z}\| \leq \mathcal{B}$.

An algorithm (or adversary) $\mathcal{A}$ is said to have advantage ϵ in solving the $\mathsf{MSIS}_{n,m,\mathcal{B},q}$ problem if:

$$\mathbf{Adv}^{\mathsf{MSIS}}_{n,m,\mathcal{B},q}(\mathcal{A}) := \Pr\left[\mathbf{Az} = \mathbf{0} \wedge 0 < \|\mathbf{z}\| \leq \mathcal{B} \mid \mathbf{A} \leftarrow \mathcal{R}_q^{n\times m}; \mathbf{z} \leftarrow \mathcal{A}(\mathbf{A})\right] \geq \epsilon\,. \tag{2.2}$$

The problem is considered computationally hard if, for all probabilistic polynomial-time (PPT) adversaries $\mathcal{A}$, the advantage $\mathbf{Adv}^{\mathsf{MSIS}}(\mathcal{A})$ is negligible in the security parameter λ.

Definition 2.4.2 ($\mathsf{MLWE}_{m,n,\chi,q}$)**.** The Module-LWE problem with parameters $m \geq n > 0$ and an error distribution χ over $\mathcal{R}$ asks an adversary $\mathcal{A}$ to distinguish between the following two cases: 1) a valid LWE sample $(\mathbf{A}, \mathbf{As}+\mathbf{e})$, where $\mathbf{A} \leftarrow U(\mathcal{R}_q^{m\times n})$, a secret vector $\mathbf{s} \leftarrow \chi^n$, and an error vector $\mathbf{e} \leftarrow \chi^m$; and 2) a uniformly random sample $(\mathbf{A}, \mathbf{b}) \leftarrow \mathcal{R}_q^{m\times n} \times \mathcal{R}_q^m$.

An adversary $\mathcal{A}$ is said to have advantage ϵ in solving the $\mathsf{MLWE}_{m,n,\chi,q}$ problem if its distinguishing advantage is:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{MLWE}}_{m,n,\chi,q}(\mathcal{A}) := \Bigg| &\Pr\left[1 \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{As}+\mathbf{e}) \mid \mathbf{A} \leftarrow \mathcal{R}_q^{m\times n}; \mathbf{s} \leftarrow \chi^n; \mathbf{e} \leftarrow \chi^m\right] - \\ &\Pr\left[1 \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{b}) \mid \mathbf{A} \leftarrow \mathcal{R}_q^{m\times n}; \mathbf{b} \leftarrow \mathcal{R}_q^m\right]\Bigg| \geq \epsilon\,.\end{aligned} \tag{2.3}$$

Similarly, the problem is considered computationally hard if, for all PPT adversaries $\mathcal{A}$, the distinguishing advantage $\mathbf{Adv}^{\mathsf{MLWE}}(\mathcal{A})$ is negligible in the security parameter λ.

2.5 Cryptographic Primitives

This section formalizes the foundational cryptographic primitives and the interaction models utilized throughout the construction of the protocols presented in this thesis.

2.5.1 Cryptographic Hash Functions

A cryptographic hash function is a deterministic algorithm $H : \{0,1\}^* \to \{0,1\}^\lambda$ that maps an arbitrary-length input to a fixed-size output, often referred to as a digest. To be cryptographically secure, a hash function must satisfy three primary properties:

- **Pre-image Resistance:** Given a hash value h, it is computationally infeasible to find an input m such that $H(m) = h$.
- **Second Pre-image Resistance:** Given an input m_1, it is computationally infeasible to find a different input $m_2 \neq m_1$ such that $H(m_1) = H(m_2)$.
- **Collision Resistance:** It is computationally infeasible to find any two distinct inputs $m_1 \neq m_2$ that produce the same hash output, i.e., $H(m_1) = H(m_2)$.

In the security analysis of advanced interactive protocols and their non-interactive variants, hash functions are frequently modeled as *Random Oracles*. The Random Oracle Model (ROM) assumes the existence of a perfectly random function accessible to all parties, which outputs a uniformly random response from its codomain for every new query and returns the identical response for repeated queries.

2.5.2 Commitment Schemes

A cryptographic commitment scheme is a two-phase protocol that allows a party to commit to a chosen value while keeping it hidden from others, retaining the ability to reveal the committed value at a later stage. Formally, a commitment scheme consists of two main algorithms:

1. **Commit Phase:** The sender chooses a message m and samples a random value r. The commitment is computed as $c \leftarrow \mathsf{Com}(m; r)$ and sent to the receiver.
2. **Open Phase:** The sender reveals m and the randomness r. The receiver accepts if and only if $\mathsf{Com}(m; r) = c$.

A secure commitment scheme must satisfy two fundamental properties:

- **Hiding:** The commitment c reveals no information about the underlying message m to the receiver.
- **Binding:** Once c is published, the sender cannot change the message. It is computationally infeasible to find another message $m' \neq m$ and randomness r' such that $\mathsf{Com}(m'; r') = \mathsf{Com}(m; r) = c$.

2.5.3 Modeling Interactive Protocols

An interactive protocol is mathematically modeled as an exchange of messages between two probabilistic polynomial-time (PPT) Turing machines: a Prover $\mathcal{P}$ and a Verifier $\mathcal{V}$. Both parties have access to a common public input x (the statement to be proven), while the Prover typically holds a private witness w.

The execution of the protocol generates a sequence of messages known as the *transcript*, denoted by $\tau = (m_1, m_2, \ldots, m_k)$. At the end of the interaction, the Verifier outputs a binary decision: 1 (accept) or 0 (reject), denoted as $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle \in \{0, 1\}$.

The interactive protocol can be transformed into a Non-Interactive Zero-Knowledge (NIZK) proof using the Fiat-Shamir transformation, where the Verifier's random challenges are replaced by the output of a cryptographic hash function evaluated over the transcript generated up to that point.

2.6 Probability Distributions

This subsection formalizes the probability distributions utilized for sampling vectors within the cryptographic schemes, specifically focusing on the discrete Gaussian distribution and the centered binomial distribution.

Definition 2.6.1 (Discrete Gaussian Distribution)**.** The discrete Gaussian distribution over $\mathcal{R}^m$, centered around a vector $\mathbf{v} \in \mathcal{R}^m$ and with standard deviation $s > 0$, is defined as:

$$D^m_{\mathbf{v},s}(\mathbf{z}) = \frac{e^{-\|\mathbf{z}-\mathbf{v}\|^2/(2s^2)}}{\sum_{\mathbf{z}' \in \mathcal{R}^m} e^{-\|\mathbf{z}'-\mathbf{v}\|^2/(2s^2)}}\,, \tag{2.4}$$

where the norm $\|\cdot\|$ evaluated on a polynomial vector denotes the Euclidean norm (ℓ_2-norm) of its coefficient vector. When the distribution is centered around the zero vector $\mathbf{0} \in \mathcal{R}^m$, it is simply denoted as $D^m_s := D^m_{\mathbf{0},s}$.

To analyze the probability of sampling elements with a large norm, the following tail bound, originally derived by Banaszczyk [Ban93], is frequently employed.

Lemma 2.6.1 (Gaussian Tail Bound)**.** Let $\mathbf{z} \leftarrow D^m_s$. Then, for any $t > 1$, the following bound holds:

$$\Pr\left[\|\mathbf{z}\| > t \cdot s\sqrt{md}\right] \leq \left(te^{\frac{1-t^2}{2}}\right)^{md}\,. \tag{2.5}$$

Definition 2.6.2 (Centered Binomial Distribution)**.** The centered binomial distribution with a positive integer parameter κ, denoted as Bin_κ, is defined as the distribution of the sum $\sum_{i=1}^{\kappa}(a_i - b_i)$, where the elements $a_i, b_i \leftarrow \{0, 1\}$ are sampled uniformly at random. The expected variance of this distribution is $\kappa/2$. Furthermore, given two independent random variables $X \sim \mathsf{Bin}_{\kappa_1}$ and $Y \sim \mathsf{Bin}_{\kappa_2}$, their sum or difference follows exactly the distribution:

$$X \pm Y \sim \mathsf{Bin}_{\kappa_1+\kappa_2}\,. \tag{2.6}$$

Chapter 3

Framework of an ℓ_2-Norm-Based ZKP

Chapter Outline.

This chapter describes a lattice-based ZKP protocol that relies on the MSIS problem evaluated under ℓ_2-norm bounds. This theoretical framework, introduced by Lyubashevsky, Nguyen, and Plançon [LNP22] (hereafter referred to as the LNP protocol), serves as the mathematical foundation from which our anonymous credential model will be constructed.

To systematically break down the mathematical and cryptographic complexity of this framework, the remainder of this chapter is structured as follows:

- **Section 3.1** reviews the state of the art in MSIS-based Zero-Knowledge Proofs, contextualizing the efficiency of the LNP framework against existing combinatorial and algebraic approaches.
- **Section 3.2** introduces the ABDLOP hybrid commitment scheme, which securely binds the secret data by combining the structural properties of the Ajtai and BDLOP schemes.
- **Section 3.3** constructs the foundational interactive protocol, defining the rejection sampling algorithms necessary to maintain zero-knowledge and the extraction mechanics for linear relations.
- **Section 3.4** explains an efficient method to represent the ℓ_2-norm evaluation and how to mask it to guarantee zero-knowledge properties.
- **Section 3.5** constructs a sub-protocol to mathematically prove the validity of the underlying quadratic equations.
- **Section 3.6** details how to prevent modular wrap-around and transition from an equality to an inequality protocol, employing approximate range proofs and binary slack variables.

- **Section 3.7** provides a comprehensive, step-by-step synthesis and visual representation of the fully assembled interactive protocol.
- **Section 3.8** explains the challenge space structure and how to evaluate the standard deviations required for the rejection sampling methods.
- **Section 3.9** establishes the formal security guarantees and presents a concrete parameter instantiation.
- **Section 3.10** concludes the chapter by applying the Fiat-Shamir transformation, compiling the interactive system into the final Non-Interactive Zero-Knowledge (NIZK) architecture required for practical credential deployment.

3.1 State of the Art

Several ZKP protocols based on the MSIS problem have been proposed in recent years. Their core objective is to demonstrate that a prover possesses a short secret vector $\mathbf{s}$ satisfying $\mathbf{As} = \mathbf{t} \pmod{q}$, without revealing any information about $\mathbf{s}$. Historically, this problem has been addressed using two primary approaches:

- **Combinatorial Approach (Stern-based):** Originally based on Stern's algorithm [Ste93], this approach proves knowledge of $\mathbf{s}$ by revealing a random permutation of the secret vector. However, it inherently suffers from a high soundness error (typically 2/3), which requires an elevated number of protocol repetitions to achieve adequate security. This dramatically increases the proof size, often exceeding 1 MB. Subsequent improvements, such as the application of generic *cut-and-choose* techniques by Beullens [Beu20], have successfully reduced the proof dimension to approximately 200 KB.
- **Algebraic Approach (Lattice Commitments):** This more recent and efficient paradigm relies on lattice-based commitments combined with ZKPs over the committed values [BLS19, YAZ+19, ESLL19]. To bound the ℓ_∞ norm of $\mathbf{s}$ (assuming it contains small coefficients, such as $\{-1, 0, 1\}$), the prover commits to a polynomial m whose Chinese Remainder Theorem (CRT) coefficients represent $\mathbf{s}$. The protocol then proves the required linear relations alongside the polynomial identity $(m-1) \cdot m \cdot (m+1) = 0$ [ALS20, ENS20]. Recent optimizations in proving polynomial products have drastically decreased the proof size to about 33 KB for standard statements [LNS21a].

  Despite its efficiency, this approach has notable limitations. First, the polynomial m inherently requires large coefficients, necessitating heavier commitments. Second, if the vector $\mathbf{s}$ contains slightly larger coefficients (as is common in trapdoor sampling operations), the degree of the polynomial product increases significantly, causing the proof size to explode.

The LNP protocol [LNP22] presented in this thesis overcomes these limitations. It introduces a novel mechanism to directly prove a tight bound based on the ℓ_2 norm of the secret vector, entirely bypassing the inefficiencies associated with ℓ_∞-norm bounds and CRT embeddings. Specifically, the protocol utilizes an MSIS-based commitment scheme combined with ZKPs for both linear and quadratic relations, efficiently verifying the vector's short norm.

3.2 Committing to the Module-SIS Secret

To formally bind the secret vector $\mathbf{s}$ that satisfies the underlying Module-SIS relation, the protocol proposed in the LNP paper [LNP22] introduces a novel hybrid commitment scheme. This construction combines the mechanisms of two distinct commitment algorithms: the BDLOP scheme and the Ajtai scheme.

3.2.1 Ajtai Commitment Scheme

The Ajtai commitment scheme [Ajt96] relies directly on the computational hardness of the Module-SIS problem. It allows a sender to hide a secret message vector $\mathbf{s}_1$ using a randomly sampled masking vector $\mathbf{s}_2$, provided that both vectors possess a sufficiently small norm. Given two uniformly random public matrices $\mathbf{A}_1$ and $\mathbf{A}_2$, the commitment $\mathbf{t}$ is computed as:

$$\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2 = \mathbf{t} \pmod{q}. \tag{3.1}$$

The *binding* property of the commitment scheme is intrinsically linked to the MSIS assumption. If a malicious sender were able to open the commitment to a different message, it would imply finding a distinct pair of small vectors $(\mathbf{s}_1', \mathbf{s}_2') \neq (\mathbf{s}_1, \mathbf{s}_2)$ such that:

$$\mathbf{A}_1\mathbf{s}_1' + \mathbf{A}_2\mathbf{s}_2' = \mathbf{t} \pmod{q}. \tag{3.2}$$

By subtracting this forged opening from the original valid commitment, the following relation is obtained:

$$\mathbf{A}_1(\mathbf{s}_1 - \mathbf{s}_1') + \mathbf{A}_2(\mathbf{s}_2 - \mathbf{s}_2') = \mathbf{0} \pmod{q}. \tag{3.3}$$

Let $\mathbf{A} := [\mathbf{A}_1 \parallel \mathbf{A}_2]$ be the concatenation of the public matrices. By defining the combined difference vector $\mathbf{z}$ in a vertical block structure as:

$$\mathbf{z} := \begin{bmatrix} \mathbf{s}_1 - \mathbf{s}_1' \\ \mathbf{s}_2 - \mathbf{s}_2' \end{bmatrix}, \tag{3.4}$$

the relation can be equivalently rewritten in the canonical MSIS form:

$$\mathbf{A}\mathbf{z} = \mathbf{0} \pmod{q}. \tag{3.5}$$

Since the initial vectors $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_1', \mathbf{s}_2'$ all have small norms by definition, their difference $\mathbf{z}$ is also a non-zero vector with a small bounded norm. Finding such a vector $\mathbf{z}$ is equivalent to solving the MSIS problem. Therefore, assuming MSIS is computationally hard, breaking the binding property of the Ajtai commitment is intractable.

Conversely, the *hiding* property is statistical rather than computational. According to the Leftover Hash Lemma (LHL) [HILL99], as long as the masking vector $\mathbf{s}_2$ is sufficiently large and chosen with enough randomness, the result of $\mathbf{A}_2\mathbf{s}_2 \pmod q$ becomes statistically indistinguishable from a purely random distribution. Consequently, this term acts as a perfectly random mask, completely hiding the original message $\mathbf{A}_1\mathbf{s}_1$ even from an adversary with infinite computing power.

A significant advantage of the Ajtai scheme is its compactness. Due to its matrix structure, the size of the commitment $\mathbf{t}$ remains fixed and independent of the length of the message. However, a major limitation is that the scheme can only be securely applied to hide message vectors $\mathbf{s}_1$ that inherently possess a small norm, precluding its use for committing to arbitrary long or large-norm strings.

3.2.2 BDLOP Commitment Scheme

An alternative approach is the BDLOP commitment scheme, introduced by Baum et al. [BDL+18]. This scheme permits committing to a message vector $\mathbf{m}$, with no restriction on the size of its coefficients, by hiding it within a pseudo-random distribution.

The commitment involves a shared public matrix $\mathbf{A}'$ and a message-specific public matrix $\mathbf{B}'$. The prover samples a small-norm secret vector $\mathbf{s}$ and two independent small-norm error vectors, $\mathbf{e}_A$ and $\mathbf{e}_B$. The expanded commitment is computed as:

$$\begin{aligned} \mathbf{t}_A &= \mathbf{A}'\mathbf{s} + \mathbf{e}_A \pmod q, \\ \mathbf{t}_B &= \mathbf{B}'\mathbf{s} + \mathbf{e}_B + \mathbf{m} \pmod q. \end{aligned} \tag{3.6}$$

The *binding* property is guaranteed by the MSIS assumption. Finding two distinct valid openings for the same commitment mathematically reduces to finding a short vector in the kernel of the public matrices, directly violating the hardness of MSIS.

Conversely, the *hiding* property relies strictly on the MLWE assumption. As seen in Equation 3.6, the components $\mathbf{A}'\mathbf{s} + \mathbf{e}_A$ and $\mathbf{B}'\mathbf{s} + \mathbf{e}_B$ are canonical MLWE samples. The MLWE assumption formally states that such samples are computationally indistinguishable from perfectly uniform vectors over $\mathcal{R}_q$. Consequently, the message $\mathbf{m}$ is securely masked by what appears to be pure randomness.

The BDLOP scheme presents two significant advantages. First, since $\mathbf{m}$ is simply added in the clear to a pseudo-random mask, its coefficients can be arbitrarily large modulo q without affecting the norm bounds required for security. Second, the expanded structure highlights the scheme's highly efficient *amortization* capabilities. The base component $\mathbf{t}_A$ effectively locks the secret $\mathbf{s}$. To commit to k additional messages, the prover does not need

to re-evaluate new base commitments; they simply append new rows:

$$\mathbf{t}_{B_i} = \mathbf{B}_i'\mathbf{s} + \mathbf{e}_{B_i} + \mathbf{m}_i \pmod q \quad \text{for } i = 1, \ldots, k\,. \tag{3.7}$$

As long as independent error terms $\mathbf{e}_{B_i}$ are drawn for each new message, the MLWE assumption holds. This implies that committing to additional elements requires only a marginal increase in the overall proof size.

Block Matrix Formulation and Compact Notation.

To achieve a more compact notation, identity matrices ($\mathbf{I}$) and zero matrices ($\mathbf{0}$) can be embedded within the public parameters. By concatenating the secret and the errors into a single short vector $\mathbf{s}_2 := \begin{bmatrix} \mathbf{s} \\ \mathbf{e}_A \\ \mathbf{e}_B \end{bmatrix}$, the entire BDLOP commitment from Equation 3.6 can be rewritten using explicit block matrix multiplication:

$$\begin{bmatrix} \mathbf{A}' & \mathbf{I} & \mathbf{0} \\ \mathbf{B}' & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{s} \\ \mathbf{e}_A \\ \mathbf{e}_B \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{m} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_A \\ \mathbf{t}_B \end{bmatrix} \pmod q\,. \tag{3.8}$$

By defining the structured public matrices as $\mathbf{A} := [\mathbf{A}' \parallel \mathbf{I} \parallel \mathbf{0}]$ and $\mathbf{B} := [\mathbf{B}' \parallel \mathbf{0} \parallel \mathbf{I}]$, the formulation is compacted into:

$$\begin{bmatrix} \mathbf{A} \\ \mathbf{B} \end{bmatrix} \mathbf{s}_2 + \begin{bmatrix} \mathbf{0} \\ \mathbf{m} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_A \\ \mathbf{t}_B \end{bmatrix} \pmod q\,. \tag{3.9}$$

For the remainder of this thesis, this compact notation will be adopted, with the strict understanding that $\mathbf{s}_2$ intrinsically contains the required independent error terms to satisfy MLWE.

3.2.3 The ABDLOP Hybrid Commitment Scheme

In the LNP protocol [LNP22], the authors introduce a hybrid commitment scheme, informally referred to as ABDLOP, which combines the structural advantages of both the Ajtai and BDLOP constructions. This scheme partitions the secret data into two distinct components: a vector $\mathbf{s}_1$ containing the elements that inherently possess a small norm, and a vector $\mathbf{m}$ containing the remaining elements (with potentially large coefficients) required to complete the protocol.

The security of this mixed commitment relies on both the MSIS and MLWE assumptions. It is formulated as follows:

$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{0} \end{bmatrix} \mathbf{s}_1 + \begin{bmatrix} \mathbf{A}_2 \\ \mathbf{B} \end{bmatrix} \mathbf{s}_2 + \begin{bmatrix} \mathbf{0} \\ \mathbf{m} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_A \\ \mathbf{t}_B \end{bmatrix} \pmod q\,. \tag{3.10}$$

The *binding* property is guaranteed by the layered structure of this construction. Specifically, examining the upper block of Equation 3.10 yields exactly the Ajtai commitment:

$$\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2 = \mathbf{t}_A \pmod{q}. \tag{3.11}$$

Consequently, finding an alternative valid opening $(\mathbf{s}_1', \mathbf{s}_2') \neq (\mathbf{s}_1, \mathbf{s}_2)$ is computationally as hard as solving the MSIS problem. Furthermore, the binding of the message $\mathbf{m}$ is linked to this upper block. Because the randomness vector $\mathbf{s}_2$ is tightly bound by the MSIS assumption in the first equation, the lower equation, $\mathbf{B}\mathbf{s}_2 + \mathbf{m} = \mathbf{t}_B \pmod{q}$, uniquely determines $\mathbf{m}$. Any attempt to alter $\mathbf{m}$ would require a corresponding forged change in $\mathbf{s}_2$, which is computationally infeasible.

The *hiding* property is guaranteed, similarly to the pure BDLOP scheme, by the MLWE assumption. As established in the previous formulation, the randomness vector $\mathbf{s}_2$ implicitly contains the necessary MLWE error terms. The commitment securely hides both $\mathbf{s}_1$ and $\mathbf{m}$ because, under the MLWE assumption, the masking term $\begin{bmatrix}\mathbf{A}_2 \\ \mathbf{B}\end{bmatrix}\mathbf{s}_2$ is computationally indistinguishable from a perfectly uniform random vector over $\mathcal{R}_q$. As a result, this pseudo-random vector perfectly obfuscates the underlying secret values.

3.3 Building an Interactive Protocol for the ABDLOP Commitment

In this section, we describe how to practically deploy the ABDLOP commitment scheme within a ZKP framework. Specifically, we construct an interactive protocol based on the *Fiat-Shamir with Aborts* paradigm [Lyu09], defining the necessary cryptographic checks required to guarantee its security.

The fundamental goal of this protocol is to prove knowledge of the underlying secret data (which we denote generally as a single vector $\mathbf{s}$ for simplicity) that satisfies the commitment relation, without revealing any information about its exact algebraic structure to the verifier. The protocol follows a canonical three-move Σ-protocol structure:

1. **Commitment:** To prevent the final response from leaking information about the secret, the prover first samples a random masking vector $\mathbf{y}$ from a discrete Gaussian distribution $D_{\mathfrak{s}}$ centered at zero with standard deviation $\mathfrak{s}$. The prover computes and sends a public commitment of this mask to the verifier.

2. **Challenge:** Upon receiving the mask commitment, the verifier responds with a random challenge c, sampled from a predefined challenge space $\mathcal{C} \subset \mathcal{R}_q$.

3. **Response:** The prover linearly combines the secret, the challenge, and the mask, generating the response vector $\mathbf{z} = c\mathbf{s} + \mathbf{y}$.

The structural properties of the challenge space $\mathcal{C}$ are critical for the *Soundness* of the protocol. They will be formally analyzed in Section 3.8.1.

In the following subsections, we will detail how this interactive sequence is structured to formally validate the ABDLOP commitment. We will explore how the *Zero-Knowledge* property is preserved through rejection sampling (Section 3.3.2), and how *Soundness* is geometrically guaranteed (Section 3.3.3).

3.3.1 Validation of the ABDLOP Commitment

To validate that the commitment is satisfied, the protocol must establish a verification framework in which the masked responses act as valid solutions. In the ABDLOP commitment scheme, the prover holds three secret components: $\mathbf{s}_1$, $\mathbf{s}_2$, and $\mathbf{m}$. However, given the public matrix $\mathbf{B}$ and the commitment vector $\mathbf{t}_B$, the message $\mathbf{m}$ is algebraically fixed and can be uniquely expressed as $\mathbf{m} = \mathbf{t}_B - \mathbf{B}\mathbf{s}_2 \pmod q$.

Consequently, the only terms required to explicitly open the commitment are $\mathbf{s}_1$ and $\mathbf{s}_2$. Furthermore, to guarantee the security of the protocol, it must be verified that both $\mathbf{s}_1$ and $\mathbf{s}_2$ possess a small norm. Thus, only $\mathbf{s}_1$ and $\mathbf{s}_2$ need to be masked by random vectors $\mathbf{y}_1$ and $\mathbf{y}_2$, resulting in the response vectors $\mathbf{z}_1$ and $\mathbf{z}_2$.

By defining the masked responses as $\mathbf{z}_1 = c\mathbf{s}_1 + \mathbf{y}_1$ and $\mathbf{z}_2 = c\mathbf{s}_2 + \mathbf{y}_2$, these elements can be substituted into the upper block of the commitment equation:

$$\begin{aligned} \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 &= \mathbf{A}_1(c\mathbf{s}_1 + \mathbf{y}_1) + \mathbf{A}_2(c\mathbf{s}_2 + \mathbf{y}_2) \\ &= c\mathbf{A}_1\mathbf{s}_1 + c\mathbf{A}_2\mathbf{s}_2 + \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 \\ &= c(\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2) + \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 \\ &= c\mathbf{t}_A + \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 \pmod q. \end{aligned} \tag{3.12}$$

To ensure the *Soundness* of this interactive protocol, it is imperative that the prover commits to the masking vectors $\mathbf{y}_1$ and $\mathbf{y}_2$ before learning the challenge c. If this chronology were not enforced, a malicious prover could trivially cheat by forging values that satisfy Equation 3.12 without actually knowing the underlying secrets. This strict interactive sequence guarantees the *Proof of Knowledge* property (formally proven by the existence of a Knowledge Extractor).

The interactive protocol is structured as follows:

1. The prover samples the masking vectors $\mathbf{y}_1$ and $\mathbf{y}_2$. It evaluates the commitment $\mathbf{w} = \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 \pmod q$ and sends $\mathbf{w}$ to the verifier.

2. The verifier samples a random challenge c from the challenge space $\mathcal{C}$ and sends it to the prover.

3. The prover computes the responses $\mathbf{z}_1$ and $\mathbf{z}_2$. To ensure these responses do not leak any statistical information regarding the geometry of the secrets, the prover performs

a *rejection sampling* check (as detailed in Section 3.3.2). If the check passes, $\mathbf{z}_1$ and $\mathbf{z}_2$ are transmitted to the verifier; otherwise, the protocol aborts.

4. Finally, the verifier checks whether $\mathbf{z}_1$ and $\mathbf{z}_2$ satisfy the accepted small-norm bounds (see Section 3.3.3) and verifies the algebraic relation:

$$\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = c\mathbf{t}_A + \mathbf{w} \pmod{q}.$$

If both checks pass, the verifier officially accepts the proof of knowledge for the commitment of $\mathbf{s}_1$ and $\mathbf{m}$.

3.3.2 Rejection Sampling

For simplicity in the following analysis, let $\mathbf{s}$ represent any generic secret component (such as $\mathbf{s}_1$ or $\mathbf{s}_2$) and $\mathbf{z}$ its corresponding masked response.

Masking a secret vector $\mathbf{s}$ by simply adding a noise vector $\mathbf{y}$ is insufficient to guarantee the Zero-Knowledge property. In a standard protocol execution, the masking vector $\mathbf{y}$ is sampled from a discrete Gaussian distribution centered at zero. Consequently, the expected value of the noise is mathematically zero ($\mathbb{E}[\mathbf{y}] = \mathbf{0}$).

When a prover computes the response $\mathbf{z} = c\mathbf{s} + \mathbf{y}$, they are geometrically shifting the center of the Gaussian distribution from zero to exactly $c\mathbf{s}$. Because the challenge c and the response $\mathbf{z}$ are transmitted in the clear, an attacker can collect multiple transcripts $(\mathbf{z}_1, c_1), (\mathbf{z}_2, c_2), \ldots, (\mathbf{z}_k, c_k)$ and perform a statistical correlation attack to retrieve the secret.

Given that the challenge space $\mathcal{C}$ is explicitly constructed to yield invertible challenges (the algebraic properties of which will be formally detailed in Section 3.8.1), an attacker can easily isolate the secret. By multiplying the intercepted response by the inverse of the challenge, the attacker obtains:

$$c_i^{-1}\mathbf{z}_i = c_i^{-1}(c_i\mathbf{s} + \mathbf{y}_i) = \mathbf{s} + c_i^{-1}\mathbf{y}_i. \tag{3.13}$$

By computing the expected value of this operation over k independent transcripts, and exploiting the linearity of expectations alongside the fact that $\mathbb{E}[\mathbf{y}_i] = \mathbf{0}$, the noise term cancels out:

$$\mathbb{E}[c_i^{-1}\mathbf{z}_i] = \mathbf{s} + c_i^{-1}\mathbb{E}[\mathbf{y}_i] = \mathbf{s}. \tag{3.14}$$

Given a sufficient number of transcripts, the noise term converges to zero due to the law of large numbers, allowing the attacker to easily compute the secret $\mathbf{s}$. This statistical leakage directly compromises the Zero-Knowledge property of the protocol.

To prevent this vulnerability, lattice-based protocols employ a *rejection sampling* technique, originally introduced by Lyubashevsky [Lyu09]. Before sending the response $\mathbf{z}$ to the verifier, the prover evaluates the probability that $\mathbf{z}$ originated from a discrete Gaussian distribution centered at zero, rather than the one shifted by $c\mathbf{s}$. If the vector $\mathbf{z}$ leaks

statistical information about the geometric shift, the prover aborts the execution, discards the values, and restarts the protocol by sampling a new masking vector $\mathbf{y}$ and requesting a new challenge c. By only transmitting responses that pass this probability filter, the prover ensures that the output distribution remains independent of the secret $\mathbf{s}$, thus preserving Zero-Knowledge.

The Rej_1 Rejection Algorithm

A fundamental approach utilized in the LNP framework [LNP22] relies on the baseline rejection sampling algorithm (often denoted as Rej_1), originally formalized by Lyubashevsky in 2012 [Lyu12].

This algorithm evaluates the probability that a shifted vector $\mathbf{z}$ could have been legitimately sampled from a zero-centered discrete Gaussian distribution. By defining the shift vector as $\mathbf{v} = c\mathbf{s}$, the acceptance probability is proportional to the ratio between the target distribution $D_{\mathfrak{s}}$ and the shifted source distribution $D_{\mathbf{v},\mathfrak{s}}$:

$$\frac{D_{\mathfrak{s}}(\mathbf{z})}{D_{\mathbf{v},\mathfrak{s}}(\mathbf{z})} = \frac{\exp\left(-\frac{\|\mathbf{z}\|^2}{2\mathfrak{s}^2}\right)}{\exp\left(-\frac{\|\mathbf{z}-\mathbf{v}\|^2}{2\mathfrak{s}^2}\right)} = \exp\left(\frac{-2\langle \mathbf{z}, \mathbf{v}\rangle + \|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right). \tag{3.15}$$

Since this ratio is not inherently bounded within (0,1], it cannot be directly used as a probability. To address this, a normalization constant M is introduced to represent the maximum possible value reached by this relation. Applying standard Gaussian tail bounds, the inner product $\langle \mathbf{z}, \mathbf{v}\rangle$ exceeds $14\mathfrak{s}\|\mathbf{v}\|$ only with negligible probability. Consequently, M is evaluated as:

$$M = \exp\left(\frac{28\mathfrak{s}\|\mathbf{v}\| + \|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right). \tag{3.16}$$

The final acceptance probability is then defined as:

$$P_{acc} = \frac{1}{M}\exp\left(\frac{-2\langle \mathbf{z}, \mathbf{v}\rangle + \|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right). \tag{3.17}$$

During execution, the rejection algorithm samples a uniformly distributed value $u \in [0,1)$ and checks whether $u > P_{acc}$. If this condition holds, the algorithm aborts the protocol; otherwise, the response is accepted.

As proven in [Lyu12], this algorithm outputs a valid transcript without leaking the secret, succeeding with an expected probability of $1/M$. Because M acts as the peak bound of Equation 3.15, a higher M severely diminishes the acceptance rate. To maintain M at a manageable level, the standard deviation $\mathfrak{s}$ applied to sample the masking vector $\mathbf{y}$ must be significantly increased. For example, to achieve an acceptance rate of approximately $1/3$ (requiring $M \approx 3$), the standard deviation must be set to $\mathfrak{s} \approx 13\|\mathbf{v}\|$. However, increasing $\mathfrak{s}$ inherently expands the magnitude of the sampled coefficients. This limits the effectiveness of algebraic compression techniques (such as the high-order bit dropping used in protocols

like Dilithium), directly increasing the overall proof size.

The Rej_0 (Bimodal) Rejection Algorithm

To overcome the severe inflation of the standard deviation required by Rej_1, an earlier and highly efficient technique known as Bimodal Rejection (Rej_0) was introduced by Ducas et al. in 2013 [DDLL13].

The protocol introduces a random sign bit $b \in \{-1,1\}$, modifying the computation of the masked response to:

$$\mathbf{z} = \mathbf{y} + b\mathbf{v}. \tag{3.18}$$

By introducing this random sign, the discrete Gaussian distribution is shifted in the positive direction half of the time, and in the negative direction the other half. The core idea is that combining these two shifted distributions creates a bimodal source distribution that remains perfectly symmetric around zero.

To evaluate the acceptance probability, we calculate the ratio between the unshifted target distribution $D_{\mathfrak{s}}$ and the bimodal source distribution:

$$\begin{aligned} \frac{D_{\mathfrak{s}}(\mathbf{z})}{\frac{1}{2}D_{\mathbf{v},\mathfrak{s}}(\mathbf{z}) + \frac{1}{2}D_{-\mathbf{v},\mathfrak{s}}(\mathbf{z})} &= \frac{\exp\left(-\frac{\|\mathbf{z}\|^2}{2\mathfrak{s}^2}\right)}{\frac{1}{2}\exp\left(-\frac{\|\mathbf{z}-\mathbf{v}\|^2}{2\mathfrak{s}^2}\right) + \frac{1}{2}\exp\left(-\frac{\|\mathbf{z}+\mathbf{v}\|^2}{2\mathfrak{s}^2}\right)} \\ &= \frac{\exp\left(\frac{\|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right)}{\cosh\left(\frac{\langle \mathbf{z},\mathbf{v}\rangle}{\mathfrak{s}^2}\right)}. \end{aligned} \tag{3.19}$$

Since the hyperbolic cosine function satisfies $\cosh(x) \geq 1$ for any real value x, the maximum value M of this ratio is strictly bounded by:

$$M = \exp\left(\frac{\|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right). \tag{3.20}$$

Maintaining a constant $M \approx 3$ with this bound only requires a standard deviation of $\mathfrak{s} \approx 0.675\|\mathbf{v}\|$. Furthermore, an important mathematical advantage of the bimodal approach [DDLL13] is that the expected acceptance probability is exactly $1/M$. Because the symmetric target distribution perfectly hides the sign, no information about the secret vector $\mathbf{v}$ is leaked.

Despite these significant benefits, the scheme introduces a structural trade-off: the randomly sampled sign b becomes a new secret variable of the protocol. Consequently, the prover must add b to the commitment scheme and provide additional cryptographic checks to mathematically prove its validity. Specifically, it must be proven in zero-knowledge that b is a valid sign scalar (a polynomial consisting solely of a constant term equal to either 1 or -1), adding computational overhead and complexity to the overall protocol.

The Rej_2 Rejection Algorithm

To maintain the optimal standard deviation bounds of the bimodal approach without paying the zero-knowledge overhead required to hide the sign bit b, Lyubashevsky et al. [LNS21b] proposed an alternative rejection sampling algorithm, called Rej_2.

The Rej_2 algorithm fundamentally simplifies the process while matching the M bound of Rej_0. It introduces a strict geometric condition: the algorithm requires the inner product $\langle \mathbf{z}, \mathbf{v} \rangle$ to be non-negative. If $\langle \mathbf{z}, \mathbf{v} \rangle < 0$, the masked vector $\mathbf{z}$ is immediately rejected.

Since $\langle \mathbf{z}, \mathbf{v} \rangle \geq 0$, the term $-2\langle \mathbf{z}, \mathbf{v} \rangle$ in the standard ratio is always negative or zero. Because of this, the maximum bound M naturally collapses to:

$$M = \exp\left(\frac{\|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right). \tag{3.21}$$

With this bound, a standard deviation of $\mathfrak{s} \approx 0.675\|\mathbf{v}\|$ is sufficient to maintain $M \approx 3$. Because the algorithm automatically discards half of the samples (when the inner product is negative), the expected number of iterations to accept a value becomes $2M$. Even with this halving of the acceptance rate, the required $\mathfrak{s}$ is reduced by nearly a factor of 10 compared to Rej_1, drastically reducing the proof size.

However, this algorithmic efficiency comes with a specific security trade-off: the protocol leaks exactly one bit of information about the shift vector $\mathbf{v}$ (the sign of the inner product). For this reason, Rej_2 is inherently insecure if utilized to mask the same static secret element multiple times. Conversely, when used within LNP to mask a one-time-use random noise vector that changes in every execution, the leakage of this single bit provides no usable information about the underlying static secret $\mathbf{s}$, making Rej_2 a perfectly optimized choice.

Applying the Rejection Algorithms

In protocols with multiple secret components (such as $\mathbf{s}_1$ and $\mathbf{s}_2$), a rejection sampling check must be evaluated independently for each masked response. The protocol proceeds only if all checks pass simultaneously; if even a single vector is rejected, the execution aborts and the prover must restart.

Following the LNP framework [LNP22], the choice of the rejection algorithm depends directly on the nature of the masked secret.

The standard Rej_1 algorithm can be used without modifying the core protocol. It is completely secure and allows us to safely reuse the same commitment randomness $\mathbf{s}_2$ across multiple executions.

However, if the algorithm is deployed for a one-time commitment scheme (such as an identification protocol), it is highly effective to combine different rejection strategies. Specifically, we can apply Rej_1 to the long-term secret $\mathbf{s}_1$ (which remains static and specific to a given user across all interactions) and apply the efficient Rej_2 to components associated with the one-time-use randomness $\mathbf{s}_2$. This protocol remains secure because the masked

temporary values change at each execution, safely absorbing the one-bit leakage.

An alternative approach for masking the static secret $\mathbf{s}_1$ involves implementing the bimodal Rej_0 algorithm. While this provides optimal efficiency without leaking any information, it introduces a randomly sampled sign bit b. As explained in Section 3.3.2, using Rej_0 requires adding extra verification equations to check the validity of this sign.

3.3.3 Verifying the Norm of the Masked Secret

Once the secret vector $\mathbf{s}$ has been masked, the rejection sampling algorithm ensures that the resulting response $\mathbf{z}$ is statistically indistinguishable from a sample drawn from a zero-centered discrete Gaussian distribution. We can evaluate the expected geometric bounds of this vector by applying the standard tail bound properties of the Gaussian distribution, which state that:

$$\Pr\left[\|\mathbf{z}\| > t \cdot \mathfrak{s}\sqrt{md}\right] < \left(te^{\frac{1-t^2}{2}}\right)^{md}. \tag{3.22}$$

By choosing the parameter $t = \sqrt{2}$ in Equation 3.22, the right-hand side of the inequality drops exponentially. Consequently, we can state that, with overwhelming probability, the norm of the masked response is strictly bounded by:

$$\|\mathbf{z}\| \leq \mathfrak{s}\sqrt{2md}. \tag{3.23}$$

The verifier establishes this theoretical bound as the strict acceptance criterion to confirm that the transmitted vector $\mathbf{z}$ possesses a sufficiently small norm.

To mathematically demonstrate the soundness of this check, we rely on the Knowledge Extractor framework.

Suppose a malicious prover attempts to cheat by using a secret $\mathbf{s}$ with a massive norm, artificially crafting a noise vector $\mathbf{y}$ to force $\mathbf{z}$ inside the valid bounds. If the prover is systematically successful, a Knowledge Extractor could rewind the protocol execution and send two distinct challenges, c_0 and c_1, for the same pre-committed noise vector $\mathbf{y}$. This would result in two valid transcripts:

$$\begin{aligned} \mathbf{z}_0 &= c_0\mathbf{s} + \mathbf{y} \quad \text{with} \quad \|\mathbf{z}_0\| \leq \mathfrak{s}\sqrt{2md}, \\ \mathbf{z}_1 &= c_1\mathbf{s} + \mathbf{y} \quad \text{with} \quad \|\mathbf{z}_1\| \leq \mathfrak{s}\sqrt{2md}. \end{aligned} \tag{3.24}$$

By algebraically subtracting the two responses, the fixed noise vector $\mathbf{y}$ is perfectly eliminated. Applying the triangle inequality to the difference yields a strict geometric boundary:

$$\|\mathbf{z}_0 - \mathbf{z}_1\| = \|(c_0 - c_1)\mathbf{s}\| \leq \|\mathbf{z}_0\| + \|\mathbf{z}_1\| \leq 2\mathfrak{s}\sqrt{2md}. \tag{3.25}$$

Crucially, because the challenge space is specifically designed to ensure that the difference $(c_0 - c_1)$ is a known, small, and *invertible* polynomial (a structural property fundamental to achieving exact extraction, as formalized in Section 3.8.1), Equation 3.25 mathematically

traps the secret $\mathbf{s}$. If the prover had used a secret exceeding this implicit bound, it would have been geometrically impossible for both $\mathbf{z}_0$ and $\mathbf{z}_1$ to fall within the verifier's acceptance radius. Therefore, checking the norm of $\mathbf{z}$ inherently guarantees that the secret $\mathbf{s}$ satisfies the small-norm requirement, fulfilling the soundness property of the protocol.

3.4 Evaluating the ℓ_2 Norm via a Quadratic Equation

Once the secret vector $\mathbf{s}_1$ has been committed, the LNP protocol [LNP22] must rigorously prove that its ℓ_2 norm satisfies the required security bounds. In this section, we explain how the protocol resolves this problem by expressing the norm calculation as a quadratic polynomial equation.

Operating over the quotient ring $\mathcal{R}_p = \mathbb{Z}_p[X]/(X^d+1)$, a secret coefficient vector $\mathbf{s} \in \mathbb{Z}_p^d$ is uniquely represented as a polynomial $s(X) = \sum_{i=0}^{d-1} s_i X^i$. The squared ℓ_2 norm of this vector natively corresponds to its standard inner product:

$$\|\mathbf{s}\|_2^2 = \langle \mathbf{s}, \mathbf{s} \rangle = \sum_{i=0}^{d-1} s_i^2. \tag{3.26}$$

To compute this inner product using purely polynomial arithmetic, the protocol exploits the cyclotomic ring structure by introducing the automorphism σ_{-1}. This ring homomorphism operates by mapping $X \mapsto X^{-1}$, transforming a polynomial $s(X)$ into its conjugate $\sigma_{-1}(s(X)) = s(X^{-1})$.

Since the space is reduced modulo X^d+1, the relation $X^d \equiv -1$ intrinsically holds, meaning $X^{-i} \equiv -X^{d-i}$. Substituting these negative exponents allows the conjugate polynomial to be explicitly evaluated as:

$$s(X^{-1}) = s_0 - \sum_{i=1}^{d-1} s_i X^{d-i}. \tag{3.27}$$

This automorphism provides a direct mathematical bridge between polynomial arithmetic and linear algebra. In lattice-based schemes, polynomial multiplication is geometrically represented by negacyclic transformation matrices. Crucially, if a matrix represents the multiplication by $s(X)$, the *transpose* of that matrix is algebraically identical to multiplying by its conjugate $s(X^{-1})$.

The squared ℓ_2 norm can now be extracted from the polynomial multiplication of $s(X)$ and its conjugate $s(X^{-1})$. Specifically, the inner product $\langle \mathbf{s}, \mathbf{s} \rangle$ corresponds exactly to the constant term (the coefficient of X^0) of this product. Let $u(X) = s(X) \cdot s(X^{-1}) \pmod{X^d+1}$. The constant term, denoted as $\tilde{u}$, is computed by multiplying s_0 by itself,

and each $s_i X^i$ by its corresponding term $-s_i X^{d-i}$ in the conjugate:

$$\begin{aligned}\tilde{u} &= s_0 \cdot s_0 + \sum_{i=1}^{d-1} \left(s_i X^i\right) \cdot \left(-s_i X^{d-i}\right) \\ &= s_0^2 - \sum_{i=1}^{d-1} s_i^2 X^d \\ &= s_0^2 - \sum_{i=1}^{d-1} s_i^2 (-1) \\ &= \sum_{i=0}^{d-1} s_i^2 = \|\mathbf{s}\|_2^2.\end{aligned} \tag{3.28}$$

By relying on this algebraic identity, the prover can demonstrate that the secret vector possesses a small ℓ_2 norm simply by proving a property about the constant term of a polynomial product, entirely avoiding the inefficiencies of the Chinese Remainder Theorem embeddings.

3.4.1 Masking the Polynomial and Isolating the Constant Term

To ensure the protocol achieves the Zero-Knowledge property, the polynomial $u(X)$ cannot be transmitted directly to the verifier, as doing so would reveal information about the secret $\mathbf{s}$. To prevent this information leakage, the prover masks $u(X)$ by sampling a random polynomial $g(X) \in \mathcal{R}_q$ whose constant term is strictly zero (i.e., $\tilde{g} = 0$).

This random polynomial $g(X)$ is added to the computation to perfectly blind all coefficients of $u(X)$ except for the constant term. The polynomial $g(X)$ itself is never revealed; otherwise, an attacker could simply subtract it from the masked result, exposing the secret. However, the verifier must be mathematically convinced that the constant term of $g(X)$ is indeed zero without ever seeing the polynomial.

To achieve this, the protocol employs a cryptographic challenge-and-response mechanism:

1. The prover commits to the masking polynomial $g(X)$ by embedding it within the BDLOP part of the ABDLOP commitment (specifically, as part of the message vector $\mathbf{m}$), and sends the commitment to the verifier.
2. The verifier responds with a uniformly random challenge $\gamma \leftarrow \mathbb{Z}_q \setminus \{0\}$.
3. The prover computes and reveals the masked polynomial $h(X) = g(X) + \gamma u(X) \pmod{q}$.

Security Analysis

The security and robustness of this interactive mechanism are analyzed based on two fundamental cryptographic properties:

- **Zero-Knowledge Property:** From the verifier's perspective, the revealed polynomial $h(X)$ leaks no information about $u(X)$. Because $g(X)$ is sampled uniformly at random and its commitment securely hides its structure, the resulting $h(X)$ appears as a perfectly uniform random polynomial over $\mathcal{R}_q$. This statistically blinds the secret vector coefficients.

- **Soundness:** A malicious prover might attempt to cheat by committing to an invalid secret (where the constant term $\tilde{u}$ does not equal the correct ℓ_2 norm) and compensating for the error by choosing a polynomial $g(X)$ with a non-zero constant term ($\tilde{g} \neq 0$). The soundness error of this attack depends heavily on the algebraic nature of the modulus q:

  - *If q is prime:* Since the challenge γ is sampled and sent by the verifier after the prover has committed to $g(X)$, the prover cannot alter $g(X)$ retroactively. The constant term of the evaluated polynomial is strictly bound by $\tilde{h} \equiv \tilde{g} + \gamma\tilde{u} \pmod{q}$. The probability of the malicious prover blindly guessing the exact challenge γ required to satisfy this linear equation is exactly $\frac{1}{q-1}$.

  - *If q is composite:* If q is not a prime number, an attacker can strategically exploit its non-trivial divisors. Consider the adversarial target equation $\gamma\tilde{u} \equiv -\tilde{g} \pmod{q}$. According to the Linear Congruence Theorem, an equation of the form $ax \equiv b \pmod{n}$ has exactly $\gcd(a, n)$ distinct solutions in the unknown x. Since the attacker has full control over the false value $\tilde{u}$, they can actively maximize this metric by choosing $\tilde{u} = \frac{q}{q_1}$, where q_1 is the smallest prime factor of q. In this way, the maximum number of valid solutions for γ becomes $\gcd\left(q, \frac{q}{q_1}\right) = \frac{q}{q_1}$. Since the verifier samples γ uniformly from the non-zero space $\mathbb{Z}_q \setminus \{0\}$, the probability that the random challenge falls into this expanded set of adversarial solutions is bounded by:

    $$\mathbb{P}[\text{Attacker Success}] \leq \frac{\frac{q}{q_1}}{q-1} \approx \frac{1}{q_1}. \tag{3.29}$$

    Therefore, the soundness error of a single protocol execution deteriorates proportionally to the smallest prime factor of the modulus.

To guarantee standard cryptographic security (e.g., ensuring a soundness error bounded by 2^{-128}), a single execution might not be sufficient if q_1 is small. In such cases, the protocol must be structurally reinforced through k parallel repetitions. The probability of a malicious prover successfully passing all k independent challenges drops to $\left(\frac{1}{q_1}\right)^k$. The parameter k is therefore strategically chosen such that $\left(\frac{1}{q_1}\right)^k \leq 2^{-128}$. To implement this, the prover samples k distinct masking polynomials $g_j(X)$, and the verifier replies with k independent challenges γ_j.

3.4.2 Zero-Knowledge Proof of the Polynomial Relation

While the previous mechanism successfully masks the secret polynomial $u(X)$, simply transmitting the evaluations $h_j(X)$ is insufficient. The verifier must be unequivocally convinced that each $h_j(X)$ was computed honestly, rather than being arbitrarily fabricated by the prover.

To prove this correct construction, the prover defines the residual polynomial relation $w_j(X) = g_j(X) + \gamma_j u(X) - h_j(X)$ and must demonstrate that $w_j(X) = 0 \pmod q$.

Because $u(X)$ is internally derived from the product of the secret vector $\mathbf{s}$ and its conjugate, demonstrating $w_j(X) = 0$ inherently reduces to evaluating a quadratic relation. To manage this efficiently without leaking information, the LNP framework [LNP22] employs a specialized Zero-Knowledge Proof sub-protocol. This mechanism provides a mathematical guarantee that the secret $\mathbf{s}$, which remains perfectly hidden within the ABDLOP commitment, is the exact and valid solution satisfying this public quadratic equation.

3.5 ZKP for Solutions of Quadratic Equations

As established in previous sections (in particular Section 3.3.1), zero-knowledge proofs for linear relations are straightforward to analyze. In a linear setting, Knowledge Soundness is easily guaranteed because an Extractor can take two valid transcripts and factor out the challenge difference $(c_0 - c_1)$ to mathematically isolate the underlying secret.

However, verifying advanced properties like the exact ℓ_2 norm requires evaluating a quadratic equation (Section 3.4). This introduces a fundamental algebraic obstacle: substituting a masked vector $\mathbf{z} = c\mathbf{s} + \mathbf{y}$ into a quadratic relation generates cross-terms involving the conjugated challenge $\sigma_{-1}(c)$. Because c and its conjugate are algebraically entangled, standard extraction techniques fail.

To illustrate this problem, consider a simplified quadratic relation $\sigma_{-1}(\mathbf{s})\mathbf{s} = \beta^2$. When the prover substitutes the masked response into the equation, the expansion yields:

$$\begin{aligned}\sigma_{-1}(\mathbf{z})\mathbf{z} &= \sigma_{-1}(c\mathbf{s}+\mathbf{y})(c\mathbf{s}+\mathbf{y}) \\ &= (\sigma_{-1}(c)\sigma_{-1}(\mathbf{s}) + \sigma_{-1}(\mathbf{y}))(c\mathbf{s}+\mathbf{y}) \\ &= \beta^2\sigma_{-1}(c)c + \sigma_{-1}(c)\sigma_{-1}(\mathbf{s})\mathbf{y} + c\sigma_{-1}(\mathbf{y})\mathbf{s} + \sigma_{-1}(\mathbf{y})\mathbf{y}.\end{aligned} \tag{3.30}$$

Ideally, to guarantee Knowledge Soundness, an Extractor would group the remaining elements into a linear combination and isolate the independent coefficients of $\sigma_{-1}(c)$ and c by solving a multivariate system. However, because the polynomials $\sigma_{-1}(c)$ and c are mathematically dependent, this standard matrix extraction method fails entirely.

To resolve this issue, the protocol imposes a strict condition on the challenge space $\mathcal{C}$ (restricting it to a subspace $\mathcal{C}^*$): all sampled challenges must be invariant under the automorphism, meaning $\sigma_{-1}(c) = c$. By enforcing this property, the relationship between

the terms becomes clear, allowing the expansion to be regrouped by powers of c:

$$\begin{aligned}\sigma_{-1}(\mathbf{z})\mathbf{z} &= \beta^2 c^2 + c\sigma_{-1}(\mathbf{s})\mathbf{y} + c\sigma_{-1}(\mathbf{y})\mathbf{s} + \sigma_{-1}(\mathbf{y})\mathbf{y} \\ &= \beta^2 c^2 + c(\sigma_{-1}(\mathbf{s})\mathbf{y} + \sigma_{-1}(\mathbf{y})\mathbf{s}) + \sigma_{-1}(\mathbf{y})\mathbf{y}.\end{aligned} \tag{3.31}$$

While this restriction guarantees algebraic correctness, it introduces a severe structural constraint. Given the cyclotomic property $X^d \equiv -1$, the conjugate of a polynomial $c(X)$ enforces that $c_i = -c_{d-i}$. If the condition $\sigma_{-1}(c) = c$ is required, the coefficients must satisfy this symmetry. Consequently, the verifier can only freely choose the first half of the coefficients $(c_0, \ldots, c_{\frac{d}{2}-1})$. The middle coefficient $c_{\frac{d}{2}}$ is mathematically forced to be 0, while the remaining upper-half coefficients are entirely determined by the lower half.

Because this symmetry effectively halves the entropy of the challenge space, it drastically reduces the difficulty for an attacker to guess the challenge. Therefore, to maintain an exponentially large challenge space and achieve standard cryptographic soundness, the degree d of the underlying polynomial ring must be proportionally increased.

Building upon this concept and leveraging the equivalence between polynomial conjugation and matrix transposition (as established in Section 3.4), this process can be expanded to evaluate a generic quadratic equation of the form $\mathbf{s}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{s} + r_0 = 0$.

By operating within the restricted challenge space $\mathcal{C}^*$ and substituting the masked response $\mathbf{z}$, the closed-form expansion becomes:

$$\begin{aligned}&\mathbf{z}^T\mathbf{R}_2\mathbf{z} + c\mathbf{r}_1^T\mathbf{z} + c^2 r_0 \\ &= (c\mathbf{s} + \mathbf{y})^T\mathbf{R}_2(c\mathbf{s} + \mathbf{y}) + c\mathbf{r}_1^T(c\mathbf{s} + \mathbf{y}) + c^2 r_0 \\ &= c^2\mathbf{s}^T\mathbf{R}_2\mathbf{s} + c\mathbf{s}^T\mathbf{R}_2\mathbf{y} + c\mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{y}^T\mathbf{R}_2\mathbf{y} + c^2\mathbf{r}_1^T\mathbf{s} + c\mathbf{r}_1^T\mathbf{y} + c^2 r_0 \\ &= c^2(\mathbf{s}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{s} + r_0) + c(\mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}) + \mathbf{y}^T\mathbf{R}_2\mathbf{y}.\end{aligned} \tag{3.32}$$

Having expressed the expansion strictly in terms of the scalar variable c, the mathematical mechanism becomes evident. If the prover can successfully demonstrate to the verifier that the evaluated response satisfies:

$$\mathbf{z}^T\mathbf{R}_2\mathbf{z} + c\mathbf{r}_1^T\mathbf{z} + c^2 r_0 = c(\mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}) + \mathbf{y}^T\mathbf{R}_2\mathbf{y}, \tag{3.33}$$

then the leading quadratic term, $c^2(\mathbf{s}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{s} + r_0)$, must inherently evaluate to zero. This algebraic equality formally proves that the underlying quadratic equation is indeed satisfied by the secret $\mathbf{s}$.

3.5.1 Integrating the Quadratic Relation with the Commitment Scheme

Consider the most general case, which involves a quadratic equation dependent on $\mathbf{s}_1, \sigma_{-1}(\mathbf{s}_1), \mathbf{m}$, and $\sigma_{-1}(\mathbf{m})$. To represent this within a single mathematical structure, a concatenated vector $\mathbf{s} = [\mathbf{s}_1 \parallel \sigma_{-1}(\mathbf{s}_1) \parallel \mathbf{m} \parallel \sigma_{-1}(\mathbf{m})]^T$ is explicitly defined.

The initial masking vectors $\mathbf{y}_1$ and $\mathbf{y}_2$ must then be assembled into a coherent composite masking vector for $\mathbf{s}$. The masking vector $\mathbf{y}_m$ associated with the message $\mathbf{m}$ is implicitly derived from the BDLOP commitment relation $\mathbf{m} = \mathbf{t}_B - \mathbf{B}\mathbf{s}_2 \pmod q$. Specifically, by substituting the masked values into this relation, we obtain $\mathbf{B}(c\mathbf{s}_2 + \mathbf{y}_2) = c\mathbf{t}_B - c\mathbf{m} + \mathbf{B}\mathbf{y}_2$. From this equality, the masked response $\mathbf{z}_m$ can be recovered locally by the verifier as $\mathbf{z}_m = c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2$, which algebraically evaluates to $c\mathbf{m} - \mathbf{B}\mathbf{y}_2$. Consequently, the implicit masking vector for the message is defined natively as $\mathbf{y}_m = -\mathbf{B}\mathbf{y}_2$. Because of this algebraic link, the prover only needs to transmit the vector $\mathbf{z}_2$ to allow the verifier to compute $\mathbf{z}_m$.

Mirroring the structure of $\mathbf{s}$, the overall masking vector $\mathbf{y}$ is assembled by concatenating these individual noise components and their automorphisms. Using the compact notation $\sigma^i(x) = [x \parallel \sigma_{-1}(x)]^T$, the masking vector takes the form $\mathbf{y} = [\sigma^i(\mathbf{y}_1) \parallel -\sigma^i(\mathbf{B}\mathbf{y}_2)]^T$. This precise alignment allows the final composite masked vector to be computed uniformly as $\mathbf{z} = c\mathbf{s} + \mathbf{y}$. Due to the homomorphic properties of the conjugation over the challenge space $\mathcal{C}^*$, this expands to the explicit block form $\mathbf{z} = [\sigma^i(\mathbf{z}_1) \parallel \sigma^i(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2)]^T$, allowing the verifier to reconstruct the entire structure using only $\mathbf{z}_1$ and $\mathbf{z}_2$.

As detailed in Section 3.3.1, to preserve the Zero-Knowledge property, the protocol must guarantee that the prover commits to the terms dependent on the masking vector $\mathbf{y}$ before the challenge c is sampled. For a generic quadratic relation, these terms are the linear polynomial $g_1 = \mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}$ and the constant polynomial $g_0 = \mathbf{y}^T\mathbf{R}_2\mathbf{y}$.

However, transmitting g_1 and g_0 in the clear would leak critical statistical information about the masking vector $\mathbf{y}$. This represents a fatal vulnerability for a ZKP, as leaking $\mathbf{y}$ inherently compromises the secrecy of $\mathbf{s}$ due to their direct algebraic correlation in the public response $\mathbf{z}$.

Within the LNP framework [LNP22], this problem is resolved by repurposing the existing commitment elements to dynamically mask these coefficients. Specifically, the protocol reuses the commitment randomness $\mathbf{s}_2$ and its associated masking vector $\mathbf{y}_2$ to securely blind g_1 and g_0. To achieve this without altering the core structure, a randomly sampled public vector $\mathbf{b}$ (part of the public parameters) is incorporated into the protocol.

The prover constructs two auxiliary masking terms: $t = g_1 + \mathbf{b}^T\mathbf{s}_2$ and $v = g_0 + \mathbf{b}^T\mathbf{y}_2$. This construction allows the verifier to mathematically neutralize the added masking elements during the verification phase. By evaluating $c\mathbf{b}^T\mathbf{s}_2 + \mathbf{b}^T\mathbf{y}_2 = \mathbf{b}^T(c\mathbf{s}_2 + \mathbf{y}_2) = \mathbf{b}^T\mathbf{z}_2$, the verifier reconstructs the mask using only the legitimate response $\mathbf{z}_2$.

Finally, the verifier actively validates the quadratic relation by checking the following equality:

$$\mathbf{z}^T\mathbf{R}_2\mathbf{z} + c\mathbf{r}_1^T\mathbf{z} + c^2 r_0 = ct + v - \mathbf{b}^T\mathbf{z}_2. \tag{3.34}$$

The complete flow of the commit-and-prove sequence is summarized in Fig. 3.1.

3.5.2 Verifying Multiple Quadratic Relations

The protocol allows verifying k quadratic statements simultaneously. To simplify the notation, let the i-th generic quadratic relation evaluated on the concatenated secret vector be expressed as $f_i(\mathbf{s}_1, \sigma_{-1}(\mathbf{s}_1), \mathbf{m}, \sigma_{-1}(\mathbf{m})) = 0$.

To prove that these k relations evaluate to zero efficiently, the individual functions f_i are combined into a single aggregated function f. This ensures that, except with negligible probability, $f = 0$ holds if and only if $f_i = 0$ for all $i \in \{1, \ldots, k\}$.

A preliminary approach to define f might be to simply sum all functions f_i. However, a malicious prover could craft non-zero evaluations that cancel each other out (e.g., $f_1 = -f_2 \neq 0$), resulting in a sum of zero despite the underlying relations being unsatisfied.

To prevent this, a random linear combination is employed. The verifier samples a set of uniformly random challenges $\mu_i \leftarrow \mathcal{R}_q$ and sends them to the prover, who constructs the aggregated function:

$$f = \sum_{i=1}^{k} \mu_i f_i. \tag{3.35}$$

Under this construction, obtaining $f = 0$ when some $f_i \neq 0$ requires the random coefficients μ_i to perfectly cancel the non-zero terms.

The statistical probability of such a forgery depends on the algebraic structure of the underlying polynomial quotient ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$. Because $\mathcal{R}_q$ is not a field, it contains zero divisors. Specifically, by choosing a modulus parameter $q \equiv 5 \pmod 8$, the polynomial X^d+1 is guaranteed to factor modulo q into exactly two irreducible polynomials of degree $d/2$. Consequently, according to the Chinese Remainder Theorem, the ring $\mathcal{R}_q$ is isomorphic to the direct product of two fields: $\mathbb{F}_{q^{d/2}} \times \mathbb{F}_{q^{d/2}}$. A malicious prover could evaluate f_i such that it yields zero modulo the first irreducible factor but remains non-zero modulo the second.

For the aggregated function f to evaluate to zero over the entire ring $\mathcal{R}_q$, the linear combination $\sum_{i=1}^{k} \mu_i f_i$ must simultaneously evaluate to zero in both underlying fields. If f_i is non-zero in at least one field of size $q^{d/2}$, fixing the remaining $k-1$ terms leaves exactly one valid choice for the challenge μ_i to force the overall sum to zero. Since every non-zero element in a field has a unique multiplicative inverse, this specific μ_i is uniquely determined. Therefore, the probability of the verifier sampling exactly this challenge uniformly at random is $q^{-d/2}$.

If the modulus q is composite, the ring structure decomposes further, and the worst-case probability is dictated by its smallest prime factor q_1 (following the logic outlined in Section 3.4.1). Consequently, the probability of a malicious prover successfully producing $f = 0$ without satisfying all underlying relations is rigorously bounded by $q_1^{-d/2}$.

This aggregation technique is essential for the practical efficiency of the protocol. By

Private information: $(\mathbf{s}_1, \mathbf{m}) \in \mathcal{R}_q^{m_1+\ell}$ so that $\|\mathbf{s}_1\| \leqslant \alpha$, $\mathbf{s}_2 \leftarrow \chi^{m_2}$
Public information: $\mathbf{A}_1 \in \mathcal{R}_q^{n \times m_1}, \mathbf{A}_2 \in \mathcal{R}_q^{n \times m_2}, \mathbf{B} \in \mathcal{R}_q^{\ell \times m_2}, \mathbf{b} \in \mathcal{R}_q^{m_2}$
$\begin{bmatrix} \mathbf{t}_A \\ \mathbf{t}_B \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{0} \end{bmatrix} \cdot \mathbf{s}_1 + \begin{bmatrix} \mathbf{A}_2 \\ \mathbf{B} \end{bmatrix} \cdot \mathbf{s}_2 + \begin{bmatrix} \mathbf{0} \\ \mathbf{m} \end{bmatrix},$
$r_0 \in \mathcal{R}_q, \mathbf{r}_1 \in \mathcal{R}_q^{k(m_1+\ell)}, \mathbf{R}_2 \in \mathcal{R}_q^{k(m_1+\ell) \times k(m_1+\ell)}, \sigma^i(x) = (x, \sigma_{-1}(x))$
$\mathbf{s}^T \mathbf{R}_2 \mathbf{s} + \mathbf{r}_1^T \mathbf{s} + r_0 = 0$

Prover		**Verifier**
$\mathbf{s} := \begin{bmatrix} (\sigma^i(\mathbf{s}_1)) \\ (\sigma^i(\mathbf{m})) \end{bmatrix}$		
$\mathbf{y}_1 \leftarrow D_{\mathfrak{s}_1}^{m_1}$ $\mathbf{y}_2 \leftarrow D_{\mathfrak{s}_2}^{m_2}$ $\mathbf{w} := \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$ $\mathbf{y} := \begin{bmatrix} (\sigma^i(\mathbf{y}_1)) \\ -(\sigma^i(\mathbf{B}\mathbf{y}_2)) \end{bmatrix}$		
$g_1 := \mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}$ $t := \mathbf{b}^T\mathbf{s}_2 + g_1$ $v := \mathbf{y}^T\mathbf{R}_2\mathbf{y} + \mathbf{b}^T\mathbf{y}_2$		
	$\xrightarrow{\mathbf{w}, t, v}$	
		$c \leftarrow \mathcal{C}^*$
	$\xleftarrow{c}$	
$\mathbf{z}_1 := c\mathbf{s}_1 + \mathbf{y}_1$ $\mathbf{z}_2 := c\mathbf{s}_2 + \mathbf{y}_2$ for $i = 1, 2$: if $\mathsf{Rej}_i(\mathbf{z}_i, c\mathbf{s}_i, \mathfrak{s}_i) = 1$ then $\mathbf{z}_1, \mathbf{z}_2 := \perp$		
	$\xrightarrow{\mathbf{z}_1, \mathbf{z}_2}$	
		$\mathbf{z} := \begin{bmatrix} (\sigma^i(\mathbf{z}_1)) \\ (\sigma^i(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2)) \end{bmatrix}$
		$f := ct - \mathbf{b}^T\mathbf{z}_2$ Accept iff $\|\mathbf{z}_1\| \leqslant \mathfrak{s}_1\sqrt{2m_1d}$ and $\|\mathbf{z}_2\| \leqslant \mathfrak{s}_2\sqrt{2m_2d}$ and $\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A$ and $\mathbf{z}^T\mathbf{R}_2\mathbf{z} + c\mathbf{r}_1^T\mathbf{z} + c^2r_0 - f = v$

Figure 3.1. Commit-and-prove interactive protocol for quadratic relations under the LNP framework.

compressing k distinct quadratic relations into a single evaluated polynomial f, the prover avoids generating and transmitting k independent proofs. While the communication cost on the verifier's side increases linearly, since a distinct challenge μ_i must be sampled and sent for each statement, the size of the actual proof returned by the prover remains strictly constant. This structural optimization ensures that the final Zero-Knowledge Proof is highly compact, regardless of the number of simultaneous quadratic constraints required by the application.

3.6 Checking the ℓ_2 Norm Bound

With the foundational tools established, the protocol can now be extended to prove bounds on the ℓ_2 norm of the secret. However, two primary challenges remain: zero-knowledge protocols evaluate exact algebraic equations rather than inequalities, and these evaluations occur modulo q, which does not inherently prevent modular wrap-around.

3.6.1 Transforming the Inequality into an Exact Equation

A direct evaluation of the exact norm would rely on the quadratic relation $\mathbf{s}^T\mathbf{s} - \beta^2 = 0$. The constant term of this evaluation equals zero if and only if the squared ℓ_2 norm is exactly β^2. To generalize this to an inequality bound $\|\mathbf{s}\|_2^2 \leqslant \beta^2$, the LNP framework [LNP22] introduces a non-negative slack variable representing the difference between the maximum bound and the actual squared norm: $\Delta = \beta^2 - \|\mathbf{s}\|_2^2$.

Since Δ is bounded by β^2, it can be represented in binary format using $\kappa = \lfloor \log_2(\beta^2) \rfloor + 1$ bits. This binary representation is encoded as the coefficients of a new polynomial $\delta(X)$ of degree d (assuming $d \geq \kappa$). To integrate this into the existing proof system, the secret vector is extended through concatenation, yielding a new composite secret $\mathbf{s}^* = [\mathbf{s} \parallel \delta]^T$. Consequently, all public matrices interacting with the secret must be proportionally expanded by appending zero columns, ensuring that the original matrix-vector products remain algebraically unchanged.

To algebraically reconstruct the integer value Δ from the binary polynomial $\delta(X)$, a public weighting polynomial is defined as $\eta(X) = \sum_{i=0}^{\kappa-1} 2^i X^{d-1-i}$. Let η represent the coefficient vector of this polynomial. The norm bounding inequality is thus successfully translated into a strict equality using the standard inner product:

$$\mathbf{s}^T\mathbf{s} + \eta^T\delta - \beta^2 = 0. \tag{3.36}$$

Despite this translation, a critical security vulnerability remains. The prover has encoded Δ into a polynomial δ, but has not yet proven to the verifier that the coefficients of δ are strictly binary. Without this guarantee, a malicious prover could inject arbitrarily large integer coefficients into δ, satisfying the equation by intentionally triggering a wrap-around modulo q, even if $\|\mathbf{s}\|_2^2$ strictly exceeds β^2.

To prevent this type of attack, the protocol must enforce the binary constraint on δ. A coefficient x is binary ($x \in \{0, 1\}$) if and only if $x(x-1) = 0$. Therefore, an additional quadratic relation is appended to the protocol to prove that the inner product of the vector with its shifted self evaluates to zero:

$$\langle \delta, \delta - \mathbf{1} \rangle = 0, \tag{3.37}$$

where $\mathbf{1}$ denotes the all-ones vector. Because the aggregated secret $\mathbf{s}^*$ is strictly bounded using the approximate range proofs detailed in Section 3.6.2, its coefficients are guaranteed to be small. This ensures that the inner product evaluates to zero natively over the integers ($\mathbb{Z}$), completely removing the possibility of satisfying the equation via modular wrap-around.

As detailed in Section 3.5.2, this auxiliary equation can be aggregated into the main proof by combining multiple quadratic relations into a single evaluated function f.

3.6.2 Preventing Modular Wrap-Around

A further complication arises because the polynomial equations are evaluated modulo q. Consequently, a vector may satisfy an algebraic modular relation while its actual ℓ_2 norm over the integers wraps around the modulus. In such cases, the norm evaluated natively over $\mathbb{Z}$ diverges entirely from its representation in $\mathbb{Z}_q$. While the equation strictly holds within the quotient ring, the physical norm of the vector remains massive. This compromises the security of the scheme, as the hardness of the underlying M-SIS problem relies strictly on the extracted solution possessing a genuinely small norm over the integers.

To mitigate this vulnerability, the protocol employs an *approximate range proof*. Rather than proving an exact tight bound, which is computationally expensive in lattice cryptography, this technique enforces a relaxed bound at a significantly reduced computational cost. The theoretical foundation for this approach within the LNP framework [LNP22] is derived from the techniques introduced by [GHL21].

Specifically, the approximate range proof relies on Lemma 2.9 of the LNP framework [LNP22](where the generalized modulus P corresponds to our ring modulus q), which states the following:

Lemma 3.6.1. Fix $m, P \in \mathbb{N}$ and a bound $b \leqslant P/41m$. Let $\mathbf{w} \in [\pm P/2]^m$ with $\|\mathbf{w}\| \geqslant b$, and let $\mathbf{y}$ be an arbitrary vector in $[\pm P/2]^m$. Then

$$\Pr_{\mathbf{R} \leftarrow \mathsf{Bin}_1^{256 \times m}} \left[\|\mathbf{R}\mathbf{w} + \mathbf{y} \pmod P\| < \frac{1}{2} b \sqrt{26} \right] < 2^{-128}. \tag{3.38}$$

Similar to the commitment phase, to demonstrate that the norm is small, the verification is performed on a masked vector $\mathbf{z} = \mathbf{R}\mathbf{s} + \mathbf{y}$. However, unlike previous constructions where the masking element is a challenge polynomial $c \in \mathcal{R}_q$, here a projection matrix $\mathbf{R} \leftarrow \mathsf{Bin}_1^{(256/d) \times m}$ is utilized. This projection allows the vector $\mathbf{s}$ to be compressed into a lower-dimensional space, significantly reducing the size of the resulting masked vector $\mathbf{z}$.

Because this projection strictly reduces the output to exactly 256 coefficients, the matrix $\mathbf{R}$ can be compactly represented using exactly $256/d$ polynomials. This differs fundamentally from the commitment phase, where the masked term $\mathbf{z}$ must preserve specific structural and geometric properties to satisfy the verification equation, thereby precluding the use of a projecting matrix.

To formally verify that the term $\mathbf{z}$ is correctly constructed, the prover defines the relation $F_i = z_i - \langle \mathbf{r}_i, \mathbf{s} \rangle - y_i = 0$, where $\mathbf{r}_i$ represents a row of the matrix $\mathbf{R}$. This relation is subsequently appended to the protocol as an additional linear equation to be checked by the verifier.

Once a valid vector $\mathbf{z}$ is computed, the prover applies a rejection sampling algorithm to prevent any leakage of the secret vector $\mathbf{s}$ before transmitting the response. Upon receiving $\mathbf{z}$, the verifier checks whether the norm of the vector falls within the expected bounds using Equation 3.22. The soundness of this mechanism relies directly on Lemma 3.6.1: by appropriately choosing the bound b, the probability that a malicious prover's invalid vector yields a masked term $\mathbf{Rw} + \mathbf{y}$ with a small norm is negligibly small. This ensures that an attacker cannot bypass the rejection sampling step undetected.

In the specific implementation of the LNP framework [LNP22], the authors optimize the protocol by employing the bimodal Rej_0 algorithm. To accommodate this, the structure of the masked term is modified to include a random sign, resulting in $\mathbf{z} = b\mathbf{Rs} + \mathbf{y}$, where $b \leftarrow \{-1, 1\}$.

While employing Rej_0 significantly increases the acceptance rate of the protocol, it introduces a strict requirement: the random sign b must be formally verified (as detailed in Section 3.3.2). First, the verification relation is updated to account for this sign, logically becoming $F_i = z_i - b\langle \mathbf{r}_i, \mathbf{s} \rangle - y_i = 0$. Subsequently, to mathematically integrate this sign into the protocol's equations, the prover must commit to the value b by encoding it into a polynomial $o(X) \in \mathcal{R}_q$, embedding it within the message vector of the BDLOP commitment.

To ensure mathematical soundness, the verifier requires proof of two properties: the polynomial $o(X)$ must consist solely of a constant term, and this constant term must be strictly 1 or -1. The binary sign condition is enforced by proving the quadratic relation $o(X)^2 - 1 = 0$.

However, this relation alone is insufficient, as it does not inherently restrict $o(X)$ to degree zero within the quotient ring. To demonstrate that $o(X)$ is strictly a constant polynomial, the protocol must prove that all its non-constant coefficients are zero.

To achieve this, the framework exploits polynomial multiplication over $\mathcal{R}_q$ to cyclically shift the coefficients. By multiplying $o(X)$ by the inverse monomial $X^{-i} \equiv -X^{d-i}$, the original i-th coefficient is algebraically shifted to the constant term position. Therefore, the protocol defines the polynomial relation $G_i(X) = -X^{d-i} \cdot o(X)$ for all $i \in \{1, \ldots, d-1\}$.

To prove that the non-constant coefficients of $o(X)$ are null, the prover must demonstrate that the constant term of each polynomial $G_i(X)$ evaluates to exactly 0. These structural constraints are integrated into the protocol's aggregated quadratic verification framework.

Both the sign-check relation $o(X)^2 - 1 = 0$ and the degree-check relations (enforcing a null constant term for every $G_i(X)$) are appended to the main set of quadratic equations f_i evaluated by the verifier.

3.7 Summary of the ℓ_2 Norm Verification Protocol

This section synthesizes the previously introduced components to construct a unified protocol, demonstrating that a prover holds a secret vector satisfying the bound $\|\mathbf{s}_1\|_2^2 \leqslant \beta^2$, without compromising the Zero-Knowledge property. The complete interactive sequence is represented in Figures 3.2 and 3.3, and can be summarized into the following distinct phases:

1. **Preparation of the Elements:** First, the prover computes the slack variable representing the difference $\beta^2 - \|\mathbf{s}_1\|_2^2$. This binary representation, denoted as δ, is concatenated with the original secret vector $\mathbf{s}_1$ to form an extended secret $\mathbf{s}_1^*$, as detailed in Section 3.6.1. Furthermore, all secondary secret elements required for the intermediate steps, specifically the random sign b, the Gaussian masking vector $\mathbf{y}_3$, and the zero-constant polynomials $\mathbf{g}$, are structurally aggregated into the message vector $\mathbf{m} = [b \parallel \mathbf{y}_3^T \parallel \mathbf{g}^T]^T$.

2. **Approximate Range Proof:** The protocol begins by ensuring that the evaluated ℓ_2 norm does not wrap around modulo q. The prover commits to the masking vector $\mathbf{y}_3$ and the random sign b. Upon receiving the challenge matrix $\mathbf{R}$ from the verifier, the prover calculates the masked vector $\mathbf{z}$ and applies the Rej_0 rejection sampling algorithm to guarantee that no statistical information is leaked, as explained in Section 3.6.2.

3. **Zero-Constant Polynomial Proof:** The protocol proceeds to construct the relations requiring a vanishing constant term (Sections 3.4.1 and 3.4.2). The prover samples a set of masking polynomials $\mathbf{g}$ whose constant terms are strictly zero. After the verifier extracts and sends multiple challenges from $\mathbb{Z}_q$, the prover integrates the following constraints:

   - The ℓ_2 norm evaluation ($\mathbf{s}_1^T\mathbf{s}_1 + \eta^T\delta - \beta^2 = 0$), explained in Sections 3.4 and 3.6.1.
   - The binary structure of the slack variable δ, explained in Section 3.6.1.
   - The correct geometric formation of the compressed vector $\mathbf{z}$, explained in Section 3.6.2.
   - The degree restrictions ensuring that the sign component b is strictly a constant term, explained in Section 3.6.2.

   These equations are algebraically aggregated using the mask g_j to yield the transmission polynomials h_j.

4. **Global Aggregation:** The verifier supplies a new set of polynomial challenges μ_j, which the prover utilizes to collapse the entire system of $F_{i,j}$ equations, the h_j components, and the binary sign constraint $(b-1)(b+1) = 0$ into a single, comprehensive quadratic function $f = 0$, as explained in Section 3.5.2.

5. **Formulating the Quadratic Relation:** This aggregated function is explicitly reformulated into a generalized matrix structure $f \equiv (\mathbf{s}^*)^T\mathbf{R}_2\mathbf{s}^* + \mathbf{r}_1^T\mathbf{s}^* + r_0 = 0$ over the fully extended secret block $\mathbf{s}^*$, which is formed by concatenating $\mathbf{s}_1^*$ and the newly generated $\mathbf{m}$. This is achieved by computing the specific matrix $\mathbf{R}_2$, the vector $\mathbf{r}_1$, and the scalar r_0.

6. **Quadratic Relation Masking:** Treating the aggregated equation as a standard LNP quadratic relation, the prover can now employ the core protocol represented in Figure 3.1 and explained in Section 3.5 to mathematically demonstrate the validity of the overarching equation.

7. **Verification:** At the conclusion of the protocol, the verifier receives the masked terms $\mathbf{z}_1$, $\mathbf{z}_2$, and $\mathbf{z}^*$. The verifier performs the standard checks outlined in Figure 3.1 (confirming that the transmitted vectors satisfy the expected norm bounds, verifying the algebraic validity of the ABDLOP commitment, and evaluating the public masked quadratic equation using f_{eval} and $\mathbf{z}^*$). Furthermore, the verifier executes additional security checks, specifically verifying the small norm bound on the compressed masked term $\mathbf{z}$ and ensuring that the constant term of each polynomial h_j evaluates exactly to zero.

By successfully completing this sequence, the verifier is mathematically assured that the underlying secret possesses a genuinely small ℓ_2 norm, while the prover maintains perfect zero-knowledge across all components of the extended secret.

3.8 Parameter Instantiation

3.8.1 Challenge Space $\mathcal{C}^*$

In the previously defined protocol, the challenge space from which the verifier samples is denoted as $\mathcal{C}^*$. This space is strictly restricted to polynomials $c \in \mathcal{R}_q$ possessing small coefficients that satisfy the automorphism invariance $\sigma_{-1}(c) = c$. It is crucial to formally define the geometric bounds of these coefficients.

When a secret vector $\mathbf{r}$ is masked by a challenge c, the norm of the resulting product $\|c\mathbf{r}\|$ must be tightly controlled. This expansion is mathematically governed by Lemma 2.15 of the LNP framework [LNP22], which states:

Lemma 3.8.1. Let $\mathbf{r} \in \mathcal{R}^\ell$ and $c \in \mathcal{R}$. Then, for any power-of-two k, we have:

$$\|c\mathbf{r}\| \leqslant \sqrt[2k]{\|\sigma_{-1}(c^k)c^k\|_1} \cdot \|\mathbf{r}\|. \tag{3.39}$$

Public Information:
- Polynomial ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ and scalar bound $\beta^2 \in \mathbb{N}$.
- Weighting polynomial $\eta(X) = \sum_{i=0}^{\kappa-1} 2^i X^{d-1-i} \in \mathcal{R}_q$.
- Base public commitment matrices: $\mathbf{A}_1^* \in \mathcal{R}_q^{n \times m_1^*}, \mathbf{A}_2 \in \mathcal{R}_q^{n \times m_2}, \mathbf{B} \in \mathcal{R}_q^{\ell \times m_2}$.
- Auxiliary public matrices/vectors for intermediate commitments: $\mathbf{B}_y, \mathbf{B}_g, \mathbf{b}_b$, such that $\mathbf{B} = [\mathbf{b}_b \parallel \mathbf{B}_y \parallel \mathbf{B}_g]^T$.
- Standard deviations for rejection sampling: $\mathfrak{s}_1, \mathfrak{s}_2, \mathfrak{s}_3$.
- Evaluated ABDLOP commitments: $\mathbf{t}_A = \mathbf{A}_1^* \mathbf{s}_1^* + \mathbf{A}_2 \mathbf{s}_2$ and $\mathbf{t}_B = \mathbf{B}\mathbf{s}_2 + \mathbf{m}$.

Private Information (Prover's Witness):
- Base secret vector $\mathbf{s}_1$ satisfying the bound $\|\mathbf{s}_1\|_2^2 \leqslant \beta^2$.
- Binary slack variable δ encoding the difference $\beta^2 - \|\mathbf{s}_1\|_2^2$.
- Extended secret vector $\mathbf{s}_1^* = [\mathbf{s}_1 \parallel \delta]^T \in \mathcal{R}_q^{m_1^*}$.
- Commitment randomness vector $\mathbf{s}_2 \leftarrow \chi^{m_2}$ (short elements).
- Composite message vector $\mathbf{m} = [b \parallel \mathbf{y}_3^T \parallel \mathbf{g}^T]^T \in \mathcal{R}_q^{\ell}$ aggregating the intermediate elements.

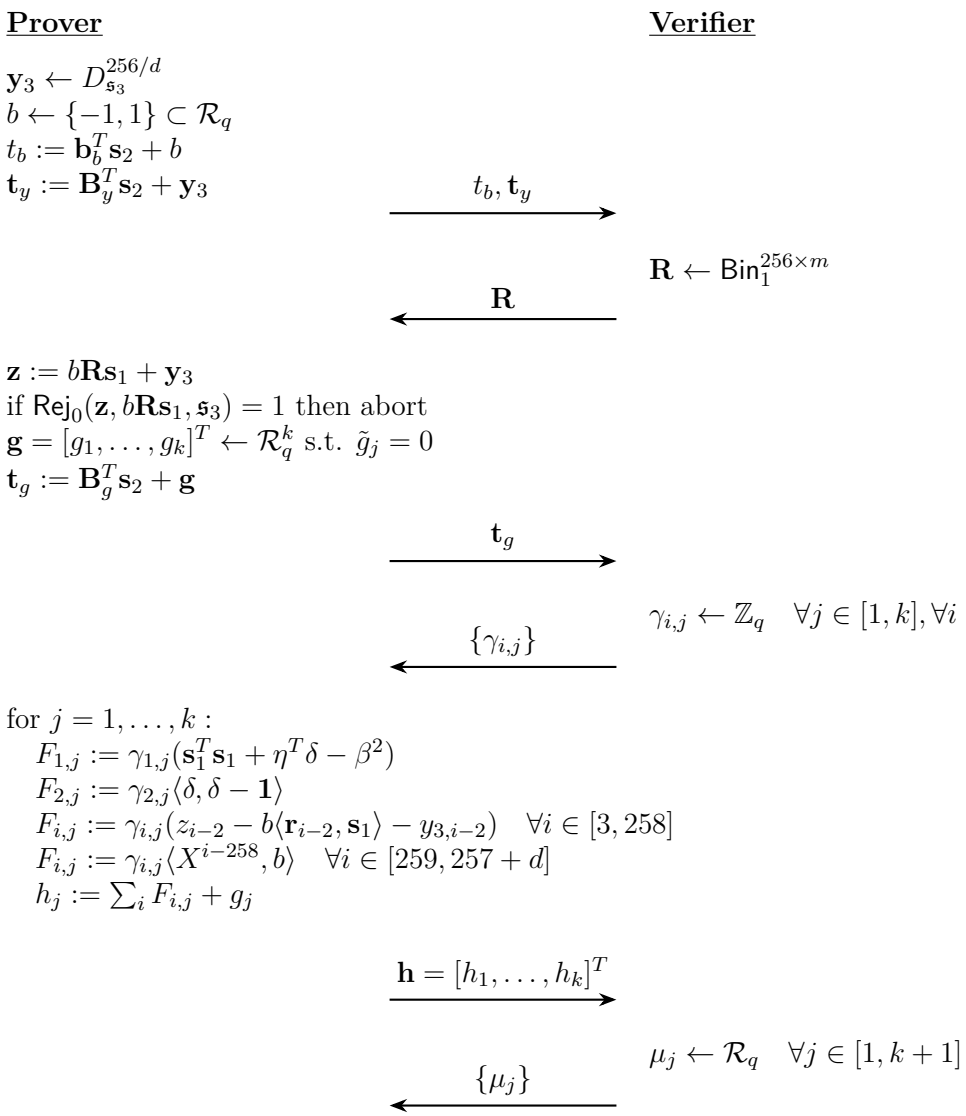

Figure 3.2. Full Interactive Protocol for ℓ_2-norm bound verification under the LNP framework (Part 1).

Prover | **Verifier**

$f := \sum_{j=1}^{k} \mu_j \left(\sum_i F_{i,j} - h_j\right) + \mu_{k+1}(b-1)(b+1)$

$\mathbf{m} := [b \parallel \mathbf{y}_3^T \parallel \mathbf{g}^T]^T$

$\mathbf{s}^* := \begin{bmatrix}(\sigma^i(\mathbf{s}_1^*))\\(\sigma^i(\mathbf{m}))\end{bmatrix}$

Represent f in matrix form:
$f \equiv (\mathbf{s}^*)^T\mathbf{R}_2\mathbf{s}^* + \mathbf{r}_1^T\mathbf{s}^* + r_0 = 0$

$\mathbf{y}_1 \leftarrow D_{\mathfrak{s}_1}^{m_1^*}$
$\mathbf{y}_2 \leftarrow D_{\mathfrak{s}_2}^{m_2}$
$\mathbf{w} := \mathbf{A}_1^*\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$
$\mathbf{y} := \begin{bmatrix}(\sigma^i(\mathbf{y}_1))\\-(\sigma^i(\mathbf{B}\mathbf{y}_2))\end{bmatrix}$

$g_1 := (\mathbf{s}^*)^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s}^* + \mathbf{r}_1^T\mathbf{y}$
$t := \mathbf{b}^T\mathbf{s}_2 + g_1$
$v := \mathbf{y}^T\mathbf{R}_2\mathbf{y} + \mathbf{b}^T\mathbf{y}_2$

$\xrightarrow{\mathbf{w}, t, v}$

$c \leftarrow \mathcal{C}^*$

$\xleftarrow{c}$

$\mathbf{z}_1 := c\mathbf{s}_1^* + \mathbf{y}_1$
$\mathbf{z}_2 := c\mathbf{s}_2 + \mathbf{y}_2$
for $i = 1, 2$:
  if $\mathsf{Rej}_1(\mathbf{z}_1, c\mathbf{s}_1^*, \mathfrak{s}_1) = 1$ or $\mathsf{Rej}_2(\mathbf{z}_2, c\mathbf{s}_2, \mathfrak{s}_2) = 1$
    then $\mathbf{z}_1, \mathbf{z}_2 := \perp$

$\xrightarrow{\mathbf{z}_1, \mathbf{z}_2}$

$\mathbf{t}_B := [t_b \parallel \mathbf{t}_y^T \parallel \mathbf{t}_g^T]^T$

$\mathbf{z}^* := \begin{bmatrix}(\sigma^i(\mathbf{z}_1))\\(\sigma^i(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2))\end{bmatrix}$

$f_{eval} := ct - \mathbf{b}^T\mathbf{z}_2$
Accept iff
  $\tilde{h}_j = 0 \quad \forall j \in [1, k]$ and
  $\|\mathbf{z}\| \leqslant \mathfrak{s}_3\sqrt{512}$ and
  $\|\mathbf{z}_1\| \leqslant \mathfrak{s}_1\sqrt{2m_1^*d}$ and
  $\|\mathbf{z}_2\| \leqslant \mathfrak{s}_2\sqrt{2m_2d}$ and
  $\mathbf{A}_1^*\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A$ and
  $(\mathbf{z}^*)^T\mathbf{R}_2\mathbf{z}^* + c\mathbf{r}_1^T\mathbf{z}^* + c^2r_0 - f_{eval} = v$

Figure 3.3. Full Interactive Protocol for ℓ_2-norm bound verification under the LNP framework (Part 2).

Another fundamental requirement to guarantee the Knowledge Soundness of the zero-knowledge proof is that the difference between any two distinct challenges, $c_1 - c_2$, must be an invertible element in the ring $\mathcal{R}_q$ for all $c_1, c_2 \in \mathcal{C}^*$. To ensure this invertibility, we rely on Lemma 2.6 from [LNP22] and its subsequent corollary:

Lemma 3.8.2. Let $p \equiv 5 \pmod 8$ be a prime. Take any $c \in \mathcal{R}_p$ such that $\sigma_{-1}(c) = c$. Then, c is invertible over $\mathcal{R}_p$ if and only if $c \neq 0$.

Corollary 3.8.1. Let $q = \prod_{i=1}^{n} q_i$ be the prime factorization of the modulus q. Denote $\mathcal{R}_q^\times$ as the set of invertible polynomials in $\mathcal{R}_q$. A polynomial f is invertible in $\mathcal{R}_q$ if and only if for each $i \in \{1, \ldots, n\}$, $f \pmod{q_i}$ is invertible in $\mathcal{R}_{q_i}$. Hence, following Lemma 3.8.2, if $f \in \mathcal{R}_q$ satisfies $0 < \|f\|_\infty < q_1$ (to explicitly prevent wrap-around modulo the smallest prime factor q_1) and $\sigma_{-1}(f) = f$, then $f \in \mathcal{R}_q^\times$.

To apply Lemma 3.8.1, we fix a power-of-two k and formally define the challenge space $\mathcal{C}^*$ as:

$$\mathcal{C}^* := \left\{ c \in S_\kappa^\sigma \;\middle|\; \sqrt[2k]{\|\sigma_{-1}(c^k)c^k\|_1} \leqslant \eta \right\}, \tag{3.40}$$

where the symmetric and bounded subsets are defined as:

$$S_\kappa^\sigma := \{c \in S_\kappa \mid \sigma_{-1}(c) = c\} \quad \text{and} \quad S_\kappa = \{x \in \mathcal{R}_q \mid \|x\|_\infty \leqslant \kappa\}. \tag{3.41}$$

To guarantee the invertibility of the challenge difference $c_1 - c_2$, Corollary 3.8.1 strictly requires that $\|c_1 - c_2\|_\infty < q_1$. Given that both $c_1, c_2 \in S_\kappa$, their maximum possible difference is tightly bounded by 2κ. Consequently, the protocol must strictly enforce the condition $2\kappa < q_1$, yielding the constraint $\kappa < q_1/2$.

During execution, to sample a valid challenge from $\mathcal{C}^*$, the verifier samples a uniformly random polynomial $c \in S_\kappa^\sigma$ and checks whether it satisfies the expansion bound $\sqrt[2k]{\|\sigma_{-1}(c^k)c^k\|_1} \leqslant \eta$. Because this verification is performed dynamically (often via rejection), the parameter k must be kept relatively small to maintain computational efficiency. The chosen threshold η represents the maximum expanding factor of the norm of the secret $\mathbf{r}$, and it plays a critical role in evaluating the standard deviation for the final geometric acceptance bounds of the entire protocol.

A practical instantiation of these parameters, as proposed in the LNP framework [LNP22] for a setting where $q_1 > 8$ and $k = 32$, is detailed in Table 3.1:

d	κ	η	$\lvert S_\kappa^\sigma \rvert$	$\lvert \mathcal{C}^* \rvert$
128	2	59	2^{148}	2^{147}

Table 3.1. Parameters for the challenge space $\mathcal{C}^*$.

3.8.2 Standard Deviation for Rejection Sampling with Polynomial Challenges

To determine the appropriate standard deviation for the rejection sampling algorithm, we recall the conditions established in Section 3.3.2. For a fixed target rejection rate (defined by the expected number of iterations M), the required standard deviation $\mathfrak{s}$ is chosen to be proportional to an upper bound of the norm of the shift vector $\mathbf{v} = c\mathbf{r}$. This yields a relation of the form $\mathfrak{s} \approx \gamma \max \|c\mathbf{r}\|$, where the constant γ depends on the specific rejection algorithm employed (we recall $\gamma \approx 11$ for Rej_1 with $M = 3$, and $\gamma \approx 0.675$ for both Rej_2 with $M = 6$ and Rej_0 with $M = 3$).

To correctly parameterize the standard deviation, the secret's maximum norm is bounded by its theoretical upper limit. By applying the bound from Lemma 3.8.1 and the definition of the challenge space $\mathcal{C}^*$, the coefficient expansion is strictly limited by η. If the ℓ_2 norm of the secret vector is bounded such that $\|\mathbf{r}\|_2 \leqslant \alpha$, the standard deviation is statically instantiated as $\mathfrak{s} = \gamma \cdot \eta \cdot \alpha$. Alternatively, if the secret elements have an ℓ_∞ constraint, such that $\|\mathbf{r}\|_\infty \leqslant v$ for a vector $\mathbf{r} \in \mathcal{R}_q^L$, the corresponding ℓ_2 norm is bounded by $v\sqrt{Ld}$, leading to a standard deviation of:

$$\mathfrak{s} = \gamma \cdot \eta \cdot v\sqrt{Ld}. \tag{3.42}$$

We now evaluate these geometric bounds within the context of the final ℓ_2 norm verification protocol, summarized in Section 3.7 and visually detailed in Figures 3.2 and 3.3.

The first masking vector $\mathbf{y}_1$ must hide the extended secret $\mathbf{s}_1^*$. This vector is formed by concatenating the base secret $\mathbf{s}_1$ (whose ℓ_2 norm is bounded by β) and the binary slack variable δ. The composite norm is therefore bounded by $\|\mathbf{s}_1^*\|_2 \leqslant \beta$. Consequently, the standard deviation $\mathfrak{s}_1$ is instantiated as:

$$\mathfrak{s}_1 = \gamma \cdot \eta\beta. \tag{3.43}$$

The second masking vector $\mathbf{y}_2$ hides the commitment randomness $\mathbf{s}_2$. Since $\mathbf{s}_2$ is a vector of length m_2 sampled from a uniform distribution in the range $[-v, v]$, its standard deviation $\mathfrak{s}_2$ is derived using the ℓ_∞ conversion:

$$\mathfrak{s}_2 = \gamma \cdot \eta \cdot v\sqrt{m_2 d}. \tag{3.44}$$

3.8.3 Standard Deviation for Rejection Sampling to Prevent Modular Wrap-Around

To determine the standard deviation $\mathfrak{s}_3$, a different analytical approach is required compared to $\mathfrak{s}_1$ and $\mathfrak{s}_2$. In this phase of the protocol, the masking operation does not involve a polynomial challenge c, but rather a projection matrix $\mathbf{R}$, as detailed in Section 3.6.2. To evaluate the norm expansion caused by this matrix multiplication, we refer to Lemma 2.8

from the LNP framework [LNP22]:

Lemma 3.8.3. Let ω denote the parameter of the binary distribution. Under the heuristic substitution of Bin_ω with the normal distribution of variance $\omega/2$, for any $\mathbf{w} \in \mathbb{Z}^m$:

$$\Pr_{\mathbf{R} \leftarrow \mathrm{Bin}_\omega^{256 \times m}} \left[\|\mathbf{Rw}\|^2 > 337\omega \|\mathbf{w}\|^2 \right] \approx \Pr_{y \leftarrow \chi^2(256)} [y > 674] \leqslant 2^{-128}. \tag{3.45}$$

Lemma 3.8.3 guarantees that, with overwhelming probability, the squared norm of the projected element does not exceed 337ω times the squared norm of the original secret vector. Since the ℓ_2 norm of the base secret $\mathbf{s}_1$ is bounded by β, the shift vector $\mathbf{v} = \mathbf{Rs}_1$ is bounded by $\|\mathbf{v}\| \leqslant \beta\sqrt{337\omega}$. Applying the fundamental rejection sampling relation, where the standard deviation scales proportionally to the maximum expected shift ($\mathfrak{s} \approx \gamma \max \|\mathbf{v}\|$), we instantiate $\mathfrak{s}_3$ as:

$$\mathfrak{s}_3 = \gamma\beta\sqrt{337\omega}. \tag{3.46}$$

Building upon this parameter, the authors of the LNP framework [LNP22] demonstrate that the protocol achieves theoretical soundness by imposing the acceptance bound $b \leqslant 189\gamma\beta$ within the wrap-around checks (as referenced in Lemma 3.6.1). Consequently, the protocol does not prove that the underlying secret norm is exactly bounded by β; rather, it proves that the actual norm is bounded by the relaxed factor $189\gamma\beta$. This gap is an inherent and accepted tradeoff in approximate range proofs, balancing exactness with computational efficiency.

3.9 Formal Security Guarantees and Parameter Constraints

Throughout this chapter, the protocol has been meticulously constructed to ensure that no statistical information about the secret is leaked and that a malicious prover cannot forge a valid proof. To formalize these properties and justify the concrete parameter instantiation provided in Table 3.2, we report the main security theorem from the LNP framework [LNP22].

Theorem 3.9.1 (Formal Security Guarantees [LNP22])**.** Consider an ABDLOP commitment to a message $(\mathbf{s}_1, \mathbf{m}) \in \mathcal{R}_q^{m_1+\ell}$ with randomness $\mathbf{s}_2 \in \mathcal{R}_q^{m_2}$ satisfying $\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2 = \mathbf{t}_A$, $\mathbf{Bs}_2 + \mathbf{m} = \mathbf{t}_B$. Let $B := 2\sqrt{\frac{256}{26}} t\gamma_3\sqrt{337}\beta$, where β is the ℓ_2 bound on the base secret and γ_3 is the rejection sampling constant used in the Approximate Range Proof. Assume that $t \geqslant 1.64$ and the parameters satisfy:

$$B < \frac{q}{41c_{max}} \tag{3.47}$$

$$B^2 + B\sqrt{N_{\text{total}} \cdot d} < q, \tag{3.48}$$

$$2\beta^2 + B^2 - 1 < q. \tag{3.49}$$

where c_{max} is the maximum absolute coefficient of any challenge in $\mathcal{C}^*$, and N_{total} is the total number of polynomial blocks composing the extended secret $\mathbf{s}^*$.

Then the interactive procedure detailed in Figures 3.2 and 3.3 constitutes a computationally secure commit-and-prove protocol, functioning as a Zero-Knowledge Argument of Knowledge. Specifically, it achieves:

- **Completeness:** If both parties follow the protocol honestly and $t = 1.64$, the verifier shall accept with a probability predominantly determined by the expected acceptance rates of the employed rejection sampling algorithms, approximated as:

$$\approx P_{\text{eval}} \exp\left(-\sum_i \frac{1}{2\gamma_i^2}\right). \tag{3.50}$$

- **Soundness:** For any adversarial probabilistic prover operating with a success probability $\epsilon \geqslant \frac{2}{|\mathcal{C}^*|} - q^{-d/2} - q^{-\lambda} - 2^{-127}$, there exists an efficient Extractor that, via rewindable black-box access, yields a mathematically valid opening of the underlying ABDLOP commitment in expected polynomial time. The difficulty of producing a fake proof reduces directly to the underlying Module-SIS (M-SIS) hardness assumption.

- **Zero-Knowledge:** The protocol is formally simulatable without access to the private witness $(\mathbf{s}_1, \mathbf{m})$. Any probabilistic polynomial-time adversary capable of distinguishing a real transcript from a simulated one with advantage ϵ can be utilized to break the Extended Module-LWE assumption with an advantage proportional to $\epsilon/2 - 2^{-100}$.

The three geometric inequalities outlined in Theorem 3.9.1 are not arbitrary; they strictly dictate the parameter selection to ensure that the algebraic relations evaluated modulo q hold true natively over the integers $\mathbb{Z}$. Specifically:

- **Projection Bound** ($B < \frac{q}{41c_{max}}$): This condition ensures that the maximum proof bound remains sufficiently small relative to the modulus q and the challenge magnitude c_{max}. It mathematically prevents any modular wrap-around during the verification of the approximate range proof (fulfilling the requirements of Lemma 3.6.1).

- **Binary Structure Bound** ($B^2 + B\sqrt{N_{\text{total}} \cdot d} < q$): This inequality guarantees that when the verifier evaluates the inner products to enforce binary constraints (such as checking the slack variable $\langle \delta, \delta - \mathbf{1} \rangle = 0$), the algebraic sum of the extracted coefficients does not exceed the modulus. This ensures the binary constraint is valid over $\mathbb{Z}$.

- **Norm Evaluation Bound** ($2\beta^2 + B^2 - 1 < q$): This ensures that the overarching quadratic relation linking the base secret's ℓ_2 norm to the binary components (e.g., $\mathbf{s}_1^T \mathbf{s}_1 + \eta^T \delta - \beta^2 = 0$) does not suffer from modular overflow. Consequently, the verifier can conclude with mathematical certainty that the geometric relation is exact.

Constraining the Masking Variances to Prevent Wrap-Around

The masking of $\mathbf{z}_1$ and $\mathbf{z}_2$ needs to guarantee that the verification equations hold over the integers (thereby avoiding modulo wrap-around). To achieve this, the standard deviations $\mathfrak{s}_1$ and $\mathfrak{s}_2$ must be carefully chosen so that the maximum extraction bound B remains strictly smaller than the system modulus q. Here, B aggregates the verifier's acceptance thresholds for both the witness masking vector ($B_1 := \mathfrak{s}_1\sqrt{2m_1d}$) and the randomness masking vector ($B_2 := \mathfrak{s}_2\sqrt{2m_2d}$). This aggregation is mathematically required because the separate commitment relations are algebraically unified into a single Module-SIS problem during the soundness proof. By concatenating the isolated secrets into one composite solution vector, the maximum geometric length corresponds precisely to the Euclidean norm of the combined parts ($\sqrt{B_1^2 + B_2^2}$).

According to Lemma 5.2 in [BNLS23], this aggregated bound is evaluated as:

$$B = 4\eta\sqrt{2d\left(\mathfrak{s}_1^2 m_1 + \mathfrak{s}_2^2 m_2\right)} \tag{3.51}$$

where η represents the maximum norm expansion factor introduced by the polynomial challenge c.

Consequently, to definitively prevent overflow and reduce the scheme's soundness to the hardness of the Module-SIS problem, the chosen variances must satisfy the following strict parameter inequality with respect to q:

$$4\eta\sqrt{2d\left(\mathfrak{s}_1^2 m_1 + \mathfrak{s}_2^2 m_2\right)} < q \tag{3.52}$$

Impact of Standard Deviation and Rejection Constants on Efficiency.

The standard deviations $(\mathfrak{s}_1, \mathfrak{s}_2, \mathfrak{s}_3)$ and their underlying rejection constants γ directly determine the protocol's communication complexity and parameter sizing. These values must be minimized for two fundamental structural reasons.

First, the constant γ dictates a strict trade-off between the prover's computational overhead and the final proof size. While a larger γ reduces the expected number of iterations M, it linearly inflates the theoretical bound B. As formalized in Theorem 3.9.1, B appears as a quadratic term in the geometric inequalities. An increase in γ therefore causes a quadratic growth in these bounds, requiring a proportionally larger modulus q to prevent modular wrap-around over $\mathbb{Z}$. Since a larger q increases the bit-length of every polynomial coefficient, the framework prioritizes a minimal γ (accepting slightly higher rejection rates, e.g., $M \approx 3$ or 7) to preserve a tight modulus and restrict the overall proof size.

Second, smaller standard deviations inherently generate final masked response vectors $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z})$ whose coefficients are tightly concentrated around zero. This narrow statistical distribution allows for highly effective data compression prior to transmission. Consequently, minimizing both γ and the associated standard deviations is essential to achieve a bandwidth-efficient transcript.

Concrete Parameter Instantiation.

Table 3.2 summarizes a concrete instantiation of the entire structural and geometric parameter set, based on the parameters provided by the LNP framework (Parameter Set II) [LNP22].

Parameter	Description	Value
q	Proof system modulus	$2^{31} - 305215$
d	Dimension of the polynomial ring $\mathcal{R}$	128
l	Number of factors $X^d + 1$ splits into mod q	2
γ_1	Rejection sampling constant for $c\mathbf{s}_1$	16
γ_2	Rejection sampling constant for $c\mathbf{s}_2$	1.5
γ_3	Rejection sampling constant for Approx range proof	1.8
c_{max}	Maximum coefficient of a challenge in $\mathcal{C}$	2
n	Height of $\mathbf{A}_1, \mathbf{A}_2$ in ABDLOP	9
m_1	Length of the base message $\mathbf{s}_1$	6
λ	$2\times$ (number of masking polynomials g_j)	12
m_2	Length of commitment randomness $\mathbf{s}_2$	29
	Repetition rate	7
	Ciphertext size	1 KB
	Commitment + proof size	18.6 KB

Table 3.2. Parameter selection, ciphertext and proof sizes for verifiable encryption (Set II).

3.10 Non-Interactive Zero-Knowledge via Fiat-Shamir

The protocol illustrated in Figures 3.2 and 3.3 is formulated as an interactive proof system requiring multiple communication rounds between two distinct entities: the prover and the verifier. A high round complexity significantly degrades practical performance, forcing both parties to remain online simultaneously throughout the execution. To use this zero-knowledge proof to generate a credential presentation and attach it as a static digital signature, the protocol must be transformed into a Non-Interactive Zero-Knowledge (NIZK) architecture.

This conversion is achieved using the Fiat-Shamir Transformation [FS86]. This technique allows the elimination of the interactive verifier by enabling the prover to locally compute the unpredictable challenges.

To better conceptualize how this cryptographic conversion operates in practice, consider its application on a trivially simplified, three-round interactive proof system:

1. The prover generates a random commitment t and sends it to the verifier.

2. The verifier responds with a uniformly random challenge c.

3. The prover computes the response z using the secret, the commitment randomness, and the challenge c, outputting the transcript (t, c, z).

Under the Fiat-Shamir transformation, the verifier's role in step 2 is entirely replaced by a cryptographic hash function H. The prover computes the challenge locally as $c \leftarrow H(\text{statement}, t)$. By hashing the public statement alongside the commitment t, the challenge becomes irrevocably tied to the prover's specific execution. The final non-interactive proof becomes simply $\pi = (t, z)$, and any verifier can later check the proof by recomputing $c = H(\text{statement}, t)$ and verifying z.

In theoretical cryptography, the Fiat-Shamir transformation is proven secure in the *Random Oracle Model* (ROM). In a real-world deployment, this theoretical construct cannot be instantiated in practice. Therefore, the transformation relies on the *Fiat-Shamir Heuristic*, which replaces the random oracle with a secure, collision-resistant cryptographic hash function (such as SHA-3 or SHAKE256).

Because a secure hash function provides pre-image resistance and pseudo-random unpredictability, it is computationally infeasible for a malicious prover to manipulate the commitment t to obtain an advantageous challenge c that would allow them to forge a proof.

By applying this transformation to our ℓ_2-norm protocol, the prover sequentially computes all intermediate commitments and uses them as inputs to hash functions to generate the challenges $(\mathbf{R}, \gamma, \mu, c)$. At the end of the execution, the prover collects all the required components into a single static proof transcript π, which is transmitted to the verifier. The verifier accepts the proof by recomputing the challenges using the elements in π and checking if the final equations are satisfied.

Under this transformation, the fully non-interactive protocol is represented in Figure 3.4.

Non-Interactive Zero-Knowledge Protocol

Prover (Offline Execution)

Repeat:
$\mathbf{y}_3 \leftarrow D_{\mathfrak{s}_3}^{256/d}$
$b \leftarrow \{-1, 1\} \subset \mathcal{R}_q$
$t_b := \mathbf{b}_b^T \mathbf{s}_2 + b, \quad \mathbf{t}_y := \mathbf{B}_y^T \mathbf{s}_2 + \mathbf{y}_3$
$\mathbf{R} \leftarrow H_1(\text{seed}, t_b, \mathbf{t}_y)$
$\mathbf{z} := b\mathbf{R}\mathbf{s}_1 + \mathbf{y}_3$
$\mathbf{g} = [g_1, \ldots, g_k]^T \leftarrow \mathcal{R}_q^k$ s.t. $\tilde{g}_j = 0$
$\mathbf{t}_g := \mathbf{B}_g^T \mathbf{s}_2 + \mathbf{g}$
$\{\gamma_{i,j}\} \leftarrow H_2(\text{seed}, t_b, \mathbf{t}_y, \mathbf{t}_g)$
Compute $F_{i,j}$ and $h_j := \sum_i F_{i,j} + g_j$
$\{\mu_j\} \leftarrow H_3(\text{seed}, t_b, \mathbf{t}_y, \mathbf{t}_g, \mathbf{h})$
Compute $f = 0$ mapping to $(\mathbf{s}^*)^T \mathbf{R}_2 \mathbf{s}^* + \mathbf{r}_1^T \mathbf{s}^* + r_0$
$\mathbf{m} := [b \parallel \mathbf{y}_3^T \parallel \mathbf{g}^T]^T$
$\mathbf{t}_B := \mathbf{B}\mathbf{s}_2 + \mathbf{m}$
$\mathbf{y}_1 \leftarrow D_{\mathfrak{s}_1}^{m_1^*}, \quad \mathbf{y}_2 \leftarrow D_{\mathfrak{s}_2}^{m_2}$
$\mathbf{w} := \mathbf{A}_1^* \mathbf{y}_1 + \mathbf{A}_2 \mathbf{y}_2$
$\mathbf{y} := \begin{bmatrix} (\sigma^i(\mathbf{y}_1)) \\ -(\sigma^i(\mathbf{B}\mathbf{y}_2)) \end{bmatrix}$
Compute t, v from $\mathbf{y}, \mathbf{R}_2, \mathbf{s}^*$
$c \leftarrow H_4(\text{seed}, t_b, \mathbf{t}_y, \mathbf{t}_g, \mathbf{h}, \mathbf{w}, t, v) \in \mathcal{C}^*$
$\mathbf{z}_1 := c\mathbf{s}_1^* + \mathbf{y}_1, \quad \mathbf{z}_2 := c\mathbf{s}_2 + \mathbf{y}_2$
Evaluate rejections:
$b_0 := \mathsf{Rej}_0(\mathbf{z}, b\mathbf{R}\mathbf{s}_1, \mathfrak{s}_3)$
$b_1 := \mathsf{Rej}_1(\mathbf{z}_1, c\mathbf{s}_1^*, \mathfrak{s}_1)$
$b_2 := \mathsf{Rej}_2(\mathbf{z}_2, c\mathbf{s}_2, \mathfrak{s}_2)$
Until $b_0 = 0$ and $b_1 = 0$ and $b_2 = 0$.

Output $\pi = (t_b, \mathbf{t}_y, \mathbf{t}_g, \mathbf{h}, \mathbf{w}, t, v, \mathbf{z}, \mathbf{z}_1, \mathbf{z}_2)$

Verifier (π, seed)

Parse π to retrieve elements.
$\mathbf{R} \leftarrow H_1(\text{seed}, t_b, \mathbf{t}_y)$
$\{\gamma_{i,j}\} \leftarrow H_2(\text{seed}, t_b, \mathbf{t}_y, \mathbf{t}_g)$
$\{\mu_j\} \leftarrow H_3(\text{seed}, t_b, \mathbf{t}_y, \mathbf{t}_g, \mathbf{h})$
$\mathbf{t}_B := [t_b \parallel \mathbf{t}_y^T \parallel \mathbf{t}_g^T]^T$
$c \leftarrow H_4(\text{seed}, \mathbf{t}_B, \mathbf{h}, \mathbf{w}, t, v)$
$\mathbf{z}^* := \begin{bmatrix} (\sigma^i(\mathbf{z}_1)) \\ (\sigma^i(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2)) \end{bmatrix}$
$f_{eval} := ct - \mathbf{b}^T \mathbf{z}_2$

Accept iff:
$\tilde{h}_j = 0 \quad \forall j \in [1, k]$ and
$\|\mathbf{z}\| \leqslant \mathfrak{s}_3 \sqrt{512}$ and
$\|\mathbf{z}_1\| \leqslant \mathfrak{s}_1 \sqrt{2m_1^* d}$ and
$\|\mathbf{z}_2\| \leqslant \mathfrak{s}_2 \sqrt{2m_2 d}$ and
$\mathbf{A}_1^* \mathbf{z}_1 + \mathbf{A}_2 \mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A$ and
$(\mathbf{z}^*)^T \mathbf{R}_2 \mathbf{z}^* + c\mathbf{r}_1^T \mathbf{z}^* + c^2 r_0 - f_{eval} = v$

Figure 3.4. Non-Interactive Zero-Knowledge version of the protocol via Fiat-Shamir transformation.

Chapter 4

Construction of an Anonymous Credential System

We now explore a practical methodology to leverage the previously discussed Zero-Knowledge framework to construct a fully functional Anonymous Credential System. As introduced in Section 1.1, such a primitive fundamentally requires the development of two core algorithmic protocols: an issuance protocol, where an Issuer ($\mathcal{I}$) generates a credential for a Holder ($\mathcal{H}$) upon verifying specific identity conditions, and a verification protocol, executed by a Verifier ($\mathcal{V}$) to cryptographically validate the presentation of the credential. An approach to construct this architecture by directly building upon the computational structures of [LNP22] is detailed in the concrete framework proposed by Bootle, Lyubashevsky, Nguyen, and Sorniotti in [BNLS23].

Chapter Outline To systematically present this construction, the remainder of this chapter is structured as follows. First, Section 4.1 establishes the foundational cryptographic hardness assumptions underlying the system, focusing on the ISIS_f problem and its interactive variants. Section 4.2 details the credential issuance phase, formalizing the mechanisms for selective disclosure, concurrent security, and modulus lifting. Section 4.3 outlines the corresponding verification phase, presenting the complete interactive authentication protocol. Section 4.4 provides the concrete security parameters and bounds necessary to instantiate the scheme safely. Finally, the chapter concludes with a theoretical analysis of the architectural differences between the original LNP framework and the BLNS instantiation (Section 4.5), followed by an in-depth look at the algorithmic optimizations introduced by the LINKS Foundation to reduce the computational bottleneck of the rejection sampling phase (Section 4.6).

Main Mathematical Approach The core paradigm relies on establishing a computationally hard algebraic relation between the Issuer and the Holder that can be expressed

via linear or quadratic equations and natively checked by the Verifier.

During the issuance phase, the protocol must support selective disclosure and attribute masking. For instance, if a user needs to prove they are of legal age without revealing their exact birthdate, the specific birthdate attribute is kept strictly hidden inside a cryptographic commitment, while the boolean attribute certifying majority status can be disclosed in the clear.

Once the attributes are properly committed, the Issuer utilizes its trapdoor to sample a solution vector. The resulting short preimage, combined locally by the Holder with the commitment randomness, forms the final valid credential. When authenticating to a third party, the Holder does not reveal this signature; rather, they employ a Zero-Knowledge Proof of Knowledge to demonstrate possession of the solution to the underlying hard problem.

4.1 A Hard Linear Problem: The ISIS_f Assumption

To construct their credential system, the authors of [BNLS23] rely on a specific family of lattice-based hard problems, denoted as the $\mathsf{ISIS}_f(n, m, k)$ (Inhomogeneous Short Integer Solution) assumption.

Given a uniformly random $n \times m$ matrix $\mathbf{A}$ over $\mathcal{R}_q$ and an efficiently computable function $f : [N] \to \mathcal{R}_q^n$, the ISIS_f problem consists of finding a pair consisting of an element $x^* \in [N]$ and a short vector $\mathbf{s}^* \in \mathcal{R}_q^m$ such that $\mathbf{A}\mathbf{s}^* = f(x^*)$.

The adversary is also provided with k distinct valid pairs $(x_i, \mathbf{s}_i)$ that already satisfy the equation (where each $\mathbf{s}_i$ is a short preimage generated by the challenger). To win the experiment, the adversary must output a new, valid pair $(x^*, \mathbf{s}^*)$ such that the norm of $\mathbf{s}^*$ is strictly bounded ($0 < \|\mathbf{s}^*\| \leq \mathcal{B}$) and the pair is distinct from the k pairs previously received.

Definition 4.1.1 (The ISIS_f Problem)**.** Let $\mathsf{pp} := (q, d, n, m, N, k, \mathfrak{s}, \mathcal{B})$ be a tuple of functions of the security parameter λ. Consider any efficiently computable function $f : [N] \to \mathcal{R}_q^n$. The ISIS_f assumption is defined by the experiment $\mathsf{Exp}_{\mathsf{pp}}^{\mathsf{ISIS}_f}(\mathcal{A})$ for an adversary $\mathcal{A}$:

1. The challenger samples a uniformly random matrix $\mathbf{A} \leftarrow \mathcal{R}_q^{n \times m}$.
2. For $i \in [k]$:
   - Sample a random element $x_i \leftarrow [N]$.
   - Sample a short preimage $\mathbf{s}_i \leftarrow \mathbf{A}_{\mathfrak{s}}^{-1}(f(x_i))$, where $\mathbf{A}_{\mathfrak{s}}^{-1}$ denotes a trapdoor sampling algorithm that outputs a discrete Gaussian preimage with standard deviation $\mathfrak{s}$.
3. The adversary $\mathcal{A}$, given the matrix $\mathbf{A}$ and the k pairs $(x_i, \mathbf{s}_i)_{i \in [k]}$, outputs a candidate pair $(x^*, \mathbf{s}^*)$.
4. The experiment returns 1 (the adversary wins) if all the following conditions hold:

- $\mathbf{A}\mathbf{s}^* = f(x^*)$
- $0 < \|\mathbf{s}^*\| \leq \mathcal{B}$
- $(x^*, \mathbf{s}^*) \notin \{(x_i, \mathbf{s}_i) : i \in [k]\}$

Otherwise, it returns 0.

For an adversary $\mathcal{A}$, we define the advantage as:

$$\mathsf{Adv}_{\mathsf{pp}}^{\mathsf{ISIS}_f}(\mathcal{A}) = \Pr\left[\mathsf{Exp}_{\mathsf{pp}}^{\mathsf{ISIS}_f}(\mathcal{A}) \to 1\right] \tag{4.1}$$

The ISIS_f assumption states that for every Probabilistic Polynomial-Time (PPT) adversary $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathsf{pp}}^{\mathsf{ISIS}_f}(\mathcal{A})$ is negligible.

4.1.1 Choosing the Function f

To design a secure credential system using the LNP framework [LNP22], the underlying cryptographic assumption must be specifically tailored to the protocol's capabilities. Since the LNP framework is specifically built to efficiently prove only linear and quadratic relations, the cryptographic problem underlying the credential system must natively reduce to these exact algebraic types. To satisfy this strict requirement, the authors of [BNLS23] instantiate f by composing a binary encoding function with a linear matrix multiplication. This reduces the problem to evaluating a linear relation over a strictly binary domain. While finding a generic solution to a linear system over a ring is trivial, finding a specifically binary solution remains computationally hard. Crucially, proving knowledge of such a binary vector requires evaluating a linear equation alongside a quadratic constraint (to enforce the binary nature of the coefficients), directly matching the native capabilities of the LNP framework.

Formally, we define this specific instantiation, denoted as the $\mathsf{ISIS}_{\mathsf{bin}}$ problem, as follows:

Definition 4.1.2 (Binary Encoding Function f)**.** Let $t \in \mathbb{N}$ and let $B \in \mathbb{Z}_q^{nd \times t}$ be an arbitrary uniformly random matrix. The function $f : [N] \to \mathcal{R}_q^n$ is defined as:

$$f(x) := \mathsf{Coeffs}^{-1}(B \cdot \vec{\mathrm{enc}}(x)) \in \mathcal{R}_q^n \tag{4.2}$$

where $\vec{\mathrm{enc}}(x) \in \{0,1\}^t$ is the binary decomposition of $x - 1$ (which implies a domain size of $N = 2^t$).

Under this construction, proving knowledge of a preimage x such that $f(x) = \mathbf{y}$ is mathematically equivalent to proving knowledge of a binary scalar vector $\vec{u} \in \{0,1\}^t$ such that $B\vec{u} = \mathsf{Coeffs}(\mathbf{y})$. This exact relation is optimally supported by the linear proof systems introduced in [LNP22].

Because the $\mathsf{ISIS}_{\mathsf{bin}}$ assumption is relatively new, it does not yet benefit from decades of extensive cryptanalysis. Nevertheless, the authors in [BNLS23] carefully analyze standard

attack vectors against it. For instance, they demonstrate that fixing an element x to blindly find a short vector $\mathbf{s}^*$ such that $\mathbf{A}\mathbf{s}^* = f(x)$ simply degenerates into an Inhomogeneous M-SIS problem, which remains hard. Furthermore, they show how setting the norm bound $\mathcal{B}$ appropriately prevents adversaries from exploiting Integer Linear Programming (ILP) techniques to forge credentials.

4.1.2 The Interactive ISIS_f Assumption

The standard $\mathsf{ISIS}_{\mathsf{bin}}$ assumption is sufficient for static scenarios, but the anonymous credential system requires a dynamic approach to securely bind the user attributes ($\mathbf{m}$) and the commitment randomness ($\mathbf{r}$) to the underlying tuple $(x, \mathbf{s})$.

To accommodate this, the base algebraic relation is modified by introducing a uniformly random public matrix $\mathbf{C} \in \mathcal{R}_q^{n\times(\ell_m+\ell_r)}$. The target problem now requires finding short vectors such that:

$$\mathbf{A}\mathbf{s}^* = f(x^*) + \mathbf{C}\begin{bmatrix}\mathbf{m}^* \\ \mathbf{r}^*\end{bmatrix} \tag{4.3}$$

where the vectors $\mathbf{s}^*$, $\mathbf{m}^*$, and $\mathbf{r}^*$ are constrained by strict norm bounds, specifically $\|\mathbf{s}^*\| \leq \mathcal{B}_s$ and $\|(\mathbf{m}^*, \mathbf{r}^*)\| \leq \mathcal{B}_m$.

The basic $\mathsf{ISIS}_{\mathsf{bin}}$ assumption cannot directly guarantee the security of this system because the adversary interacts adaptively. An attacker can choose specific attributes and randomness vectors to query, effectively requesting targeted credentials to influence the assumption's security.

To formalize this dynamic behavior, the authors in [BNLS23] define the Interactive ISIS_f problem ($\mathsf{Int\text{-}ISIS}_f$). In this security model, the adversary is granted access to a preimage oracle $\mathcal{O}_{\text{pre}}$. Upon receiving adaptively chosen short vectors $(\mathbf{m}_i, \mathbf{r}_i)$, the oracle samples a random tag $x_i \leftarrow [N]$ and returns a short preimage $\mathbf{s}_i$ satisfying the linear relation. To successfully forge a credential, the adversary must produce a valid tuple $(x^*, \mathbf{s}^*, \mathbf{m}^*, \mathbf{r}^*)$ where the message vector $\mathbf{m}^*$ was never queried to the oracle (i.e., $\mathbf{m}^* \notin \{\mathbf{m}_1, \ldots, \mathbf{m}_k\}$).

Although the adversary gains the ability to make adaptive queries, Theorem 3.3 in [BNLS23] provides a strict mathematical reduction from the $\mathsf{Int\text{-}ISIS}_f$ problem to the standard ISIS_f problem. This reduction guarantees that, given the correct parameter instantiation, the interactive variant remains computationally as hard as the static ISIS_f assumption.

4.1.3 Trapdoor Generation

Having established the hardness of finding a valid pair $(x, \mathbf{s})$ that solves the ISIS_f problem, it is necessary to detail how an Issuer can construct a trapdoor to efficiently compute such solutions. This is achieved by leveraging the NTRU trapdoor generation mechanism, which relies on the same underlying structure used in the FALCON signature scheme [PFH+17]. The trapdoor generation algorithm ($\mathsf{NTRU.TrapGen}$) outputs a structured public matrix $\mathbf{A}$ alongside a correspondingly short, full-rank basis matrix $\mathbf{T}$ that serves as the trapdoor.

To generate a valid solution pair given the input vectors $(\mathbf{m}, \mathbf{r})$, the entity holding the trapdoor first samples a random element $x \leftarrow [N]$. This reduces the problem to a standard preimage sampling task over the lattice defined by $\mathbf{A}$. Using the trapdoor matrix $\mathbf{T}$ and a Gaussian sampler algorithm ($\mathsf{GSampler}$), the Issuer can efficiently sample a short preimage $\mathbf{s}$ that satisfies the required linear equation. This process successfully produces a valid pair $(x, \mathbf{s})$ without violating the established norm bounds.

Because the matrix $\mathbf{A}$ is no longer sampled uniformly at random but is instead generated according to an NTRU distribution, the underlying cryptographic assumption is structurally modified. This variant is formalized as the $\mathsf{NTRU\text{-}ISIS}_f$ assumption. As established by Stehlé and Steinfeld [SS11], we rely on the standard NTRU assumption, which states that a public matrix $\mathbf{A}$ generated via $\mathsf{NTRU.TrapGen}$ is computationally indistinguishable from a uniformly random matrix over the ring. Under this assumption, we can state that the $\mathsf{NTRU\text{-}ISIS}_f$ problem is computationally as hard as the foundational ISIS_f problem. Furthermore, as demonstrated in [BNLS23], the interactive version of this problem ($\mathsf{Int\text{-}NTRU\text{-}ISIS}_f$) can be tightly reduced to the static $\mathsf{NTRU\text{-}ISIS}_f$ assumption.

4.2 Issuing a Credential

This section details the protocol executed between the Issuer and the Holder to generate a credential via a Zero-Knowledge Proof (ZKP).

In this construction, the Holder possesses a set of plaintext attributes $a_1, \ldots, a_\ell$ that need to be certified. To integrate these attributes into the algebraic structure of the protocol, they are first hashed using a function H_M and then mapped to a polynomial vector $\mathbf{m} \in \mathcal{R}_q^{\ell_m}$. Specifically, $\mathbf{m}$ is defined as:

$$\mathbf{m} = \mathsf{Coeffs}^{-1}(H_M(a_1) \parallel \cdots \parallel H_M(a_\ell)) \tag{4.4}$$

Thus, the vector $\mathbf{m}$ strictly encapsulates all the hashed attributes. Alongside it, a corresponding vector $\mathbf{r} \in \mathcal{R}_q^{\ell_r}$ is generated to provide bounded commitment randomness.

The protocol requires a secure mechanism to commit the pair of vectors $(\mathbf{m}, \mathbf{r})$ to the Issuer, enabling the Issuer to compute a solution to the interactive $\mathsf{NTRU\text{-}ISIS}_{\mathsf{bin}}$ problem (as defined in Section 4.1.2). To allow the Issuer to generate a valid credential without learning the underlying attributes within $\mathbf{m}$ or the randomness $\mathbf{r}$, a Zero-Knowledge Proof based on the LNP framework [LNP22] is employed.

Initially, the Holder computes and transmits the value $t = \mathbf{c}_0^T\mathbf{m} + \mathbf{c}_1^T\mathbf{r}$, which represents the final term of Equation 4.3. The LNP protocol proves that the components of $\mathbf{m}$ and the randomness $\mathbf{r}$ correctly satisfy this commitment equation without disclosing their actual values. Consequently, the elements required for the generation of the credential are securely bound without exposing $\mathbf{m}$ and $\mathbf{r}$.

Furthermore, the intrinsic ℓ_2-norm bounds of the proof system are leveraged to mathematically guarantee the validity of the vectors. Because lattice-based zero-knowledge proofs require secret vectors to have inherently small coefficients to maintain both security and efficiency, arbitrary plaintext attributes cannot be embedded directly into the algebraic structure. Instead, the hash function H_M is specifically utilized to map these arbitrary attributes into a restricted coefficient space, ensuring that its output image systematically falls within a tightly bounded interval $[-\psi, \psi]$. Similarly, the commitment randomness $\mathbf{r}$ is uniformly sampled from this same interval. Consequently, the proof system can enforce a precise geometric bound dependent on the maximum coefficient size ψ and the structural dimensions of the vectors. The ℓ_2-norm of the concatenated secret vector is thus strictly bounded by:

$$\psi\sqrt{\ell_m d + \ell_r d} \tag{4.5}$$

where $\ell_m d$ and $\ell_r d$ represent the total number of integer coefficients in the attribute and randomness vectors, respectively.

4.2.1 Implementing Selective Disclosure

As discussed in Section 1.2, a fundamental requirement for a useful anonymous credential system is selective disclosure. To achieve this, the set of attributes a_i is partitioned into two distinct groups: public attributes, which are disclosed to verify eligibility for a service and are identified by the index set idx, and hidden attributes, denoted by the complementary index set $\overline{\text{idx}}$.

To support selective disclosure, the protocol mathematically reorganizes the vectors before generating the Zero-Knowledge Proof. The original attribute vector $\mathbf{m}$ is split into a public component and a hidden component. The public coefficients, $\mathsf{Coeffs}(\mathbf{m})_{\text{idx}}$, are transmitted in the clear to the Issuer (and subsequently to the Verifier). Conversely, the hidden attributes are concatenated with the randomness vector $\mathbf{r}$ to form the adjusted secret vector $\vec{s}$ required by the proof system:

$$\vec{s} = \begin{bmatrix} \mathsf{Coeffs}(\mathbf{m})_{\overline{\text{idx}}} \\ \mathsf{Coeffs}(\mathbf{r}) \end{bmatrix}$$

Because both the hashed attributes and the randomness are sampled from the bounded interval $[-\psi, \psi]$, the ℓ_2-norm of this newly formed secret vector $\vec{s}$ is strictly bounded by $\psi\sqrt{h \cdot |\overline{\text{idx}}| + \ell_r d}$, where h is the output length of the hash function per attribute.

During the issuance phase, the Holder still transmits the commitment $t = \mathbf{c}_0^T\mathbf{m} + \mathbf{c}_1^T\mathbf{r}$ in the clear, which is required by the Issuer to compute the signature. However, the Zero-Knowledge Proof is no longer evaluated directly on t. Instead, the protocol computes an adjusted public target $\vec{t}$, which isolates the hidden components by subtracting the known

contribution of the public attributes:

$$\vec{t} = \mathsf{Coeffs}(t) - \mathsf{rot}(\mathbf{c}_0^T)_{\mathrm{idx}}\mathsf{Coeffs}(\mathbf{m})_{\mathrm{idx}}$$

Consequently, the verification equation for the LNP protocol is reformulated over a projection matrix $P = [\mathsf{rot}(\mathbf{c}_0^T)_{\overline{\mathrm{idx}}} \parallel \mathsf{rot}(\mathbf{c}_1^T)]$. The proof now strictly asserts knowledge of a short secret $\vec{s}$ that satisfies the relation $P\vec{s} = \vec{t}$.

This algebraic reorganization fundamentally ensures privacy for the undisclosed attributes. The adjusted term $\vec{t}$ can be explicitly rewritten as the product of the projection matrix and the hidden vector components:

$$\vec{t} = \mathsf{rot}(\mathbf{c}_0^T)_{\overline{\mathrm{idx}}}\mathsf{Coeffs}(\mathbf{m})_{\overline{\mathrm{idx}}} + \mathsf{rot}(\mathbf{c}_1^T)\mathsf{Coeffs}(\mathbf{r}) = P\vec{s}$$

Because the short vector $\mathbf{r}$ acts as a cryptographic masking term, the hidden attributes remain securely protected. Under the hardness of the Module Learning with Errors (MLWE) assumption, the resulting term $\vec{t}$ leaks no computationally useful information regarding the undisclosed attributes.

4.2.2 Security in Concurrent Executions

Ensuring the knowledge soundness of the interactive protocol against concurrent attacks traditionally requires complex "rewinding" techniques. However, standard rewinding fails when an adversary initiates multiple parallel sessions, as rewinding one session inevitably disrupts the others. To solve this theoretical limitation in the standard model, the base protocol could apply the Katsumata transformation [KY17], a cryptographic technique that modifies the commitment structure to enable "straight-line" extraction (i.e., recovering the secret without rewinding).

While theoretically significant, this structural modification introduces a massive computational and bandwidth overhead. Consequently, for practical implementations such as the anonymous credential system analyzed in this thesis, the Katsumata transformation is omitted. Instead, the protocol efficiently mitigates concurrent attacks and achieves straight-line extraction by applying the Fiat-Shamir heuristic in the Random Oracle Model (as explored in Section 3.10), which compiles the interactive protocol into a secure Non-Interactive Zero-Knowledge (NIZK) architecture.

4.2.3 The Modulus Lifting Technique: Two Approaches

To securely and efficiently evaluate the Zero-Knowledge Proof, the original generalized relation, initially defined modulo q, must be scaled to a larger space modulo q_1 (where $q_1 \gg q$). This modulus lifting technique prevents modular wrap-around during the proof, while the relations we work on are evaluated on a smaller space. Architecturally, this lifting can be

performed in two distinct ways, presenting a trade-off between the dimension of the secret witness and the size of the challenge space.

Case 1: q_1 is a multiple of q In the first approach, the new modulus is chosen strictly as a multiple of the original one, such that $q_1 = c \cdot q$ for some integer c. This introduces a direct algebraic advantage: a natural homomorphism exists between the two quotient spaces. We can map the original equation to the new space simply by multiplying both sides by the scalar c.

Given the original matrix relation established for the credential verification:

$$P\vec{s} \equiv \vec{t} \pmod{q}$$

We scale it algebraically to obtain:

$$c \cdot P\vec{s} \equiv c \cdot \vec{t} \pmod{q_1}$$

Because of the multiplicative relation, verifying this equation modulo q_1 mathematically implies its correctness modulo q. However, because q_1 is a multiple of q (meaning that q and q_1/q are themselves divisors of q_1), its smallest divisor is inherently small. Since the security of the proof is bounded by this smallest divisor, the protocol is forced to sample a large number of challenge components, as explained in Section 3.4.1, (increasing the repetition parameter k) to achieve a sufficient level of security, which negatively impacts the proof size.

Case 2: q_1 is independent of q In the second approach, q_1 is chosen entirely independently of q (typically as a large prime). Without a multiplicative relation, the direct ring homomorphism is lost. To pass from one relation to the other, the matrix equation modulo q must be resolved into an exact relation over the integers ($\mathbb{Z}$) by introducing a vector of integer slack variables $\vec{w} \in \mathbb{Z}^d$:

$$P\vec{s} = \vec{t} + q\vec{w}$$

This exact equation can then be safely projected modulo the new prime q_1:

$$P\vec{s} - q\vec{w} \equiv \vec{t} \pmod{q_1}$$

More formally, this algebraic shift effectively expands the linear system into a block-matrix formulation:

$$[P \parallel -qI] \begin{bmatrix} \vec{s} \\ \vec{w} \end{bmatrix} \equiv \vec{t} \pmod{q_1}$$

In the context of the Zero-Knowledge Proof, the Prover must now mask and prove knowledge of $\vec{w}$ alongside the original components of $\vec{s}$, expanding the overall dimension of the secret witness. However, by choosing q_1 as a large prime number, its smallest and only

divisor is q_1 itself. Since the security of the challenge relies on the smallest divisor of the modulus, a prime q_1 intrinsically maximizes security, allowing the protocol to drastically diminish the required number of challenges k.

4.2.4 The Full Protocol

Having established the foundational techniques, this section outlines the complete issuance protocol. For our concrete instantiation, the scheme adopts the modulus lifting strategy outlined in the Case 1 approach in Section 4.2.3: the lifted modulus $\hat{q}_1$ is strictly configured as a multiple of q to avoid expanding the secret witness with additional slack variables.

Based on this architecture, the system requires a Non-Interactive Zero-Knowledge (NIZK) proof leveraging the LNP framework [LNP22]. This allows the Holder to securely transmit a masked commitment t of the pair $(\mathbf{m}, \mathbf{r})$ to the Issuer, as described in Section 4.2. The proving phase is detailed in Figure 4.1. Furthermore, a corresponding verification protocol ensures that the Issuer can cryptographically validate the correct construction of the commitment t without learning the underlying attributes, as illustrated in Figure 4.2.

4.3 Verification of the Credential

This section outlines the construction of the protocol between the Holder and the Verifier. As previously discussed, the Holder now possesses a credential $(\mathbf{s}, \vec{u}, \mathbf{m}', \mathbf{r}')$ and must demonstrate that it is a valid solution to the associated relation $\mathbf{As} = B\vec{u} + \mathbf{C}(\mathbf{m}', \mathbf{r}')$, where $\vec{u}$ is a binary scalar vector. To achieve this, a modified version of the LNP framework is employed, where the aggregated scalar witness w is constructed by concatenating the scalar coefficients of the credential components $w = (\mathsf{Coeffs}(\mathbf{s}), \mathsf{Coeffs}(\mathbf{r}'), \vec{u})$. This fully scalar vector is then mapped back into the polynomial secret $\mathbf{s}_1 = \mathsf{Coeffs}^{-1}(w)$ required by the lattice proof system.

The total bound on this aggregated secret is derived from the combination of its individual bounds: $\mathbf{s}$ is bounded by $\mathcal{B}_s$ (strictly linked to the ISIS_f problem); $\vec{u}$ is a binary vector, thus its ℓ_2-norm is tightly bounded by the square root of its dimension; and $\mathbf{r}'$ is bounded by $\psi\sqrt{h \cdot |\overline{\mathrm{idx}}| + \ell_r d}$, as detailed in Section 4.2.1.

Consequently, the proof must concurrently assert three conditions: the small norm of $\mathbf{s}_1$, the satisfaction of the underlying ISIS_f equation, and the binary nature of $\vec{u}$ (enforced by the polynomial constraint $u_i(u_i - 1) = 0$, as explained in Section 3.6.1).

Furthermore, consistent with the architecture of the issuance phase, this protocol also employs the modulus lifting strategy (Case 1, detailed in Section 4.2.3).

The resulting NIZK scheme [BNLS23] for this verification phase is detailed across three figures: Figure 4.4 illustrates the offline proving algorithm executed by the Holder, Figure 4.5 depicts the verification checks performed by the Verifier, and Figure 4.6 presents the complete interactive protocol.

$\mathsf{Prove}_{\mathsf{Com}}$: NI Credential Issuance Protocol (Prove)

Prover (Offline Execution)

Input: $\mathrm{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}), \quad x = (P, \vec{t}, \mathcal{B}, m'), \quad w = \vec{s}$

$rst := 0, \quad idx := 0$
$\mathbf{s}_1 := \mathbf{s}$
while $rst = 0 \wedge idx < N$ do:
  $idx := idx + 1$
  $\mathbf{s}_2 \leftarrow \chi^{m_2}$
  $\mathbf{t}_A := \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2$
  $\mathbf{y}_1 \leftarrow D^{m_1 d}_{\mathfrak{s}_1}, \quad \mathbf{y}_2 \leftarrow D^{m_2 d}_{\mathfrak{s}_2}, \quad \mathbf{y}_3 \leftarrow D^{256}_{\mathfrak{s}_3}$
  $\mathbf{g} \leftarrow \{x \in \mathcal{R}_{\hat{q}_1} : \tilde{x} = 0\}^k$
  $\mathbf{w} := \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$
  $\mathbf{t}_y := \mathbf{B}_y\mathbf{s}_2 + \mathbf{y}_3$
  $\mathbf{t}_g := \mathbf{B}_g\mathbf{s}_2 + \mathbf{g}$
  $a_1 := (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w})$
  $(\mathcal{R}_0, \mathcal{R}_1) := H(1, \mathrm{crs}, x, a_1) \in \{0,1\}^{256\times m_1 d} \times \{0,1\}^{256\times m_1 d}$
  $\mathcal{R} := \mathcal{R}_0 - \mathcal{R}_1$
  $\vec{s}_1 := \mathsf{Coeffs}(\mathbf{s}_1), \quad \vec{y}_3 := \mathsf{Coeffs}(\mathbf{y}_3)$
  $\vec{z}_3 := \vec{y}_3 + \mathcal{R}\vec{s}_1 \in \mathbb{Z}^{256}$
  $b_3 \leftarrow \mathsf{Rej}_1(\vec{z}_3, \mathcal{R}\vec{s}_1, \mathfrak{s}_3, M_3)$
  $a_2 := \vec{z}_3$
  $(\gamma_{i,j})_{i\in[k],j\in[256+n+1]} := H(2, \mathrm{crs}, x, a_1, a_2) \in \mathbb{Z}^{k\times(256+n+1)}_{\hat{q}_1}$
  for $i \in [k]$ do compute h_i
  $a_3 := \mathbf{h} = (h_1, \ldots, h_k)$
  $(\mu_i)_{i\in[k]} := H(3, \mathrm{crs}, x, a_1, a_2, a_3) \in \mathcal{R}^k_{\hat{q}_1}$

$$\mathbf{B} := \begin{bmatrix}\mathbf{B}_y \\ \mathbf{B}_g\end{bmatrix}, \quad \mathbf{t}_B := \begin{bmatrix}\mathbf{t}_y \\ \mathbf{t}_g\end{bmatrix}, \quad \mathbf{m} := \begin{bmatrix}\mathbf{y}_3 \\ \mathbf{g}\end{bmatrix} \in \mathcal{R}^{256/d+k}_{\hat{q}_1}, \quad \hat{\mathbf{s}} := \begin{bmatrix}\mathbf{s}_1 \\ \sigma_{-1}(\mathbf{s}_1) \\ \mathbf{m} \\ \sigma_{-1}(\mathbf{m})\end{bmatrix}$$

  Construct the quadratic relation in the form $\mathbf{s}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{s} + \mathbf{r}_0$ batching the constraints:
    1. The correct evaluation of the challenge bits z_j
    2. The linear combinations binding the secret to the public targets $\mathbf{t}$
    3. The exact ℓ_2-norm bound constraint on the secret $\sigma_{-1}(\mathbf{s}_1)^T\mathbf{s}_1 = \mathcal{B}$

  $f_1 := \mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}$
  $f_0 := \mathbf{y}^T\mathbf{R}_2\mathbf{y} + \mathbf{b}^T\mathbf{y}_2$
  $t := \mathbf{b}^T\mathbf{s}_2 + f_1$
  $a_4 := (t, f_0)$
  $c \leftarrow H(4, \mathrm{crs}, x, a_1, a_2, a_3, a_4) \in \mathcal{C}^*$
  for $i \in \{1, 2\}$ do:
    $\mathbf{z}_i := \mathbf{y}_i + c\mathbf{s}_i$
    $b_i \leftarrow \mathsf{Rej}_1(\mathbf{z}_i, c\mathbf{s}_i, \mathfrak{s}_i, M_i)$
  $rst := b_1 \wedge b_2 \wedge b_3$

if $rst = 1$ then return $\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2)$
else return $\perp$

Figure 4.1. The full Non-Interactive Zero-Knowledge Prove algorithm, modified for practical implementation.

$\mathsf{Verify}_{\mathsf{Com}}$: NI Credential Issuance Protocol (Verify)

Verifier (Issuer Execution)

Input: $\text{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}), \quad x = (P, \vec{t}, \mathcal{B}, m'), \quad \pi$
Output: bit $b \in \{0, 1\}$

parse $\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2)$
$a_1 := (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w})$
$a_2 := \vec{z}_3$
$a_3 := \mathbf{h}$
$a_4 := (t, f_0)$
$(\mathcal{R}_0, \mathcal{R}_1) := H(1, \text{crs}, x, a_1)$
$(\gamma_{i,j})_{i\in[k],j\in[256+n+1]} := H(2, \text{crs}, x, a_1, a_2)$
$(\mu_i)_{i\in[k]} := H(3, \text{crs}, x, a_1, a_2, a_3)$
$c := H(4, \text{crs}, x, a_1, a_2, a_3, a_4)$

$$\mathbf{B} := \begin{bmatrix}\mathbf{B}_y \\ \mathbf{B}_g\end{bmatrix}, \quad \mathbf{t}_B := \begin{bmatrix}\mathbf{t}_y \\ \mathbf{t}_g\end{bmatrix}$$

$$\mathbf{z} := \begin{bmatrix}\mathbf{z}_1 \\ \sigma_{-1}(\mathbf{z}_1) \\ c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2 \\ \sigma_{-1}(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2)\end{bmatrix}$$

Reconstruct the Credential Issuance Relation coefficients $\mathbf{D}_2, \mathbf{d}_1, d_0$

if one of the conditions below does not hold, then return 0:

1. $\|\mathbf{z}_1\| \le \mathfrak{s}_1\sqrt{2m_1\vec{d}}, \quad \|\mathbf{z}_2\| \le \mathfrak{s}_2\sqrt{2m_2\vec{d}}, \quad \|\vec{z}_3\| \le 1.7\sqrt{256}$
2. $\tilde{h}_i \stackrel{?}{=} 0$ for $i \in [k]$
3. $\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 \stackrel{?}{=} \mathbf{w} + c\mathbf{t}_A$
4. $\mathbf{z}^T\mathbf{D}_2\mathbf{z} + c\mathbf{d}_1^T\mathbf{z} + c^2d_0 - (ct - \mathbf{b}^T\mathbf{z}_2) \stackrel{?}{=} f_0$

else return 1

Figure 4.2. The verification algorithm of the NICIP protocol executed by the Issuer.

$\mathsf{AnonCreds.Issue}_{\mathcal{I},\mathcal{H}}\left(\langle \mathbf{isk}, \mathbf{attrs}', \mathbf{idx}\rangle, \langle \mathbf{ipk}, \mathbf{attrs}, \mathbf{idx}\rangle\right)$

$\underline{\mathcal{H}_1(\mathbf{ipk}, \mathbf{attrs}, \mathbf{idx})}$

parse attrs $= (a_1, \ldots, a_\ell)$
$\mathbf{m} = \mathsf{Coeffs}^{-1}(H_M(a_1) \parallel \cdots \parallel H_M(a_\ell)) \in \mathcal{R}_q^{\ell_m}$
$\mathbf{r} \leftarrow S_\psi^{\ell_r}$
$t = \mathbf{c}_0^T\mathbf{m} + \mathbf{c}_1^T\mathbf{r}$
$P = [\mathsf{rot}(\mathbf{c}_0^T)_{\overline{\text{idx}}} \parallel \mathsf{rot}(\mathbf{c}_1^T)] \in \mathbb{Z}_q^{d\times(|\overline{\text{idx}}|\cdot h+\ell_r d)}$
$\vec{s} = \begin{bmatrix}\mathsf{Coeffs}(\mathbf{m})_{\overline{\text{idx}}} \\ \mathsf{Coeffs}(\mathbf{r})\end{bmatrix} \in \mathbb{Z}_q^{|\overline{\text{idx}}|\cdot h+\ell_r d}$
$\vec{t} = \mathsf{Coeffs}(t) - \mathsf{rot}(\mathbf{c}_0^T)_{\text{idx}}\mathsf{Coeffs}(\mathbf{m})_{\text{idx}} \in \mathbb{Z}_q^d$
$\pi \leftarrow \mathsf{Prove}_{\mathsf{Com}}\left(\text{crs}_{\text{Com}}, \left(\hat{q}_1/q \cdot P, \hat{q}_1/q \cdot \vec{t}, \psi\sqrt{h \cdot |\overline{\text{idx}}| + \ell_r d}\right), \vec{s}\right)$
return $(\rho_1 = (t, \pi), \text{st}_{\mathcal{H}} = (\mathbf{m}, \mathbf{r}))$

$\underline{\mathcal{I}(\mathbf{isk}, \mathbf{attrs}', \mathbf{idx}, \rho_1)}$

parse $\rho_1 = (t, \pi)$, attrs$' = (a'_1, \ldots, a'_\ell)$
$\mathbf{m}' = \mathsf{Coeffs}^{-1}(H_M(a'_1) \parallel \cdots \parallel H_M(a'_\ell)) \in \mathcal{R}_q^{\ell_m}$
$P = [\mathsf{rot}(\mathbf{c}_0^T)_{\overline{\text{idx}}} \parallel \mathsf{rot}(\mathbf{c}_1^T)] \in \mathbb{Z}_q^{d\times(|\overline{\text{idx}}|\cdot h+\ell_r d)}$
$\vec{t} = \mathsf{Coeffs}(t) - \mathsf{rot}(\mathbf{c}_0^T)_{\text{idx}}\mathsf{Coeffs}(\mathbf{m}')_{\text{idx}} \in \mathbb{Z}_q^d$
if $\mathsf{Verify}_{\mathsf{Com}}\left(\text{crs}_{\text{Com}}, \left(\hat{q}_1/q \cdot P, \hat{q}_1/q \cdot \vec{t}, \psi\sqrt{h \cdot |\overline{\text{idx}}| + \ell_r d}\right), \pi\right) = 0$
  return $\perp$
$x \leftarrow [N]$
$\mathbf{s} \leftarrow \mathsf{GSampler}(\mathbf{a}_1, \mathbf{a}_2, \mathbf{T}, \bar{\mathbf{s}}, f(x) + t)$
return $\rho_2 = (\mathbf{s}, x)$

$\underline{\mathcal{H}_2(\mathbf{ipk}, \rho_2, \mathbf{st}_{\mathcal{H}})}$

parse $\rho_2 = (\mathbf{s}, x)$, $\text{st}_{\mathcal{H}} = (\mathbf{m}, \mathbf{r})$
if $\left(\|\mathbf{s}\| > s\sqrt{(m+2)d}\right) \vee (\|\mathbf{r}\| > \psi\sqrt{\ell_r d}) \vee ([\mathbf{a}_1 \parallel \mathbf{a}_2^T \parallel 1]\mathbf{s} \neq f(x) + \mathbf{c}_0^T\mathbf{m} + \mathbf{c}_1^T\mathbf{r})$
  return $\perp$
else return cred $= (\mathbf{s}, \mathbf{r}, x)$

Figure 4.3. The full interactive protocol for Anonymous Credential Issuance between the Issuer and the Holder.

$\mathsf{Prove}_{\mathsf{ISIS}}$: NI Credential Verification Protocol (Prove)

Prover (Offline Execution)

Input: $\text{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b})$, $\quad x = (P, C, \vec{m}, B, \text{Bounds}, \text{aux})$, $\quad w = (\vec{s}_{crs}, \vec{r}, \vec{u})$

$rst := 0, \quad idx := 0$
$\mathbf{s}_1 := \mathsf{Coeffs}^{-1}(w)$
while $rst = 0 \wedge idx < N$ do:
  $idx := idx + 1$
  $\mathbf{s}_2 \leftarrow \chi^{m_2}$
  $\mathbf{t}_A := \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2$
  $\mathbf{y}_1 \leftarrow D^{m_1\hat{d}}_{\mathfrak{s}_1}, \quad \mathbf{y}_2 \leftarrow D^{m_2\hat{d}}_{\mathfrak{s}_2}, \quad \mathbf{y}_3 \leftarrow D^{256}_{\mathfrak{s}_3}$
  $\mathbf{g} \leftarrow \{x \in \mathcal{R}_{\hat{q}_2} : \tilde{x} = 0\}^k$
  $\mathbf{w} := \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$
  $\mathbf{t}_y := \mathbf{B}_y\mathbf{s}_2 + \mathbf{y}_3$
  $\mathbf{t}_g := \mathbf{B}_g\mathbf{s}_2 + \mathbf{g}$
  $a_1 := (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w})$
  $(\mathcal{R}_0, \mathcal{R}_1) := H(1, \text{crs}, x, a_1) \in \{0,1\}^{256\times m_1\hat{d}} \times \{0,1\}^{256\times m_1\hat{d}}$
  $\mathcal{R} := \mathcal{R}_0 - \mathcal{R}_1$
  $\vec{s}_1 := \mathsf{Coeffs}(\mathbf{s}_1), \quad \vec{y}_3 := \mathsf{Coeffs}(\mathbf{y}_3)$
  $\vec{z}_3 := \vec{y}_3 + \mathcal{R}\vec{s}_1 \in \mathbb{Z}^{256}$
  $b_3 \leftarrow \mathsf{Rej}_1(\vec{z}_3, \mathcal{R}\vec{s}_1, \mathfrak{s}_3, M_3)$
  $a_2 := \vec{z}_3$
  $(\gamma_{i,j})_{i\in[k],j\in[256+n+3]} := H(2, \text{crs}, x, a_1, a_2) \in \mathbb{Z}^{k\times(256+n+3)}_{\hat{q}_2}$
  for $i \in [k]$ do compute h_i
  $a_3 := \mathbf{h} = (h_1, \ldots, h_k)$
  $(\mu_i)_{i\in[k]} := H(3, \text{crs}, x, a_1, a_2, a_3) \in \mathcal{R}^k_{\hat{q}_2}$

$$\mathbf{B} := \begin{bmatrix}\mathbf{B}_y \\ \mathbf{B}_g\end{bmatrix}, \quad \mathbf{t}_B := \begin{bmatrix}\mathbf{t}_y \\ \mathbf{t}_g\end{bmatrix}, \quad \mathbf{m} := \begin{bmatrix}\mathbf{y}_3 \\ \mathbf{g}\end{bmatrix} \in \mathcal{R}^{256/\hat{d}+k}_{\hat{q}_2}, \quad \hat{\mathbf{s}} := \begin{bmatrix}\mathbf{s}_1 \\ \sigma_{-1}(\mathbf{s}_1) \\ \mathbf{m} \\ \sigma_{-1}(\mathbf{m})\end{bmatrix}$$

  Construct the quadratic relation, in the form $\mathbf{s}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{s} + \mathbf{r}_0$ batching the constraints:
    1. The correct evaluation of the challenge bits z_j
    2. The satisfaction of the underlying ISIS relation
    3. The exact ℓ_2-norm bound constraints on the secret vectors $\mathbf{s}$ and $\mathbf{r}'$
    4. The enforcement of the binary constraint on the vector $\vec{u}$

  $f_1 := \mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}$
  $f_0 := \mathbf{y}^T\mathbf{R}_2\mathbf{y} + \mathbf{b}^T\mathbf{y}_2$
  $t := \mathbf{b}^T\mathbf{s}_2 + f_1$
  $a_4 := (t, f_0)$
  $c \leftarrow H(4, \text{crs}, x, a_1, a_2, a_3, a_4) \in \mathcal{C}^*$
  for $i \in \{1, 2\}$ do:
    $\mathbf{z}_i := \mathbf{y}_i + c\mathbf{s}_i$
    $b_i \leftarrow \mathsf{Rej}_1(\mathbf{z}_i, c\mathbf{s}_i, \mathfrak{s}_i, M_i)$
  $rst := b_1 \wedge b_2 \wedge b_3$

if $rst = 1$ then return $\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2)$
else return $\perp$

Figure 4.4. The $\mathsf{Prove}_{\mathsf{ISIS}}$ non-interactive zero-knowledge algorithm for the verification phase.

$\mathsf{Verify}_{\mathsf{ISIS}}$: NI Credential Verification Protocol (Verify)

Verifier Execution

Input: $\text{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}), \quad x = (P, C, \vec{m}, B, \text{Bounds}, \text{aux}), \quad \pi$
Output: bit $b \in \{0, 1\}$

parse $\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2)$
$a_1 := (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w})$
$a_2 := \vec{z}_3$
$a_3 := \mathbf{h}$
$a_4 := (t, f_0)$
$(\mathcal{R}_0, \mathcal{R}_1) := H(1, \text{crs}, x, a_1)$
$(\gamma_{i,j})_{i\in[k],j\in[256+n+3]} := H(2, \text{crs}, x, a_1, a_2)$
$(\mu_i)_{i\in[k]} := H(3, \text{crs}, x, a_1, a_2, a_3)$
$c := H(4, \text{crs}, x, a_1, a_2, a_3, a_4)$

$$\mathbf{B} := \begin{bmatrix} \mathbf{B}_y \\ \mathbf{B}_g \end{bmatrix}, \quad \mathbf{t}_B := \begin{bmatrix} \mathbf{t}_y \\ \mathbf{t}_g \end{bmatrix}$$

$$\mathbf{z} := \begin{bmatrix} \mathbf{z}_1 \\ \sigma_{-1}(\mathbf{z}_1) \\ c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2 \\ \sigma_{-1}(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2) \end{bmatrix}$$

Reconstruct the quadratic relation coefficients $\mathbf{D}_2, \mathbf{d}_1, d_0$

if one of the conditions below does not hold, then return 0:

1. $\|\mathbf{z}_1\| \le \mathcal{B}_1 := \mathfrak{s}_1\sqrt{2m_1\hat{d}}, \quad \|\mathbf{z}_2\| \le \mathcal{B}_2 := \mathfrak{s}_2\sqrt{2m_2\hat{d}}, \quad \|\vec{z}_3\| \le \mathcal{B}_3 := 1.7\sqrt{256}$
2. $\tilde{h}_i \stackrel{?}{=} 0$ for $i \in [k]$
3. $\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 \stackrel{?}{=} \mathbf{w} + c\mathbf{t}_A$
4. $\mathbf{z}^T\mathbf{D}_2\mathbf{z} + c\mathbf{d}_1^T\mathbf{z} + c^2d_0 - (ct - \mathbf{b}^T\mathbf{z}_2) \stackrel{?}{=} f_0$

else return 1

Figure 4.5. The $\mathsf{Verify}_{\mathsf{ISIS}}$ non-interactive zero-knowledge algorithm for the verification phase.

$$\mathsf{AnonCreds.Verify}_{\mathcal{H},\mathcal{V}}\left(\langle \mathbf{cred}, \mathbf{attrs}, \mathbf{idx}\rangle, \langle \mathbf{ipk}, \mathbf{attrs}', \mathbf{idx}\rangle\right)$$

$\underline{\mathcal{H}(\mathbf{ipk}, \mathbf{attrs}, \mathbf{cred}, \mathbf{idx})}$

parse attrs $= (a_1, \ldots, a_\ell)$, cred $= (\mathbf{s}, \mathbf{r}, x)$
$\mathbf{m} = \mathsf{Coeffs}^{-1}(H_M(a_1) \parallel \cdots \parallel H_M(a_\ell)) \in \mathcal{R}_q^{\ell_m}$
$P = \mathsf{rot}([\mathbf{a}_1 \parallel \mathbf{a}_2^T \parallel 1]) \in \mathbb{Z}_q^{d\times(m+2)d}$
$C = [\mathsf{rot}(\mathbf{c}_0^T)_{\overline{\text{idx}}} \parallel \mathsf{rot}(\mathbf{c}_0^T)_{\text{idx}} \parallel \mathsf{rot}(\mathbf{c}_1^T)] \in \mathbb{Z}_q^{d\times(\ell_m+\ell_r)d}$
$\vec{m} = \mathsf{Coeffs}(\mathbf{m})_{\text{idx}} \in \mathbb{Z}_q^{|\text{idx}|\cdot h}$
$\vec{s}_{crs} = \mathsf{Coeffs}(\mathbf{s}) \in \mathbb{Z}_q^{(m+2)d}$
$\vec{r} = \begin{bmatrix}\mathsf{Coeffs}(\mathbf{m})_{\overline{\text{idx}}} \\ \mathsf{Coeffs}(\mathbf{r})\end{bmatrix} \in \mathbb{Z}_q^{|\overline{\text{idx}}|\cdot h+\ell_r d}$
$\vec{u} = \vec{\text{enc}}(x) \in \{0,1\}^t$
$\text{Bounds} = \left(s\sqrt{(m+2)d}, \psi\sqrt{h\cdot|\overline{\text{idx}}| + \ell_r d}\right)$
$\pi \leftarrow \mathsf{Prove}_{\mathsf{ISIS}}\left(\text{crs}_{\text{ISIS}}, (\hat{q}_2/q\cdot P, \hat{q}_2/q\cdot C, \vec{m}, \hat{q}_2/q\cdot B, \text{Bounds}, \text{idx}), (\vec{s}_{crs}, \vec{r}, \vec{u})\right)$
return π

$\underline{\mathcal{V}(\mathbf{ipk}, \mathbf{attrs}', \mathbf{idx}, \pi)}$

parse attrs$' = (a_1', \ldots, a_\ell')$
$\mathbf{m}' = \mathsf{Coeffs}^{-1}(H_M(a_1') \parallel \cdots \parallel H_M(a_\ell')) \in \mathcal{R}_q^{\ell_m}$
$P = \mathsf{rot}([\mathbf{a}_1 \parallel \mathbf{a}_2^T \parallel 1]) \in \mathbb{Z}_q^{d\times(m+2)d}$
$C = [\mathsf{rot}(\mathbf{c}_0^T)_{\overline{\text{idx}}} \parallel \mathsf{rot}(\mathbf{c}_0^T)_{\text{idx}} \parallel \mathsf{rot}(\mathbf{c}_1^T)] \in \mathbb{Z}_q^{d\times(\ell_m+\ell_r)d}$
$\vec{m}' = \mathsf{Coeffs}(\mathbf{m}')_{\text{idx}} \in \mathbb{Z}_q^{|\text{idx}|\cdot h}$
$\text{Bounds} = \left(s\sqrt{(m+2)d}, \psi\sqrt{h\cdot|\overline{\text{idx}}| + \ell_r d}\right)$
return $\mathsf{Verify}_{\mathsf{ISIS}}\left(\text{crs}_{\text{ISIS}}, (\hat{q}_2/q\cdot P, \hat{q}_2/q\cdot C, \vec{m}', \hat{q}_2/q\cdot B, \text{Bounds}, \text{idx}), \pi\right)$

Figure 4.6. The full interactive protocol for Anonymous Credential Verification between the Holder and the Verifier.

4.4 Security Parameters and Instantiation

To ensure a secure and efficient instantiation of the protocol, we rely on the concrete parameter sets proposed in [BNLS23].

Table 4.1 details the foundational parameters of the lattice-based credential scheme, including dimensions, moduli, and statistical bounds required for the correct functioning of the underlying SIS and LWE problems.

Parameter	Description	Value
d	Ring dimension of the base polynomial ring $\mathcal{R}$	4096
q	Base modulus of the credential system	$17\,179\,861\,781 \approx 2^{34}$
N	Maximum number of iterations for the Prover	2^{640}
ω_c	Fixed weight (number of non-zero coefficients) of the challenge polynomial	640
h	Fixed weight (number of non-zero coefficients) of the hashed attributes	512
ℓ	Total number of attributes encoded in the credential	8
ℓ_r	Length of the randomness polynomial vector $\mathbf{r}$	2
ℓ_m	Length of the message polynomial vector $\mathbf{m}$	1
m	Dimension parameter for the secret vector $\mathbf{s}$	3
ψ	Gaussian standard deviation parameter for the randomness	2
$\mathfrak{s}$	Gaussian standard deviation parameter for the secret elements	$\approx 2^{27}$
δ	Auxiliary structural parameter for the lattice bound	284
κ	Statistical zero-knowledge and soundness parameter	56
α	Scaling factor for the standard deviation	12
M	Expected number of repetitions for rejection sampling	3
$T_{\max}$	Maximum runtime threshold	678

Table 4.1. Base parameters for the core credential scheme.

The verification phase, executed by the Verifier to check the ISIS relations, utilizes the parameters defined in Table 4.2. The modulus $\hat{q}_2$ is specifically scaled to avoid modular wrap-around during the verification checks.

Similarly, Table 4.3 presents the corresponding parameters required for the credential issuance phase ($\Pi^{\text{Com}}_{\text{NIZK}}$). The scaled modulus $\hat{q}_1$ is utilized in this context.

Parameter	Description	Value
$\hat{d}$	Ring dimension of the proof ring $\mathcal{R}_{\hat{q}_2}$	128
$\hat{p}_1$	The smallest prime divisor of $\hat{q}_2$	17 179 861 781
$\hat{q}_2$	Proof system modulus divisible by q	$\approx 2^{88}$
ξ	Infinity norm on the challenges in $\mathcal{C}^*$	17
ν	Parameter introduced for masking standard deviation	350
k	Number of repetitions for soundness	10
n	Length of the commitment vector $\mathbf{t}_A$	30
m_2	Length of the randomness vector $\mathbf{s}_2$	68

Table 4.2. Parameters for the verification proof system $\Pi^{\text{ISIS}}_{\text{NIZK}}$.

Parameter	Description	Value
$\hat{d}$	Ring dimension of the proof ring $\mathcal{R}_{\hat{q}_1}$	128
$\hat{p}_1$	The smallest prime divisor of $\hat{q}_1$	16253
$\hat{p}_2$	Second prime divisor of $\hat{q}_1$	17 179 861 781
$\hat{q}_1$	Proof system modulus divisible by q	$\approx 2^{48}$
n	Length of the commitment vector $\mathbf{t}_A$	18
m_2	Length of the randomness vector $\mathbf{s}_2$	51

Table 4.3. Parameters for the credential issuance proof system $\Pi^{\text{Com}}_{\text{NIZK}}$.

4.5 Differences Between the LNP Framework and the BLNS Instantiation

By analyzing the protocols utilized during the issuance and verification phases, it becomes evident that the number of equations to be verified differs from those in the original LNP framework. Notably, BLNS implements an alternative, highly efficient method to transform the norm inequality into an exact equality constraint.

While LNP introduces a binary-represented slack variable to reach the theoretical bound (as detailed in Section 3.6.1), BLNS expresses the difference between the squared bound and the actual squared norm as the sum of four integer squares [RS86]. The four integer bases of these squares are then concatenated with the original secret vector $\mathbf{s}_1$ and treated as standard elements strictly for the overall norm evaluation. To ensure these additional components do not interfere with the underlying linear relations, the public matrices are correspondingly padded with blocks of zeros, analogous to the approach described in Section 3.6.1.

This exact representation is achieved efficiently by employing the randomized algorithm proposed by Rabin and Shallit, which computes the four squares with an expected time complexity of $\mathcal{O}(\log^2(\Delta))$, where Δ represents the integer difference being decomposed.

Another significant distinction lies in the choice of the rejection sampling technique. The BLNS instantiation consistently relies on the Rej_1 algorithm. Consequently, the sign bit b is entirely omitted from the masking process, removing the necessity to cryptographically verify its domain, unlike the original LNP approach described in Section 3.3.2.

These architectural divergences will be further explored and theoretically compared in Section 5.

4.6 Optimization of the Protocol by LINKS Foundation

The LINKS Foundation has optimized both the $\mathsf{Prove}_{\mathsf{Com}}$ (Figure 4.1) and $\mathsf{Prove}_{\mathsf{ISIS}}$ (Figure 4.4) algorithms.

The primary computational bottleneck of the protocol resides in the rejection sampling phase. In the standard implementation, a single failure in any of the rejection conditions triggers a restart of the entire `while` loop, forcing a complete recomputation of all previous steps. To address this inefficiency, the team at the LINKS Foundation restructured the algorithm by dividing the main execution into multiple distinct cycles. Consequently, a rejection now only requires the repetition of a specific, smaller subset of operations rather than the entire protocol.

Particular care was taken during this optimization to preserve the underlying soundness of the protocol. It is crucial to ensure that, within each independent cycle, every commitment remains cryptographically bound to the equations involving the extracted challenges.

This structural integrity prevents a malicious Prover from forging ad-hoc elements to bypass the verification. Furthermore, this modular approach allows the rejection sampling steps concerning the vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ to be evaluated independently from the sampling of $\vec{z}_3$.

The optimized version of the $\mathsf{Prove}_{\mathsf{ISIS}}$ algorithm is illustrated in Figure 4.7. For the subsequent parameter analysis and comparison, we will refer to this optimized protocol to evaluate the total number of operations.

Optimized NICIP: $\mathsf{Prove}_{\mathsf{ISIS}}$ Algorithm

Prover (Offline Execution)

Input: $\text{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b})$, $x = (P, C, \vec{m}_{\text{idx}}, B, \text{Bounds}, \text{aux})$, $w = (\vec{s}_{crs}, \vec{r}, \vec{u})$

$rst_1 := 0$, $rst_2 := 0$, $idx_1 := 0$, $idx_2 := 0$
$\mathbf{s}_1 := \mathsf{Coeffs}^{-1}(w)$

$\mathbf{s}_2 \leftarrow \chi^{m_2}$
$\mathbf{t}_A := \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2$
while $rst_1 = 0 \wedge idx_1 < N$ do:
  $idx_1 := idx_1 + 1$
  $\mathbf{y}_3 \leftarrow D^{256}_{\mathfrak{s}_3}$
  $\mathbf{t}_y := \mathbf{B}_y\mathbf{s}_2 + \mathbf{y}_3$
  $a_1 := (\mathbf{t}_A, \mathbf{t}_y)$
  $(\mathcal{R}_0, \mathcal{R}_1) := H(1, \text{crs}, x, a_1) \in \{0,1\}^{256\times m_1 d} \times \{0,1\}^{256\times m_1 d}$
  $\mathcal{R} := \mathcal{R}_0 - \mathcal{R}_1$
  $\vec{s}_1 := \mathsf{Coeffs}(\mathbf{s}_1)$, $\vec{y}_3 := \mathsf{Coeffs}(\mathbf{y}_3)$
  $\vec{z}_3 := \vec{y}_3 + \mathcal{R}\vec{s}_1 \in \mathbb{Z}^{256}$
  $b_3 \leftarrow \mathsf{Rej}_1(\vec{z}_3, \mathcal{R}\vec{s}_1, \mathfrak{s}_3, M_3)$
  $rst_1 := b_3$

$\mathbf{g} \leftarrow \{x \in \mathcal{R}_{\hat{q}_2} : \tilde{x} = 0\}^k$
$\mathbf{t}_g := \mathbf{B}_g\mathbf{s}_2 + \mathbf{g}$
$a_2 := (\vec{z}_3, \mathbf{t}_g)$
$(\gamma_{i,j})_{i\in[k], j\in[256+n+3]} := H(2, \text{crs}, x, a_1, a_2) \in \mathbb{Z}^{k\times(256+n+3)}_{\hat{q}_2}$
for $i \in [k]$ do compute h_i
  $a_3 := \mathbf{h} = (h_1, \ldots, h_k)$
  $(\mu_i)_{i\in[k]} := H(3, \text{crs}, x, a_1, a_2, a_3) \in \mathcal{R}^k_{\hat{q}_2}$
  $$\mathbf{B} := \begin{bmatrix}\mathbf{B}_y \\ \mathbf{B}_g\end{bmatrix}, \mathbf{t}_B := \begin{bmatrix}\mathbf{t}_y \\ \mathbf{t}_g\end{bmatrix}, \mathbf{m} := \begin{bmatrix}\mathbf{y}_3 \\ \mathbf{g}\end{bmatrix} \in \mathcal{R}^{256/d+k}_{\hat{q}_2}, \hat{\mathbf{s}} := \begin{bmatrix}\mathbf{s}_1 \\ \sigma_{-1}(\mathbf{s}_1) \\ \mathbf{m} \\ \sigma_{-1}(\mathbf{m})\end{bmatrix}$$

Construct the quadratic relation in the form $\mathbf{s}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{s} + \mathbf{r}_0$ batching the following target constraints:
1. The correct evaluation of the challenge bits z_j
2. The satisfaction of the underlying ISIS relation
3. The exact ℓ_2-norm bound constraints on the secret vectors $\mathbf{s}$ and $\mathbf{r}'$
4. The enforcement of the binary constraint on the scalar vector $\vec{u}$

while $rst_2 = 0 \wedge idx_2 < N$ do:
  $idx_2 := idx_2 + 1$
  $\mathbf{y}_1 \leftarrow D^{m_1 d}_{\mathfrak{s}_1}$, $\mathbf{y}_2 \leftarrow D^{m_2 d}_{\mathfrak{s}_2}$
  $\mathbf{w} := \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$
  $f_1 := \mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}$
  $f_0 := \mathbf{y}^T\mathbf{R}_2\mathbf{y} + \mathbf{b}^T\mathbf{y}_2$
  $t := \mathbf{b}^T\mathbf{s}_2 + f_1$
  $a_4 := (\mathbf{w}, t, f_0)$
  $c \leftarrow H(4, \text{crs}, x, a_1, a_2, a_3, a_4) \in \mathcal{C}^*$
  for $i \in \{1, 2\}$ do:
    $\mathbf{z}_i := \mathbf{y}_i + c\mathbf{s}_i$
    $b_i \leftarrow \mathsf{Rej}_1(\mathbf{z}_i, c\mathbf{s}_i, \mathfrak{s}_i, M_i)$
    $rst_2 := b_1 \wedge b_2$

if $rst_1 = 1 \wedge rst_2 = 1$ then return $\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2)$
else return $\perp$

Figure 4.7. The optimized $\mathsf{Prove}_{\mathsf{ISIS}}$ zero-knowledge algorithm featuring nested optimizations and segregated rejection sampling loops.

Chapter 5

Parameter and Algorithm Comparison

This section analyzes how the differences between the LNP and BLNS protocols, as described in Section 4.5, influence the overall performance of the system. To establish a clear baseline for this comparison, we use as a reference the LNP protocols depicted in Figure 3.2 and Figure 3.3, alongside the BLNS issuance and verification protocols illustrated in Figure 4.3 and Figure 4.6.

To systematically evaluate these architectural choices and their impact on computational efficiency, the remainder of this section is structured as follows:

- **Inequality to Equality Transformation:** Contrasts the LNP and BLNS methods for bounding the slack difference, introducing and benchmarking a novel Greedy decomposition algorithm to reduce the computational overhead.
- **Rejection Algorithm Trade-offs:** Provides a mathematical analysis of the Rej_0, Rej_1, and Rej_2 algorithms, identifying the optimal crossover points for their deployment based on variance constraints and expected repetition rates.
- **LINKS Parameter Validation:** Formally validates the theoretical framework against the concrete parameter instantiation implemented in the LINKS Foundation codebase, extracting the exact masking bounds $(\mathbf{z}_1, \mathbf{z}_2, \vec{z}_3)$ and evaluating the medium number of required sampling operations.

5.1 Managing the Transformation from Inequality to Equality

This section analyzes how each protocol manages the transformation of the norm inequality into an exact equality with respect to the theoretical upper bound $\mathcal{B}$ of the permitted slack.

As explained in Section 3.6.1, the LNP protocol expresses the slack difference δ through its binary representation $\vec{x}$. A new polynomial, whose degree d must satisfy $d \geq \lceil \log_2(\mathcal{B}) \rceil$, is assembled using these bits as coefficients and concatenated to the secret vector $\mathbf{s}_1$. Subsequently, the protocol must cryptographically enforce that $\vec{x}$ is strictly composed of binary elements and incorporate it into the overall exact norm evaluation.

Conversely, the BLNS scheme identifies four integers whose squares sum perfectly to the difference δ, relying on Lagrange's four-square theorem. These four integer elements are simply used as coefficients to form a new polynomial (with the remaining $d-4$ coefficients padded with zeros), which is then concatenated with the secret $\mathbf{s}_1$ and utilized directly to evaluate the ℓ_2-norm.

In both cases, the procedure fundamentally resolves to the structural concatenation of exactly one new polynomial to the vector $\mathbf{s}_1$. Because this addition introduces an identical structural overhead in both frameworks, the spatial consequences of the concatenation will not be considered in the comparative analysis; the focus remains strictly on the computational operations.

LNP Strategy

Excluding the shared structural overhead, the specific computational cost of the LNP strategy consists of the following operations:

- Expressing the difference δ in its binary form $\vec{x}$, which requires $\mathcal{O}(\log_2(\mathcal{B}))$ bitwise operations.

- Proving that the components of $\vec{x}$ are strictly binary. This step relies on masked polynomial multiplications to evaluate the required quadratic relations (e.g., $\vec{x} \circ (\vec{x} - \mathbf{1}) = \mathbf{0}$). Since modern lattice cryptography natively relies on the Number Theoretic Transform (NTT), the computational complexity for these evaluations over a polynomial of degree $d \approx \log_2(\mathcal{B})$ is $\mathcal{O}(\log_2(\mathcal{B}) \log(\log_2(\mathcal{B})))$. Crucially, this binary constraint imposes a symmetric burden on the Verifier, who must recompute these masked equations to verify their validity.

Consequently, the total asymptotic computational cost for the inequality-to-equality transformation in the LNP strategy is distributed between the Prover and Verifier:

$$C_{\text{LNP-Prover}} = \mathcal{O}(\log_2(\mathcal{B}))_{\text{bin}} + \mathcal{O}(\log_2(\mathcal{B}) \log(\log_2(\mathcal{B})))_{\text{NTT}} \tag{5.1}$$

$$C_{\text{LNP-Verifier}} = \mathcal{O}(\log_2(\mathcal{B}) \log(\log_2(\mathcal{B})))_{\text{verification}} \tag{5.2}$$

BLNS Strategy

The specific computational cost of the BLNS strategy is defined by:

- Identifying the four integers whose squares sum exactly to δ. This decomposition is efficiently achieved via the Rabin-Shallit randomized algorithm. The performance depends on the magnitude of δ, resulting in an expected offline computational complexity of $\mathcal{O}((\log_2(\mathcal{B}))^2)$ probabilistic arithmetic operations.
- Unlike the LNP strategy, the BLNS formulation intrinsically accepts any integer coefficients. Consequently, no supplementary zero-knowledge equations are required to prove the format of these appended scalars, entirely relieving the Verifier of any associated algebraic burden.

Thus, the total asymptotic cost in the BLNS strategy is strictly confined to the Prover's offline execution:

$$C_{\text{BLNS-Prover}} = \mathcal{O}((\log_2(\mathcal{B}))^2)_{\text{Rabin-Shallit}} \tag{5.3}$$

$$C_{\text{BLNS-Verifier}} = 0 \tag{5.4}$$

Trade-off Analysis

These two methods present distinct advantages and inherent trade-offs. The LNP approach relies on a lightweight binary conversion for the Prover; however, the resulting zero-knowledge constraints distribute a symmetric computational burden onto the Verifier. This shared overhead is generally undesirable in real-world cryptographic deployments, as it directly introduces latency into the online verification process.

Conversely, the BLNS scheme shifts the entirety of the computational cost to the Prover's offline execution of the Rabin-Shallit algorithm. While this algebraic decomposition increases the initial operational complexity, it achieves the highly desirable property of zero-cost online verification for this algorithmic step. Furthermore, eliminating the binary slack structure fundamentally tightens the protocol's geometric security constraints. By expressing the slack directly via the sum of squares, the BLNS framework reduces the worst-case algebraic ceiling in the Norm Evaluation Bound (Section 3.9) from $2\beta^2 - 1$ down to exactly β^2, transforming the condition to:

$$\beta^2 + B^2 < q \tag{5.5}$$

This tighter bound directly reduces the geometric pressure on the requisite bit-size of the zero-knowledge modulus q.

5.1.1 A Novel Greedy Approach for Inequality Management

To optimize the norm inequality transformation, an alternative approach is to express the slack δ as an arbitrary sum of quadratic terms, thereby bypassing the offline computational overhead of the Rabin-Shallit algorithm. Since a pre-allocated polynomial of degree d is concatenated to the secret regardless of the decomposition method, this spatial buffer can be leveraged to divide the slack into a flexible number of integer squares k, provided $k \leq d$.

These k integer roots simply become the first coefficients of the slack polynomial, padded with zeros.

A naive strategy would represent δ purely as a sum of units ($\delta \times 1^2$), yielding a computational and memory overhead of $\mathcal{O}(\delta)$. This linear scaling is practically unfeasible.

The objective is to represent δ using at most d squares. A highly efficient strategy is to iteratively extract the largest perfect square, leaving a strictly bounded remainder. This translates to a deterministic greedy algorithm:

- Initialize the target slack: $x_0 := \delta$.
- While $x_i > 0$:
  - Find the base of the largest perfect square: $a_i := \lfloor \sqrt{x_i} \rfloor$.
  - Update the remainder for the next iteration: $x_{i+1} := x_i - a_i^2$.

Iteration Complexity Analysis

Because x_i is strictly bounded by the next perfect square $(a_i + 1)^2 = a_i^2 + 2a_i + 1$, the generated remainder is mathematically constrained to $x_{i+1} \leq 2a_i$. Given that $a_i \leq \sqrt{x_i}$, the remainder is asymptotically bounded by $2\sqrt{x_i}$. If the element x_i has a bit-length L_i, the bit-length of the subsequent remainder essentially halves ($L_{i+1} \approx L_i/2$).

Evaluating the worst-case scenario where the remainder halts at $x_k = 1$, the total number of iterations is upper-bounded by the exact number of times the initial bit-length $L_0 = \log_2(\mathcal{B})$ can be halved. Consequently, the iteration complexity is strictly bounded by $\mathcal{O}(\log_2(\log_2(\mathcal{B})))$.

Worst-Case Numerical Boundaries

To analytically determine the exact numerical boundary up to which any integer is guaranteed to decompose within k iterations, we define a sequence X_k representing the smallest integer that forces the greedy algorithm to use exactly k squares. By definition, the first step extracts the largest square a^2. For the algorithm to have validly chosen a^2 rather than $(a+1)^2$, the remainder X_{k-1} must satisfy $X_{k-1} \leq 2a$. This establishes that the minimum integer base is $a = \lceil X_{k-1}/2 \rceil$. Substituting this minimal base back into the decomposition yields the recurrence relation:

$$X_k = \left\lceil \frac{X_{k-1}}{2} \right\rceil^2 + X_{k-1} \tag{5.6}$$

Starting from $X_1 = 1$, the sequence exhibits strict double-exponential growth, as illustrated in Table 5.1.

While this sequence is established in number theory [Lem1896], its application to replace the randomized Rabin-Shallit algorithm in lattice-based zero-knowledge proofs represents a novel architectural optimization. Even for exceptionally large slack values, the required

Number of Squares (k)	**Smallest Integer Requiring k Squares (X_k)**
1	1
2	2
3	3
4	7
5	23
6	167
7	7223
8	13053767
9	42600227751223
10	453694851167777998510220167

Table 5.1. Worst-case sequence for the Greedy Sum of Squares decomposition up to $k = 10$.

squares scale proportionally to $\mathcal{O}(\log \log \mathcal{B})$, remaining comfortably below the standard polynomial degree limit d.

Practical Computational Efficiency

The actual computational time heavily depends on the underlying hardware. If $\mathcal{B}$ fits within standard hardware registers (64-bit), the integer square root ($\lfloor \sqrt{x_i} \rfloor$) is natively supported as a single CPU instruction, scaling exactly to $\mathcal{O}(\log \log \mathcal{B})$ operations.

If the security parameters dictate bounds exceeding standard registers, arbitrary-precision libraries (e.g., GMP) must be used. Using Newton-Raphson methods, software-based integer square roots run in time proportional to large-integer multiplication, yielding an asymptotic bitwise complexity of $\mathcal{O}(\log \mathcal{B} \log \log \mathcal{B})$, a cost dominated almost entirely by the first iteration.

Compared to Rabin-Shallit, the greedy approach is strictly deterministic, requires zero heavy modular arithmetic, and relies exclusively on fast integer square roots applied to rapidly shrinking remainders. This structural simplicity drastically reduces hidden operational constants and significantly eases software implementation, establishing it as a highly efficient alternative for practical cryptographic deployments.

Experimental Benchmark

To validate the theoretical efficiency of the proposed Greedy decomposition against the standard Rabin-Shallit approach, a software simulation was conducted over 50,000 uniformly random targets sampled in the range $[1000, 2^{32}]$.

The Rabin-Shallit execution inherently requires heavy probabilistic prime searches (via Miller-Rabin testing) and modular exponentiations. Conversely, the Greedy algorithm exclusively leverages deterministic, arbitrary-precision integer square roots. To ensure practical execution times during the simulation, a strict bound of 5,000 iterations was imposed on the randomized Rabin-Shallit algorithm, triggering a timeout abort if a valid decomposition was not found.

The empirical results confirm the superior performance of the Greedy approach. The randomized Rabin-Shallit algorithm required an average of 149.55 μs to resolve a decomposition, successfully finding a solution within the iteration limit in approximately 75% of the trials (experiencing 12,632 timeouts). In stark contrast, the Greedy algorithm resolved 100% of the decompositions in an average of just 2.13 μs, yielding a massive acceleration factor of $\approx 70.3\times$, with a worst-case maximum latency of 374.30 μs.

While Rabin-Shallit strictly guarantees an output of exactly four squares, the Greedy algorithm required an average of 5.37 squares for this bit-length, with a recorded absolute maximum of just 8 squares. However, since the underlying ZKP framework inherently requires appending a padding polynomial to the secret data regardless of the exact number of squares used, this slight increase in the number of elements has practically no impact on the

overall proof size or memory footprint. Given its strictly deterministic execution, absolute reliability and substantial computational acceleration, the Greedy approach perfectly aligns with the efficiency requirements of the protocol.

5.2 Comparison of Rejection Algorithms

This section analyzes the use of different rejection sampling algorithms and compares their impact on the overall computational overhead. While the LNP protocol [LNP22] employs three distinct rejection algorithms (Rej_0, Rej_1, and Rej_2) depending on the specific phase of the proof, the BLNS protocol [BNLS23] uniformly utilizes the Rej_1 algorithm across all its rejection sampling steps.

To establish a fair and consistent comparison, we fix the expected repetition rate M and evaluate the corresponding scaling multiplier γ required by each protocol. Recalling from Section 3.8.2 that the masking standard deviation is strictly defined as $\mathfrak{s} = \gamma\|\mathbf{v}\|$ (where $\mathbf{v}$ is the target secret vector), we can express the expected number of iterations M purely as a function of γ.

Depending on the chosen rejection algorithm, this relationship is formulated, and conversely inverted to compute the required multiplier for a target rate, as follows:

- When applying the Rej_1 algorithm, the expected repetition rate is:

$$M_1 = \exp\left(\frac{28\mathfrak{s}\|\mathbf{v}\| + \|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right) = \exp\left(\frac{28\gamma + 1}{2\gamma^2}\right), \tag{5.7}$$

  By isolating γ via the quadratic formula, we obtain the inverse relation:

$$\gamma_1 = \frac{14 + \sqrt{196 + 2\ln(M_1)}}{2\ln(M_1)}. \tag{5.8}$$

- When applying the Rej_2 algorithm, the rate simplifies to:

$$M_2 = 2\exp\left(\frac{\|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right) = 2\exp\left(\frac{1}{2\gamma^2}\right), \tag{5.9}$$

  Inverting this equation yields:

$$\gamma_2 = \frac{1}{\sqrt{2\ln(M_2/2)}}. \tag{5.10}$$

- Finally, when applying the Rej_0 algorithm, the constant factor of 2 is omitted:

$$M_0 = \exp\left(\frac{\|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right) = \exp\left(\frac{1}{2\gamma^2}\right), \tag{5.11}$$

which can be similarly inverted as:

$$\gamma_0 = \frac{1}{\sqrt{2\ln(M_0)}}\,. \tag{5.12}$$

It is crucial to note that the relationship between the scaling multipliers (γ_1 and γ_0) and the repetition rate M is highly non-linear. Specifically, Rej_1 is drastically more sensitive to parameter tightening than Rej_0. As the target rate approaches its theoretical minimum ($M \to 1^+$), γ_1 grows as $\mathcal{O}(1/\ln M)$, whereas γ_0 scales at a much slower $\mathcal{O}(1/\sqrt{\ln M})$ rate. Due to this structural difference, enforcing a strictly low repetition rate in Rej_1 incurs a massive penalty, causing the required standard deviation to explode. Conversely, Rej_0 remains highly stable across a much wider operational range.

The following analysis examines the various trade-offs associated with utilizing the Rej_1 algorithm compared to its counterparts, Rej_2 and Rej_0.

5.2.1 Comparison Between the Rej_2 and Rej_1 Approaches

As detailed in Section 3.3.2, the primary trade-off between the Rej_2 and Rej_1 algorithms revolves around the single-bit information leakage inherent to Rej_2. In the LNP protocol [LNP22], the Rej_2 algorithm is specifically deployed when bounding the norm of the randomness $\mathbf{s}_2$. Given that $\mathbf{s}_2$ is a one-time value, freshly sampled for every individual credential creation or verification instance, this minimal leakage does not compromise the long-term security of the cryptographic scheme.

However, the Rej_2 algorithm exhibits a fundamental structural limitation. Because the algorithm inherently discards the masked element with a baseline probability of 50%, its expected repetition rate cannot drop below 2 (i.e., $M > 2$). This intrinsic constraint prevents the application of Rej_2 in scenarios where highly optimized, low-iteration overheads are required.

Despite this limitation, analyzing the inverse functions reveals a clear mathematical crossover point. As illustrated in Figure 5.1, for target repetition rates exceeding $M \approx 2.05$, the Rej_2 algorithm becomes more efficient, requiring a smaller scaling multiplier γ compared to Rej_1. Therefore, if the protocol architecture can tolerate a baseline repetition rate above this threshold, employing Rej_2 is an effective choice to minimize standard deviation growth.

5.2.2 Comparison Between the Rej_0 and Rej_1 Approaches

The Rej_0 algorithm is theoretically applicable to any masking element. Compared to Rej_1, it consistently requires a significantly smaller scaling multiplier γ. Consequently, it minimizes the standard deviation required to achieve a target repetition rate, as clearly illustrated in Figure 5.2.

However, this mathematical efficiency comes at an operational cost. As explained in Section 3.3.2, guaranteeing strict zero-knowledge with Rej_0 requires masking the sign element

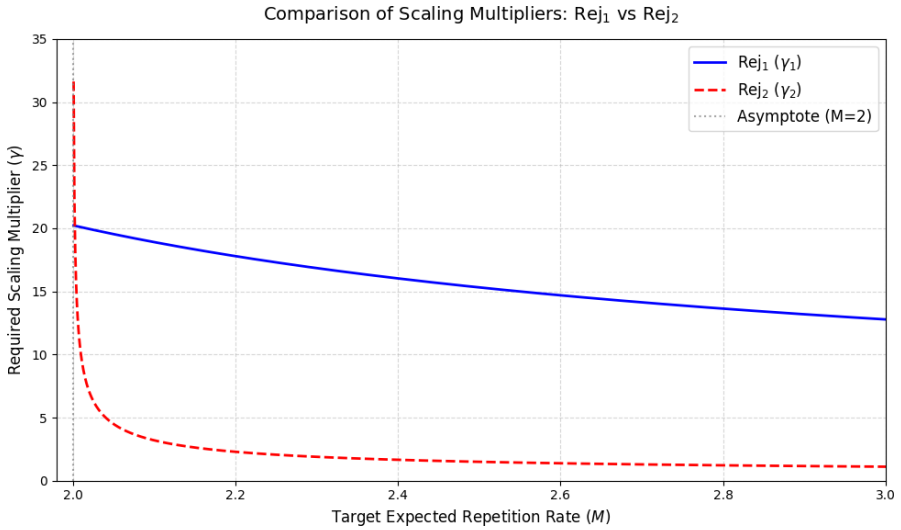

Figure 5.1. Comparison of the required scaling multiplier γ as a function of the target expected repetition rate M for the Rej_1 and Rej_2 algorithms.

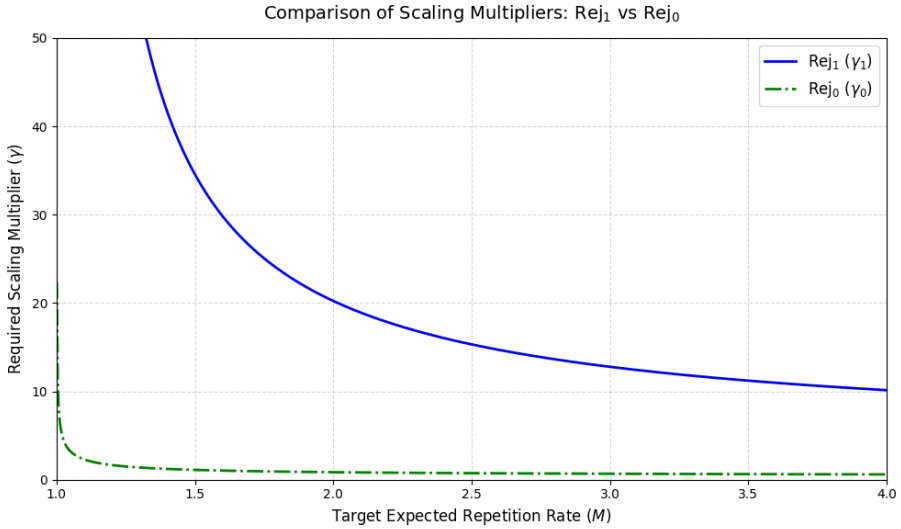

Figure 5.2. Comparison of the required scaling multiplier γ as a function of the target expected repetition rate M for the Rej_1 and Rej_0 algorithms.

$b \in \{-1,1\}$ used to construct the target vector $\mathbf{z} = b\mathbf{v} + \mathbf{y}$. This structural requirement imposes additional computational overhead on the Prover and forces the Verifier to compute and evaluate extra verification equations.

Therefore, the deployment of Rej_0 is justified only when it is strictly necessary to drastically reduce the variance while maintaining a manageable rejection rate. A prime candidate fitting this exact profile is the masking vector $\vec{z}_3$, which provides the theoretical justification for its specific implementation in the LNP protocol [LNP22]. This vector is utilized in the approximate range proof to prevent modulo wrap-around. The scaling multiplier for this vector, denoted as γ_3, directly impacts both the security and efficiency of the protocol. As formalized in Section 3.9, the unified extraction bound B is strictly proportional to γ_3. Given that the protocol's soundness constraints depend on inequalities involving B^2, minimizing γ_3 plays a crucial role in bounding the required size of the modulus q. Ultimately, this optimization contains the parameter space and significantly reduces the overall byte size of the cryptographic proof.

Conversely, an analysis of the constraint on $\mathbf{z}_1$ and $\mathbf{z}_2$ in Equation 3.52 reveals that γ affects the condition only linearly. In fact, the standard deviations $\mathfrak{s}_1$ and $\mathfrak{s}_2$ are strictly proportional to the scaling multiplier γ, and their geometric aggregation under the square root ($\sqrt{\mathfrak{s}_1^2 m_1 + \mathfrak{s}_2^2 m_2}$) ensures that the total bound B scales linearly ($\mathcal{O}(\gamma)$), rather than quadratically.

5.2.3 Summary of Rejection Algorithm Selection

To summarize the trade-off analysis, Rej_1 serves as the optimal baseline algorithm across the protocol, offering a robust balance between verification simplicity and parameter growth. However, this default configuration can be strategically optimized based on the system's operational constraints:

- **Relaxed Repetition Rate ($M_2 > 2.05$):** If the architecture can tolerate a higher rejection baseline, substituting Rej_1 with Rej_2 specifically for the one-time randomness vector $\mathbf{s}_2$ becomes mathematically more efficient, requiring a smaller multiplier γ.

- **Strict Parameters (Low M_3 and minimal lifted modulus):** If the protocol demands a low repetition rate M_3 while keeping the lifted modulus $\hat{q}$ strictly contained, Rej_0 should be applied exclusively to the masking vector $\vec{z}_3$, as it acts as the primary bottleneck for the geometric expansion of the parameters.

5.3 Validation of the LINKS Foundation Parameter Instantiation for Rejection Sampling

In this section, we apply the previously established theoretical framework to validate the exact parameter instantiation implemented in the LINKS Foundation codebase. Specifically, we focus on verifying the security and efficiency bounds of the rejection sampling algorithm (notably Rej_1) by theoretically evaluating the masked response vectors $\mathbf{z}_1$, $\mathbf{z}_2$, and $\vec{z}_3$. Table 5.2 summarizes the structural dimensions, moduli, and statistical constraints utilized for the experimental execution of the optimized protocol.

Parameter	Description	Value
q	Base modulus to generate the credential signature	1 228 913
$\hat{q}_1$	Lifted modulus to evaluate the ZKP equations	1 125 899 907 006 629
d	Ring dimension of the base polynomial ring $\mathcal{R}$	512
$\hat{d}$	Ring dimension of the ZKP evaluation ring $\hat{\mathcal{R}}$	64
$m+2$	Total number of polynomials in the signature vector $\mathbf{s}$	5
η	Attribute commitment scaling parameter	140
m_2	Length of the ZKP randomness vector $\mathbf{s}_2$	64
ℓ	Total number of attributes encoded in the credential	8
ℓ_r	Length of the randomness polynomial vector $\mathbf{r}$	2
h	Fixed weight (number of non-zero coefficients) of the hashed attributes	64
ψ	Gaussian standard deviation bound for the commitment randomness	2
σ_{Falcon}	Gaussian standard deviation for the Falcon trapdoor sampling	165.74
γ	Rejection sampling multiplier	44
ℓ_x	Bit-length of the binary tag x (defining the domain $N = 2^{\ell_x}$)	512

Table 5.2. Concrete parameter instantiation provided by the LINKS Foundation for the optimized credential system.

Evaluating the Bound for $\mathbf{z}_1$

The vector $\mathbf{z}_1$ is utilized to mask the aggregated secret $\mathbf{s}_1$, which is constructed by concatenating the internal vectors $(\mathbf{s}, \mathbf{r}, \vec{u})$ through the linear relation $\mathbf{z}_1 = c\mathbf{s}_1 + \mathbf{y}_1$.

As established in Section 3.8.2, the standard deviation $\mathfrak{s}_1$ governing the masking noise $\mathbf{y}_1$ must be carefully calibrated according to the formula:

$$\mathfrak{s}_1 = \gamma \cdot \eta \cdot B\,, \tag{5.13}$$

where η limits the maximum magnitude of the challenge polynomial c, γ is the scaling multiplier guaranteeing the zero-knowledge property during rejection sampling, and B represents the maximum expected ℓ_2-norm (the unified bound) of the underlying concatenated secret $\mathbf{s}_1$.

To accurately compute this unified bound B, we must systematically evaluate the geometric norm of each constituent block:

Signature Bound (B_s): The secret signature $\mathbf{s}$ is generated via the Falcon trapdoor sampler. Since its coefficients are independently drawn from a discrete Gaussian distribution with standard deviation σ_{Falcon}, the expected squared ℓ_2-norm of the vector equals the variance multiplied by the total dimension. Given that the vector comprises $m + 2 = 5$ polynomials of degree $d = 512$, this theoretical maximum bound evaluates to:

$$B_s = \sigma_{\text{Falcon}}\sqrt{(m+2)d} = 165.74 \cdot \sqrt{5 \cdot 512} \approx 8385.68\,. \tag{5.14}$$

Randomness and Hidden Attributes Bound (B_r): Next, we evaluate the vector embedding the hidden attributes and the commitment randomness. Operating under the worst-case assumption where all $\ell = 8$ attributes remain undisclosed ($|\overline{\text{idx}}| = 8$), the internal components are strictly bounded by the maximum absolute coefficient value $\psi = 2$. Following the structural derivation from Equation 4.5, this yields an absolute maximum bound of:

$$B_r = \psi\sqrt{h \cdot |\overline{\text{idx}}| + \ell_r d} = 2 \cdot \sqrt{64 \cdot 8 + 2 \cdot 512} = 2 \cdot \sqrt{1536} \approx 78.38\,. \tag{5.15}$$

Binary Tag Bound (B_u): The final structural component, $\vec{u}$, represents the binary encoding of the interactive tag x. Since its coefficients are strictly boolean (belonging to $\{0, 1\}$), the maximum squared norm of the vector naturally equates to its total non-zero length. Assuming the tag fully utilizes all available $\ell_x = 512$ bits, the bound is trivial:

$$B_u = \sqrt{\ell_x} = \sqrt{512} \approx 22.63\,. \tag{5.16}$$

Unified Aggregation: Due to the orthogonal nature of the concatenated blocks, the total maximum ℓ_2-norm for the complete secret $\mathbf{s}_1$ is computed by geometrically aggregating the individual bounds:

$$B = \sqrt{B_s^2 + B_r^2 + B_u^2} = \sqrt{8385.68^2 + 78.38^2 + 22.63^2} \approx 8386.08\,. \tag{5.17}$$

Finally, by substituting this unified bound B and the reference parameters from Table 5.2 back into the primary equation, we derive the exact standard deviation required to safely instantiate the masking vector $\mathbf{y}_1$:

$$\mathfrak{s}_1 = \gamma \cdot \eta \cdot B = 44 \cdot 140 \cdot 8386.08 \approx 51\,658\,252.80\,. \tag{5.18}$$

Evaluating the Bound for $\mathbf{z}_2$

The vector $\mathbf{z}_2$ masks the internal randomness $\mathbf{s}_2$ via the relation $\mathbf{z}_2 = c\mathbf{s}_2 + \mathbf{y}_2$. As established in Section 3.8.2, the standard deviation $\mathfrak{s}_2$ of the masking noise $\mathbf{y}_2$ requires scaling the maximum expected ℓ_2-norm of $\mathbf{s}_2$ by the global parameters γ and η, yielding the formula:

$$\mathfrak{s}_2 = \gamma \cdot \eta \cdot \sqrt{m_2 \hat{d}}\,. \tag{5.19}$$

The root term $\sqrt{m_2\hat{d}}$ represents the exact maximum ℓ_2-norm of $\mathbf{s}_2$. Unlike the primary secret, the vector $\mathbf{s}_2$ operates exclusively within the reduced evaluation ring $\hat{\mathcal{R}}$ (dimension $\hat{d}$). Furthermore, its coefficients are sampled from a ternary distribution χ^{m_2}, meaning every coefficient strictly belongs to $\{-1, 0, 1\}$. Consequently, its maximum squared ℓ_2-norm trivially equates to its total structural dimension $m_2\hat{d}$.

By substituting the concrete parameters from Table 5.2 into this established equation, we straightforwardly compute the necessary masking standard deviation:

$$\mathfrak{s}_2 = 44 \cdot 140 \cdot \sqrt{64 \cdot 64} = 394\,240\,. \tag{5.20}$$

Evaluating the Bound for $\vec{z}_3$

The integer scalar vector $\vec{z}_3 \in \mathbb{Z}^{256}$ masks the transformed secret $\vec{s}_1$ via the linear relation $\vec{z}_3 = R\vec{s}_1 + \vec{y}_3$, where R acts as the binomial challenge matrix. As extensively detailed in Section 3.8.3, the standard deviation $\mathfrak{s}_3$ of the noise vector $\vec{y}_3$ requires scaling the unified bound B by the global parameter γ and the matrix expansion factor, yielding the formula:

$$\mathfrak{s}_3 = \gamma \cdot B \cdot \sqrt{337\omega}\,. \tag{5.21}$$

The root term $\sqrt{337\omega}$ represents the theoretical upper bound on the norm expansion induced by the multiplication with the random matrix R. The constant 337 is the specific threshold required to guarantee a target cryptographic security level of $\lambda = 128$, while the parameter ω defines the number of independent Bernoulli trials summed to construct the centered binomial distribution from which R is sampled. For this specific protocol instantiation, the parameter is fixed to $\omega = 1$.

By substituting these structural parameters alongside the previously computed unified norm $B \approx 8386.08$ into the equation, we determine the concrete standard deviation required for the masking noise $\vec{y}_3$:

$$\mathfrak{s}_3 = 44 \cdot 8386.08 \cdot \sqrt{337} \approx 6\,773\,710.52\,. \tag{5.22}$$

Evaluation of the Expected Repetition Rate (M)

Given that the LINKS Foundation implementation exclusively employs the Rej_1 algorithm across all of its rejection sampling steps, we can directly derive the expected repetition

parameter M. Recalling the fundamental scaling relationship $\mathfrak{s} = \gamma\|\mathbf{v}\|$ (where $\mathbf{v}$ represents the unmasked secret vector and $\mathfrak{s}$ its corresponding masking standard deviation), the theoretical repetition rate is formulated as:

$$M = \exp\left(\frac{28\mathfrak{s}\|\mathbf{v}\| + \|\mathbf{v}\|^2}{2\mathfrak{s}^2}\right) = \exp\left(\frac{28\gamma\|\mathbf{v}\|^2 + \|\mathbf{v}\|^2}{2\gamma^2\|\mathbf{v}\|^2}\right) = \exp\left(\frac{14}{\gamma} + \frac{1}{2\gamma^2}\right). \tag{5.23}$$

By substituting the strict security multiplier $\gamma = 44$ defined in the protocol instantiation (as summarized in Table 5.2), we obtain the concrete expectation:

$$M = \exp\left(\frac{14}{44} + \frac{1}{2(44)^2}\right) \approx \exp(0.3184) \approx 1.375\,. \tag{5.24}$$

This result guarantees a highly optimized acceptance probability of approximately 72.7% (calculated as $1/M$) for an isolated execution of the rejection sampling step. However, to evaluate the true computational overhead on the Prover, we must account for the specific architectural flow of the LINKS Foundation protocol (Figure 4.7), which segregates the sampling into two distinct, nested execution loops:

1. The first loop samples the masking vector $\vec{y}_3$ and tests $\vec{z}_3$. Since it depends on a single rejection check, the expected number of iterations to succeed is simply $E[I_1] = M \approx 1.375$.

2. The second loop samples $(\mathbf{y}_1, \mathbf{y}_2)$ and tests the joint acceptance of $(\mathbf{z}_1, \mathbf{z}_2)$. Since a rejection in either condition triggers a restart of this second loop, the joint acceptance probability is approximately $(1/M) \cdot (1/M) = 1/M^2$. Consequently, the expected number of iterations for the second stage is $E[I_2] = M^2 \approx (1.375)^2 \approx 1.891$.

Because these two loops are executed sequentially rather than globally restarting from scratch upon failure, the total expected number of sampling attempts across both stages is additive:

$$M_{\text{tot}} = E[I_1] + E[I_2] = M + M^2 \approx 1.375 + 1.891 = 3.266\,. \tag{5.25}$$

This confirms that the Prover experiences an exceptionally low computational overhead, requiring on average fewer than 3.3 total internal sampling iterations. This structural segregation highlights the critical advantage of the LINKS Foundation optimization: in a monolithic, unsegregated execution where all three checks are batched together, any failure restarts the entire procedure, demanding an overall repetition rate of $M^3 \approx 2.60$ complete protocol executions from scratch. In contrast, the segregated architecture isolates the cost of each stage, ensuring rapid convergence and predictable latency for real-world deployments.

Chapter 6

Conclusion

This thesis presented and rigorously analyzed a post-quantum zero-knowledge protocol for managing anonymous credentials. Specifically, it explored the core components of the foundational zero-knowledge framework [LNP22], deeply investigating the theoretical rationale behind its specific design choices and methodologies. Key elements examined include the ABDLOP hybrid commitment scheme, the transition from ℓ_2-norm bounds to polynomial inner products, and the interactive proof systems for linear and quadratic relations.

Furthermore, the thesis detailed the adaptation of this theoretical framework into a fully functional anonymous credential system [BNLS23]. This involved formalizing the ISIS_f assumption and its variants, alongside the structural mechanics of the credential Issuance and Verification phases.

Finally, a comparative analysis was conducted between the methodologies employed in the foundational LNP framework and the applied BLNS credential scheme. A focal point of this comparison was the mathematical transformation from norm inequalities to exact equalities. To optimize this step, an original Greedy algorithm for integer decomposition was introduced. By leveraging a flexible number of squares, this deterministic approach significantly reduces the average computational overhead. Unlike the randomized Rabin-Shallit algorithm, the Greedy approach strictly prevents worst-case execution bottlenecks, eliminating high variance in runtime and probabilistic resampling iterations. Software simulations empirically confirmed its overwhelming performance superiority over the standard four-square decomposition.

The thesis also evaluated different rejection sampling strategies, assessing the trade-offs of each approach. The analysis demonstrated that the bimodal strategy (Rej_2) operates most efficiently when a repetition rate of $M > 2.05$ is acceptable, making it ideal for the secret randomness vector $\mathbf{s}_2$. Conversely, the standard bimodal approach (Rej_0) is highly recommended when low variance and a low repetition rate M are strictly required. This perfectly aligns with the requirements for $\vec{z}_3$, the compressed masked vector used to prevent modular wrap-around.

In conclusion, the analyses and optimizations proposed in this work demonstrate that

transitioning to lattice-based assumptions does not inherently compromise protocol efficiency. By carefully balancing mathematical design choices, it is possible to construct highly secure, post-quantum anonymous credential systems that remain practical for real-world deployment.

Bibliography

[LNP22] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. *Lattice based-zero-knowledge proofs and applications: Shorter, simpler, and more general*. In *Advances in Cryptology — CRYPTO 2022*, pages 208–238. Springer, 2022.

[BNLS23] Bootle, J., Nguyen, N. K., Lyubashevsky, V., & Sorniotti, A. (2023). *A Framework for Practical Anonymous Credentials from Lattices*. IBM Research Europe, Zurich & EPFL, Switzerland.

[VC] Photo taken from https://rc.sunbird.org/help/comprehensive-overview-electronic-registries-and-verifiable-credentials/verifiable-credentials/what-are-the-key-roles-in-verifiable-credentials

[ASM06] Man Ho Au, Willy Susilo, and Yi Mu. *Constant-size dynamic k-TAA*. In *International Conference on Security and Cryptography for Networks*, pages 111–125. Springer, 2006.

[PS16] David Pointcheval and Olivier Sanders. *Short randomizable signatures*. In *The Cryptographers' Track at the RSA Conference*, pages 111–126. Springer, 2016.

[CL02] Jan Camenisch and Anna Lysyanskaya. *A signature scheme with efficient protocols*. In *Advances in Cryptology — CRYPTO 2002*, pages 268–289. Springer, 2002.

[LS15] Adeline Langlois and Damien Stehlé. *Worst-case to average-case reductions for module lattices. Designs, Codes and Cryptography*, 75(3):565–599, 2015.

[Ban93] Wojciech Banaszczyk. *New bounds in some transfer theorems in the geometry of numbers. Mathematische Annalen*, 296(4):625–635, 1993.

[Ste93] Jacques Stern. *A new identification scheme based on syndrome decoding*. In *Advances in Cryptology — CRYPTO '93*, pages 13–21. Springer, 1993.

[Beu20] Ward Beullens. *Sigma protocols for MQ, PKP and LWE with relaxed soundness*. In *Advances in Cryptology — CRYPTO 2020*, pages 54–83. Springer, 2020.

[BLS19] Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. *Algebraic techniques for short(er) lattice-based zero-knowledge proofs*. In *Advances in Cryptology — CRYPTO 2019*, pages 317–346. Springer, 2019.

[YAZ+19] Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. *Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications*. In *Advances in Cryptology — CRYPTO 2019*, pages 277–316. Springer, 2019.

[ESLL19] Muhammed F. Esgin, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. *Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications*. In *Advances in Cryptology — CRYPTO 2019*, pages 115–146. Springer, 2019.

[ALS20] Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. *Practical product proofs for lattice commitments*. In *Advances in Cryptology — CRYPTO 2020*, pages 470–499. Springer, 2020.

[ENS20] Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. *Practical exact proofs from lattices: New techniques to exploit fully-splitting rings*. In *Advances in Cryptology — ASIACRYPT 2020*, pages 259–288. Springer, 2020.

[LNS21a] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. *Shorter lattice-based zero-knowledge proofs via one-time commitments*. In *Public-Key Cryptography — PKC 2021*, pages 215–241. Springer, 2021.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In STOC, pages 99–108, 1996.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. *A Pseudorandom Generator from any One-way Function*. SIAM Journal on Computing, 28(4):1364–1396, 1999.

[BDL+18] Carsten Baum, Ivan Damgard, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In SCN, pages 368–385, 2018.

[Lyu09] Vadim Lyubashevsky. *Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures*. In *Advances in Cryptology — ASIACRYPT 2009*, pages 598–616. Springer, 2009.

[Lyu12] Vadim Lyubashevsky. *Lattice Signatures without Trapdoors*. In *Advances in Cryptology — EUROCRYPT 2012*, pages 738–755. Springer, 2012.

[LNS21b] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. *Practical Lattice-Based Zero-Knowledge Proofs for Integer Relations*. In *Advances in Cryptology — EUROCRYPT 2021*, pages 632–660. Springer, 2021.

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. *Lattice Signatures and Bimodal Gaussians*. In *Advances in Cryptology — CRYPTO 2013*, pages 40–56. Springer, 2013.

[GHL21] Craig Gentry, Shai Halevi, and Vadim Lyubashevsky. "Practical non-interactive publicly verifiable secret sharing with thousands of parties." *IACR Cryptol. ePrint Arch.*, page 1397, 2021.

[FS86] Fiat, A., & Shamir, A. (1986). How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology—CRYPTO' 86* (pp. 186–194). Springer Berlin Heidelberg.

[PFH+17] Prest, T., Fouque, P.-A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., & Zhang, Z. (2017). *FALCON*. Tech. rep.

National Institute of Standards and Technology.

[KY17] S. Katsumata and S. Yamada. *Group Signatures without NIZK: From Lattices in the Standard Model*. In *Advances in Cryptology – EUROCRYPT 2017*, pages 312–344. Springer, 2017.

[RS86] M. O. Rabin and J. O. Shallit. *Randomized Algorithms in Number Theory*. In *Communications on Pure and Applied Mathematics*, 39(S1):S239–S256, 1986.

[SS11] Damien Stehlé and Ron Steinfeld. *Making NTRU as Secure as Standard Worst-Case Problems over Ideal Lattices*. In *Advances in Cryptology — EUROCRYPT 2011*, pages 27–47. Springer, 2011.

[Lem1896] Lemoine, É. (1896). Sur la décomposition d'un nombre en ses carrés maxima. *Association Française pour L'Avancement des Sciences*, 73–77.