Compact and Secure Zero-Knowledge Proofs for Quantum-Resistant Cryptography from Modular Lattice Innovations

Samuel Lavery

sam@trustlessprivacy.com

April 29, 2024

Abstract

This paper presents a comprehensive security analysis of the Adh zero-knowledge proof system, a novel lattice-based, quantum-resistant proof of possession system. The Adh system offers compact key and proof sizes, making it suitable for real-world digital signature and public key agreement protocols. We explore its security by reducing it to the hardness of the Module-ISIS problem and introduce three new variants: Module-ISIS+, Module-ISIS*, and Module-ISIS**. These constructions enhance security through variations on chaining mechanisms. We also provide a reduction to the module modulus subset sum problem under conservative assumptions.

Empirical evidence and statistical testing support the zero-knowledge, completeness, and soundness properties of the Adh proof system. Comparative analysis demonstrates the Adh system's advantages in terms of key and proof sizes over existing post-quantum schemes like Kyber and Dilithium.

This paper represents an early preprint and is a work in progress. The core security arguments and experimental results are present, and formal proofs and additional analysis are provided. We invite feedback and collaboration from the research community to further strengthen the security foundations of the Adh system and explore its potential applications in quantum-resistant cryptography.

Contents

1 Introduction

As the quantum computing era approaches, the imperative for quantum-resilient cryptographic systems becomes increasingly urgent. The Adh zero-knowledge proof system addresses this need by leveraging the hardness of the Module-ISIS problem, offering a robust solution designed to withstand future quantum threats.

This paper explores the Adh zero-knowledge proof system, a novel quantum-resilient solution based on the hardness of the Module-ISIS problem. We introduce key innovations such as nested Number Theoretic Transforms (NTT), extreme rejection sampling, and novel chaining constructions that collectively enhance security without increasing communication overhead.

Nested NTT operations enhance polynomial arithmetic efficiency and security through increased confusion and diffusion, akin to mid-round modulus switching. Combined, along with our novel chaining constructions, forms a dense lattice structure that robustly defends against diverse attacks.

A key strength of the Adh system lies in its extensive use of rejection sampling of 0 value coefficients. By eliminating zero coefficients in the lattice basis, the Adh system constructs a full lattice structure with high density. This property significantly enhances attack resilience, as the absence of sparsity renders many common lattice reduction techniques less effective. The complete lattice structure ensures that the system is as reduced as possible, making it challenging for adversaries to exploit vulnerabilities.

The core computational hardness of the Adh system is based on the Module-ISIS problem, which requires finding an exact solution to the equation $\mathbf{t} = \mathbf{A} \cdot \mathbf{z} \bmod q$ for a target vector $\mathbf{t}$. This problem is considered harder than other approximation-based lattice problems due to the additional constraint of matching an exact target vector. We introduce three new variants of the Module-ISIS problem and provide reductions from these variants to the original Module-ISIS problem. Additionally, by relaxing the constraint of multiplication to addition, we establish a reduction to the Module Modulus Subset Sum Problem, further strengthening the security argument of the Adh system.

Table 1 presents the key parameters and estimated security strengths of the Adh system for two different dimensions, $n = 128$ and $n = 256$.

Parameter	**n=128**	**n=256**
Public Key Size	192 bytes	384 bytes
Secret Key Size	192 bytes	384 bytes
Signature/Key Agreement Proof Size	192 bytes	384 bytes
Original Estimate Bits of Security	112 bits	260 bits
Demonstrated Bits of Security	331 bits	673 bits
Theoretical Max Bits of Security	448 bits	1040 bits

Table 1: Adh system parameters and security strengths for different dimensions.

The Adh system achieves compact key and proof sizes, with 192 bytes for $n = 128$ and 384 bytes for $n = 256$. While the original calculated hardness was 112 bits and 260 bits for $n = 128$ and $n = 256$, respectively, our analysis demonstrates a significant increase after applying the techniques presented in this paper. The theoretical estimates for the new constructions reach 448 bits for $n = 128$ and 1040 bits for $n = 256$. Remarkably, our experimental results indicate bit security strengths of 331 bits and 673 bits for $n = 128$ and $n = 256$, respectively. The impact of more accurate BKZ cost estimates on bit security remains an open research question. Nonetheless, this work showcases the effectiveness of the full lattice structure and the chaining mechanism employed in the Adh system.

The comparison of the Adh system with the widely-recognized post-quantum cryptographic schemes Kyber (ML-KEM) and Dilithium (ML-DSA) highlights the significant advantages of the Adh system in terms of key and ciphertext/signature sizes. The Adh system achieves substantially smaller public keys, secret keys, and ciphertexts/signatures compared to both Kyber and Dilithium at their respective security levels.

Metric	Adh-128	Adh-256	ML-KEM1	ML-KEM5	ML-DSA3	ML-DSA5
PK	192B	384B	736B	1440B	1472B	2592B
SK	192B	384B	1632B	3168B	4000B	4864B
CT/SIG	192B	384B	768B	1568B	3293B	4595B
BitSec	331 bits	673 bits	118 bits	256 bits	192 bits	256 bits
	Experimental	Experimental	Proven	Proven	Proven	Proven

Table 2: Comparison of Adh, Kyber, and Dilithium parameters and security strengths.

For example, at a demonstrated bit security level of 331 bits, the Adh-128 variant requires only 192 bytes for each of its public key, secret key, and ciphertext/signature. In contrast, Kyber-512, which offers a proven bit security level of 118 bits, has a public key size of 736 bytes, a secret key size of 1632 bytes, and a ciphertext size of 768 bytes. Similarly, Dilithium-3, with a proven bit security level of 192 bits, has a public key size of 1472 bytes, a secret key size of 4000 bytes, and a signature size of 3293 bytes.

The Adh-256 variant, which demonstrates a bit security level of 673 bits, maintains a compact size of 384 bytes for its public key, secret key, and ciphertext/signature. This is a remarkable achievement considering that Kyber-1024 and Dilithium-5, which offer proven bit security levels of 256 bits, have much larger key and ciphertext/signature sizes.

The smaller sizes not only lead to reduced storage requirements but also result in improved efficiency in terms of communication bandwidth and processing overhead. Beyond that, smaller key and ciphertext/signature sizes of the Adh system make it an attractive candidate for resource-constrained environments, such as embedded systems and IoT devices, where memory and bandwidth are limited. Additionally, the reduced sizes can lead to faster key generation, encryption, decryption, signing, and verification operations, thereby enhancing the overall performance of cryptographic protocols built upon the Adh system.

Furthermore, the compact sizes of the Adh system, combined with its post-quantum security, make it a promising solution for future-proofing cryptographic implementations. As the threat of quantum computers looms on the horizon, the Adh system offers a secure and efficient alternative to traditional cryptographic schemes that are vulnerable to quantum attacks. The smaller key and ciphertext/signature sizes also facilitate easier migration from classical to post-quantum cryptography, as they minimize the impact on existing systems and protocols.

Thesis 1. *The Adh zero-knowledge proof system is secure under the hardness assumption of the Module-ISIS problem, providing soundness, completeness, and zero-knowledge properties.*

2 Slicing into the Variants of the Module-ISIS Problem: A Pie Analogy

In lattice-based cryptography, the Module-ISIS problem and its variants serve as a foundation for constructing secure cryptographic primitives. To elucidate the differences and relationships between the variants described in this paper, we present an analogy based on pies. Let us explore the distinct flavors of Module-ISIS, ISIS+, ISIS*, and ISIS**, and uncover the complexities that each variant introduces.

Consider the Module-ISIS problem as a classic pumpkin pie—homogeneous, consistent, and unambiguous in its composition. The Module-ISIS problem presents a well-defined lattice structure, just as every slice of pumpkin pie offers a uniform taste and texture.

Module-ISIS+ can be thought of as an apple pie, where the filling consists of distinct slices of apples, each with its own unique characteristics, yet harmoniously combined to form a cohesive whole. Each slice of apple represents an instance of the Module-ISIS problem, chained together to create a more intricate composition. The ISIS+ construction uses a chaining mechanism, similar to WOTS+, to bind the components of the problem together. While each slice is made of apple, each piece of apple represents its own instance of the Module-ISIS problem to solve.

Module-ISIS* can be likened to a mixed berry pie, where the filling is a medley of similar yet distinct problems, each with its own secret ingredients. The assortment of berries represents the variations in the problem instances while maintaining a relationship with the original Module-ISIS problem. The various types of fruit symbolize individual instances of the Module-ISIS problem, with the additional constraint of part of the chain having a distinct secret key.

These crustless pie constructions, Module-ISIS+ and ISIS*, can be reduced to well-established hard lattice problems. The hardness of these variants is rooted in the underlying hardness of the Module-ISIS problem.

Now, consider ISIS**. If the previous variants were pies without a crust, ISIS** is the golden, flaky crust that elevates the pie to new heights of complexity. The crust represents the additional features introduced by ISIS**, such as projection to higher dimensions, modular addition, and the inversion back to the input domain. While the increased complexity brought by ISIS** is not formally proven in this paper, empirical evidence suggests that the pie with crust exhibits a more intricate internal structure.

The presence of the crust (ISIS**) is unlikely to make the core pie problems easier to solve. We conjecture that the added complexity of ISIS** enhances the difficulty of the problem, but a formal proof requires further research. The solution to the "soggy bottom" problem remains an open question in the field of lattice-based cryptography. The rest of this paper is structured as follows:

- Preliminary Notations, Definitions and Concepts.
- A high level overview of the proof system.
- Problem definitions
- Security Analysis
- Reduction to Module-ISIS variants
- Reduction to Subset Sum
- Implementation Considerations
- Experimental Results
- Performance Analysis
- Use Cases and Applications
- Comparative Analysis, Known Problems, Conclusion and Future Work
- Detailed Appendix

3 Preliminaries

3.1 Notation and Definitions

Throughout this paper, we use the following notation:

- $\mathbb{Z}_q$: The ring of integers modulo q.
- $\mathbb{Z}_q[x]$: The ring of polynomials over $\mathbb{Z}q$.
- $R_q = \mathbb{Z}q[x]/(x^n + 1)$: The quotient ring of polynomials modulo $x^n + 1$, where n is a power of 2.
- $\mathbf{a} \in R_q^m$: A vector of m polynomials in R_q.
- $\mathbf{A} \in R_q^{m\times m}$: A matrix of $m \times m$ polynomials in R_q.
- $||\mathbf{a}||\infty$: The infinity norm of a vector $\mathbf{a}$, defined as $||\mathbf{a}||\infty = \max i|a_i|$.

We also define the following terms:

Definition 1 (Zero Vector)**.** *A vector* $\mathbf{a} \in R_q^m$ *is called a zero vector if all its coefficients are zero.*

Definition 2 (Sparse Vector)**.** *A vector* $\mathbf{a} \in R_q^m$ *is called a sparse vector if it contains a significant number of zero coefficients.*

Definition 3 (Full Vector)**.** *A vector* $\mathbf{a} \in R_q^m$ *is called a full vector if all its coefficients are non-zero.*

Definition 4 (Sparse Lattice)**.** *A lattice* $\mathcal{L}$ *is called a sparse lattice if it is generated by a basis matrix containing a significant number of zero coefficients.*

Definition 5 (Complete Lattice)**.** *A lattice* $\mathcal{L}$ *is called a complete lattice if it is generated by a basis matrix containing only non-zero coefficients.*

3.2 Unique Features

The Adh system incorporates several unique features that distinguish it from other zero-knowledge proof systems:

- **Nested NTT Calls**: The ZKVolute function used in the Adh system employs recursive NTT operations, allowing for efficient polynomial arithmetic, maintaining the necessary algebraic structure, while allowing for a diffusive dimensional shift and mix operation.
- **Rejection Sampling**: The rejection sampling technique is used throughout the Adh system to ensure that all the vectors involved are full vectors, eliminating the presence of zero coefficients and maintaining a complete lattice structure.
- **Chaining Functions**: Adh implements multiple WOTS+ like chaining function using number theoretic primitives to amplify hardness of the core module-ISIS problem, especially the module-ISIS* instance.

3.3 Module-ISIS Problem

The Module-ISIS (Module Inhomogeneous Short Integer Solution) problem is a lattice-based cryptographic problem that generalizes the SIS problem[12] to rings. It is defined as follows:

Definition 6. *(Module-ISIS Problem) Given a uniformly random matrix* $\mathbf{A} \in R_q^{m\times n}$*, a target vector* $\mathbf{t} \in R_q^m$*, and a predefined bound* β*, find a non-zero vector* $\mathbf{z} \in R_q^n$ *such that:*

$$\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \pmod q ||\mathbf{z}||_\infty \leq \beta \tag{1}$$

Explanation:

- Ring Setting: Module-ISIS operates over the ring of polynomials modulo a prime q, denoted as R_q. This allows for more compact representations and efficient operations compared to standard lattices.
- Dimensions: The matrix $\mathbf{A}$ has dimensions $m \times n$. Typically, Module-ISIS instances are set up with more columns than rows ($n > m$).
- Hardness Basis: The computational difficulty of the Module-ISIS problem is believed to be linked to the worst-case hardness of specific lattice problems over module lattices, such as the Shortest Independent Vectors Problem (SIVP) in this context.
- **Complexity Comparison**: The Module-ISIS problem is considered to be at least as hard as the Module-SIS problem. In the Module-SIS problem, the goal is to find a short non-zero vector $\mathbf{z}$ such that $\mathbf{A} \cdot \mathbf{z} = \mathbf{0} \pmod q$, where $\mathbf{A}$ is a random matrix. In contrast, the Module-ISIS problem requires finding a short non-zero vector $\mathbf{z}$ such that $\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \pmod q$, where $\mathbf{t}$ is a target vector. The additional constraint of matching a specific target vector $\mathbf{t}$ makes the Module-ISIS problem potentially harder than Module-SIS.

3.4 Number Theoretic Transform (NTT)

The Number Theoretic Transform (NTT) is a special case of the Discrete Fourier Transform (DFT) over a finite field. It is widely used in lattice-based cryptography for efficient polynomial multiplication. The NTT has the following properties:

- It is a bijective linear transformation that maps a vector of coefficients to another vector of coefficients.
- It preserves the structure of the polynomial ring, allowing for efficient polynomial arithmetic.
- The forward and inverse NTT operations can be computed in $O(n \log n)$ time using the Cooley-Tukey algorithm.

In the Adh system, the NTT plays a crucial role in the construction of the proof and verification algorithms, enabling efficient computations and maintaining the necessary algebraic structures.

4 The Adh Zero-Knowledge Proof System

In this section, we provide a detailed description of the Adh zero-knowledge proof system, including its key generation, proof generation, and verification algorithms. We also highlight the unique features of the system, such as nested NTT calls, multiple levels, and rejection sampling.

4.1 Overview

The Adh system is a lattice-based zero-knowledge proof of possession system that aims to provide quantum-resilient security. It leverages the hardness of the Module-ISIS problem and employs a novel construction based on nested NTT operations and rejection sampling techniques.

4.2 Assumptions

The security of the Adh system relies on the following assumptions:

Assumption 1 (Module-ISIS Hardness). *The Module-ISIS problem is computationally hard for the chosen parameters (q, n, m, β). Specifically, no probabilistic polynomial-time algorithm can solve the Module-ISIS problem with non-negligible probability.*

Assumption 2 (NTT Invertibility). *The NTT operation used in the Adh system is a bijective mapping that preserves the structure of the polynomial ring R_q. The inverse NTT operation exists and can be efficiently computed.*

Assumption 3 (Rejection Sampling Uniformity). *The rejection sampling technique employed in the Adh system produces uniformly distributed full vectors and complete lattices, eliminating the presence of zero coefficients.*

Assumption 4 (Pseudorandomness of Iterated NTT). *The iterated NTT operation, denoted as $NTT^{(i)}$, is assumed to exhibit pseudorandom behavior when applied to uniformly random inputs, making it computationally indistinguishable from a truly random function when chosen decisionaly from set of possible NTT representations.*

4.3 Key Generation

The core key generation algorithm of the Adh system proceeds as follows:

1. Generate a uniformly random secret key $\mathbf{sk} \in R_q^m$ with coefficients in the range $[1, q-1]$.
2. Apply rejection sampling to ensure that $\mathbf{sk}$ is a full vector.
3. Generate a uniformly random public challenge $\mathbf{pk}$_chal $\in R_q^m$ with coefficients in the range $[1, q-1]$.
4. Apply rejection sampling to ensure that $\mathbf{pk}$_chal is a full vector.
5. Generate a uniformly random public randomness $\mathbf{pk}$_rand $\in R_q^m$ with coefficients in the range $[1, q-1]$.
6. Apply rejection sampling to ensure that $\mathbf{pk}$_rand is a full vector.
7. Compute the public key $\mathbf{pk}$ as $\mathbf{pk} = \text{ZKVolute\_}(\mathbf{sk}, \mathbf{pk}\text{\_chal}, \mathbf{pk}\text{\_rand})$, where ZKVolute is a function that performs nested NTT operations and polynomial arithmetic.
8. Output the public key $\mathbf{pk}$ and the secret key $\mathbf{sk}$.
9. The storage format of the public key is composed of the public challenge, public random and $\mathbf{pk}$ and the secret key $\mathbf{sk}$ also includes both public values in order to regenerate the public key correctly.

The key generation algorithm ensures that all the vectors involved (secret key, public challenge, and public randomness) are full vectors, eliminating the presence of zero coefficients. This property is crucial for the security and correctness of the Adh system.

4.4 Proof Generation

The proof core generation algorithm of the Adh system takes as input the secret key $\mathbf{sk}$, a message $\mathbf{m}$, and a public challenge $\mathbf{pk}$_chal. It proceeds as follows:

1. Generate a uniformly random signature challenge $\mathbf{sig}\_\text{chal} \in R_q^m$ as a function of m via hash_to_poly with coefficients in the range $[1, q-1]$.
2. Apply rejection sampling to ensure that **sig**_chal is a full vector.
3. Generate a uniformly random signature randomness $\mathbf{sig}\_\text{rand} \in R_q^m$ with coefficients in the range $[1, q-1]$.
4. Apply rejection sampling to ensure that **sig**_rand is a full vector.
5. Compute the proof **sig** as $\mathbf{sig} = \text{ZKVolute\_}(\mathbf{sk}, \mathbf{sig}\_\text{chal}, \mathbf{sig}\_\text{rand})$.
6. Output the proof **sig** along with **sig**_chal and **sig**_rand.

The proof generation algorithm ensures that the signature challenge and signature randomness are full vectors, maintaining the complete lattice structure throughout the computation.

4.5 Verification

The verification algorithm of the Adh system takes as input the public key **pk**, the proof **sig**, the signature challenge **sig**_chal, and the signature randomness **sig**_rand. It proceeds as follows:

1. Compute the left-hand side **lhs** as $\mathbf{lhs} = \text{ZKVolute}(\mathbf{pk}, \mathbf{sig}\_\text{chal}, \mathbf{sig}\_\text{rand})$.
2. Compute the right-hand side **rhs** as $\mathbf{rhs} = \text{ZKVolute}(\mathbf{sig}, \mathbf{pk}\_\text{chal}, \mathbf{pk}\_\text{rand})$.
3. Check if $\mathbf{lhs} = \mathbf{rhs}$. If true, accept the proof; otherwise, reject it.

The core verification algorithm leverages the equivariance property of the ZKVolute function to check the validity of the proof. The use of nested NTT operations and rejection sampling ensures that all the vectors involved in the verification process are full vectors, maintaining the complete lattice structure.

5 Problem Definitions

5.1 Module-ISIS+ definition

Definition 7 (Module-ISIS+ Problem). *Let k be a positive integer denoting the number of chained instances. Given a uniformly random matrix:*

$$\mathbf{A}_1 \in R_q^{m \times m} \tag{2}$$

and a set of target vectors

$$\mathbf{t}_1, \ldots, \mathbf{t}_k \in R_q^m \tag{3}$$

find a non-zero vector $\mathbf{z} \in R_q^m$ such that:

$$\mathbf{A}_1 \cdot \mathbf{z} = \mathbf{t}_1 \bmod q \tag{4}$$

$$\mathbf{A}_2 \cdot \mathbf{z} = \mathbf{t}_2 \bmod q \tag{5}$$

$$\vdots \tag{6}$$

$$\mathbf{A}_k \cdot \mathbf{z} = \mathbf{t}k \bmod q \tag{7}$$

where $\mathbf{A}i = NTT(\mathbf{A}i-1) \cdot NTT(\mathbf{R})$ for $i = 2, \ldots, k$, with $\mathbf{R}$ being a random matrix in $R_q^{m\times m}$, and $||\mathbf{z}||\infty \le \beta$.

The Module-ISIS+ problem captures the chaining mechanism of the Adh system, where each instance is related to the previous one through an NTT operation and a random matrix multiplication. The hardness of Module-ISIS+ is based on the hardness of the underlying Module-ISIS problem.

Theorem 1 (Reduction to Module-ISIS+)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS+ problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS+ problem. Given a Module-ISIS+ instance $(\mathbf{A}_1, \mathbf{t}_1, \ldots, \mathbf{t}_k, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS+ instance.
2. $\mathcal{B}$ generates the public key $\mathbf{pk}$ and sends it to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a forged proof $(\mathbf{sig}'\mathbf{sig}\_\text{chal}'\mathbf{sig}\_\text{rand}^{)}$.
4. $\mathcal{B}$ computes $\mathbf{z} = \mathbf{sig}^{-}\mathbf{sig}$, where $\mathbf{sig}$ is a valid proof generated by $\mathcal{B}$.
5. If $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||\infty \leq 2\beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the Module-ISIS+ instance.

A complete proof is provided in Appendix A.2. □

This reduction shows that if an adversary can forge a valid proof in the Adh system, then they can solve the Module-ISIS+ problem, which is assumed to be computationally infeasible for appropriately chosen parameters. Therefore, the Adh system is secure against forgery attacks, assuming the hardness of Module-ISIS+.

The reduction to Module-ISIS+ captures the chaining mechanism of the Adh system and provides a stronger security guarantee compared to the basic Module-ISIS problem. It demonstrates that forging a valid proof in the Adh system is at least as hard as solving the Module-ISIS+ problem, which is a generalization of the Module-ISIS problem that takes into account the multiple chained instances and the NTT operations used in the Adh system.

5.1.1 Module-ISIS* Problem and Its Application to the Adh System

In this section, we introduce a variant of the Module-ISIS+ problem, which we call Module-ISIS*, and discuss its potential application to the Adh zero-knowledge proof system. The Module-ISIS* problem incorporates the use of multiple secret keys, one for each instance of the module lattice, to enhance the hardness of the problem against lattice reduction and algebraic attacks.

5.2 Definition of Module-ISIS*

Definition 8 (Module-ISIS* Problem)**.** *Let k be a positive integer denoting the number of chained instances. Given uniformly random matrices $\mathbf{A}_1, \ldots, \mathbf{A}_k \in R_q^{m \times m}$ and a set of* *target vectors* $\mathbf{t}_1, \ldots, \mathbf{t}_k \in R_q^m$*, find non-zero vectors* $\mathbf{z}_1, \ldots, \mathbf{z}_k \in R_q^m$ *such that:*

$$\begin{aligned}
\mathbf{A}_1 \cdot \mathbf{z}_1 &= \mathbf{t}_1 \bmod q \\
\mathbf{A}_2 \cdot \mathbf{z}_2 &= \mathbf{t}_2 \bmod q \\
&\vdots \\
\mathbf{A}_k \cdot \mathbf{z}_k &= \mathbf{t}_k \bmod q
\end{aligned}$$

where $\mathbf{t}_i = mask(\mathbf{A}i \cdot \mathbf{z}i - 1) \cdot \mathbf{z}_i$ *for* $i = 2, \ldots, k$*, with* $\mathbf{t}_1 = \mathbf{A}_1 \cdot \mathbf{z}_1$*, and* $||\mathbf{z}i||\infty \leq \beta$ *for all* i*.*

The key difference between Module-ISIS* and Module-ISIS+ is that in Module-ISIS*, each instance of the module lattice uses a unique secret key $\mathbf{z}_i$, whereas in Module-ISIS+, a single secret key $\mathbf{z}$ is used to generate the target vector $\mathbf{t}$ for the next lattice instance. In Module-ISIS*, the target vectors $\mathbf{t}_i$ are obtained by masking the product $\mathbf{A}i \cdot \mathbf{z}i - 1$ and multiplying it with the current secret key $\mathbf{z}_i$, creating a chain of dependencies between the instances.

5.2.1 Hardness of Module-ISIS*

The use of multiple secret keys in Module-ISIS* adds an extra layer of complexity to the problem, potentially making it harder to solve using lattice reduction and algebraic techniques. Intuitively, an attacker would need to simultaneously recover all the secret keys $\mathbf{z}_1, \ldots, \mathbf{z}_k$ to solve the problem, which could be more challenging than recovering a single secret key as in Module-ISIS+. The introduction of multiple secret keys and the chaining mechanism in Module-ISIS* creates a new problem structure that requires further analysis to establish its hardness formally.

One potential approach to analyzing the hardness of Module-ISIS* is to consider the complexity of solving the problem using lattice reduction algorithms. The use of multiple secret keys and the chaining mechanism may increase the dimension and density of the lattices involved, making them more resistant to lattice reduction attacks. We provide experimental results in subsequent sections.

5.2.2 Application to the Adh System

Incorporating the Module-ISIS* problem into the Adh zero-knowledge proof system potentially enhances its security. Instead of using a single secret key to generate the target vector for the next lattice instance, the prover would generate a unique secret key for each instance and use them to compute the proofs accordingly. The verification algorithm would need to be modified to account for the multiple secret keys. The verifier would compute the left-hand side and right-hand side of the verification equation using the appropriate secret keys and public parameters for each instance.

While the use of multiple secret keys may increase the storage requirements and computational overhead of the Adh system, it could provide an additional layer of security against potential attacks. The increased complexity introduced by the Module-ISIS* problem may make it more challenging for an adversary to forge proofs or recover the secret keys.

However, it is crucial to carefully analyze the impact of using Module-ISIS* on the security of the Adh system. Further research is needed to ensure that the use of multiple secret keys does not introduce any unforeseen vulnerabilities or weaknesses that could be exploited by an adversary.

5.2.3 Future Directions

The Module-ISIS* problem and its application to the Adh system open up several avenues for future research:

- Investigating the concrete security of the Adh system when instantiated with Module-ISIS* with different parameters.
- Exploring the trade-offs between the increased security and the additional storage and computational requirements introduced by the use of multiple secret keys.
- Studying potential optimizations and efficiency improvements to the Adh system when using Module-ISIS*.

In conclusion, the Module-ISIS* problem presents an interesting variant of Module-ISIS+ that incorporates the use of multiple secret keys. While it has the potential to enhance the security of the Adh zero-knowledge proof system, further research is needed to formally establish its hardness, analyze its impact on the system's security, and explore its practical implications. The Module-ISIS* problem opens up new possibilities for designing lattice-based cryptographic protocols with enhanced security guarantees, and it warrants further investigation by the cryptographic community.

5.3 Definition of Module-ISIS**

In this section, we present a refined variant of the Module-ISIS* problem, called Module-ISIS**, which incorporates the use of different roots of unity and/or primes at each level of the chained instances of recursive NTT transformations. This approach aims to enhance the security of the Adh zero-knowledge proof system by introducing distinct algebraic structures at each stage. This structure serves to obfuscate the real underlying lattice basis underneath it.

Definition 9 (Module-ISIS** Problem)**.** *Let k be a positive integer denoting the number of chained instances, and let p_i be a prime modulus. Let $\omega_1, \ldots, \omega_k$ be distinct roots of unity for each level. Given uniformly random matrices $\mathbf{A}_1, \ldots, \mathbf{A}_k \in R_p^{m\times m}$ and a set of target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_k \in R_p^m$, find non-zero vectors $\mathbf{z}_1, \ldots, \mathbf{z}_k \in R_p^m$ such that:*

$$\begin{aligned} \mathbf{A}_1 \cdot \mathbf{z}_1 &= \mathbf{t}_1 \bmod p_1 \\ \mathbf{A}_2 \cdot \mathbf{z}_2 &= \mathbf{t}_2 \bmod p_2 \\ &\vdots \\ \mathbf{A}_k \cdot \mathbf{z}_k &= \mathbf{t}_k \bmod p_k \end{aligned}$$

where $\mathbf{t}_i = mask(\mathbf{A}i \cdot \mathbf{z}i - 1) \cdot \mathbf{z}_i$ for $i = 2, \ldots, k$, with $\mathbf{t}_1 = \mathbf{A}_1 \cdot \mathbf{z}_1$, and $||\mathbf{z}i||\infty \leq \beta$ for all i.

In Module-ISIS**, all levels of the chained instances may use the same prime modulus p for all p_i, ensuring consistency in the problem space. However, each level may also use unique, increasing values for p_i with an alternative root of unity ω_i, introducing distinct algebraic structures at each stage.

5.3.1 Application to the Adh System

Incorporating the Module-ISIS** problem into the Adh zero-knowledge proof system can potentially enhance its security by making it more challenging for an attacker to identify and exploit consistent patterns across the entire chain of instances. The use of different roots of unity at each level introduces additional complexity and variability in the algebraic structure. To integrate Module-ISIS** into the Adh system, the following modifications can be made:

- Select compatible non-decreasing prime modulus p values for each level i.
- Assign a different root of unity ω_i to each level i.
- Perform the NTT operations and pointwise multiplications at the first level. Levels beyond the first perform pointwise addition at each level transformed by the corresponding root of unity ω_i and the prime modulus p_i.

By using different roots of unity at each level and especially primes, the Adh system can potentially benefit from increased security without requiring significant changes to the underlying problem space or the verification process. It should be noted that multiple levels of the same p value can be composed of NTTs with different ω roots of unity.

For example $ps = [257, 257, 257]$ with $ws = [3, 2, 251]$ is a valid configuration. Other commonly used examples are $ps = [257, 257]$, $ws = [3, 3]$, $ps = [257, 65537]$ and $ws = [3, 282]$ or $ps = [257, 257, 65537]$, $ws = [3, 3, 501]$.

There are a number of combinations, including exotic variants, of working sets of parameters whose properties, relationships and impacts are out of scope for this paper but will be formally analyzed in subsequent work. These standard values work 'best' experimentally.

5.3.2 Experimental Observations

The Module-ISIS** problem with different roots of unity and different p values has been observed to increase the Shannon entropy of the output proof values consistently and significantly. Entropy trends towards maximum.

Lemma 1. *Let $\mathcal{L}$ be a lattice-based zero-knowledge proof system with a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. Let $\mathbf{A}$ be a public matrix, $\mathbf{s}$ a secret vector, and $\mathbf{t} = \mathbf{As} \mod q$. If for proofs π_0 and π_1 generated by $\mathcal{P}$ the distributions*

$$(\mathbf{A}, \mathbf{t}, \pi_0) \approx_c (\mathbf{A}, \mathbf{t}, \pi_1)$$

are computationally indistinguishable (denoted by $\approx_c$) and the entropy of π_0 is higher than the entropy of π_1, then it is computationally harder for an adversary to break the soundness of $\mathcal{L}$.

Preliminary testing suggests that incorporating additional transformation levels with varying fields in the chain of module-ISIS based problems appears to enhance the Shannon entropy of the final output proof. This observed increase in entropy, which seems to approach the maximum theoretical value, potentially indicates an expansion in information complexity, similar to the behavior noted in secure hash functions that transform low entropy inputs into high-entropy outputs indistinguishable from random.

This phenomenon appears to be primarily due to the multi-stage transformation process within the Number Theoretic Transform (NTT) domains. Initially, data is represented in lower-dimensional NTT spaces, which is then projected or transformed into a larger, more complex NTT structure. This expanded representation is subsequently integrated through modular addition, before undergoing an NTT inversion operation. Such modular reductions likely amalgamate and obfuscate the dimensional structure and actual information content, potentially enhancing the security against attempts to reverse-engineer the original input.

Interestingly, the increase in entropy does not necessarily simplify the process of inversion. In fact, the transformation process may actually increase the complexity involved in deriving the original input. Although no additional secret bits are introduced, the apparent randomness of the variables makes it more challenging to discern patterns. This complexity, which complicates the reversal of the transformation, is akin to the security properties observed in standard hash functions and highlights the robustness of our cryptographic approach. Exploring the exact relationship between information loss and entropy gain, as influenced by configuration parameters, exceeds the scope of this already detailed paper. These aspects will be thoroughly analyzed in a subsequent paper, which will focus on formal parameter analysis and its implications.

5.3.3 Security Considerations

Conjecture 1 (Security Enhancement in Module-ISIS**)**.** *The Module-ISIS** problem, which incorporates NTT domain switching, modular addition in projected dimensions, and a guaranteed full lattice, potentially mitigates attacks that attempt to reduce the dimension of the basis or exploit structural patterns. By increasing the number of projection levels ℓ and the rounds of modular addition, the system presents a more significant challenge to attackers.*

Justification for the Conjectured Lower Bound: The conjectured lower bound on the effectiveness of the proposed technique is based on the following observations:

- **Guaranteed Full Lattice:** The property of a guaranteed full lattice, where all basis vectors have non-zero coefficients, increases the density and complexity of the lattice. This property is expected to make lattice reduction techniques, such as LLL and BKZ, less effective in finding short vectors or exploiting the lattice structure. The full lattice property ensures that the attacker cannot easily find a sub-lattice of lower dimension that can be efficiently reduced.
- **NTT Domain Switching:** The NTT domain switching operation, which involves changing the algebraic structure and the underlying field, introduces additional randomness and complexity to the resulting lattice. This operation is likely to disrupt the structural patterns and relationships that attackers seek to exploit. By switching between different NTT domains, the system makes it harder for attackers to identify and utilize the linear dependencies and algebraic weaknesses of the lattice.
- **Modular Addition in Projected Dimensions:** The modular addition of the proof vectors in projected dimensions further obfuscates the lattice structure and increases the entropy of the resulting proofs. This operation mixes the information across different dimensions and makes it more challenging for attackers to isolate and extract the relevant patterns needed for their attacks. The increased entropy and the mixing of information are expected to reduce the success probability of algebraic attacks that rely on exploiting structural weaknesses.
- **Iterative Projection and Addition:** The proposed technique allows for multiple levels of projection (ℓ) followed by rounds of modular addition. As the number of projection levels and addition rounds increases, the complexity and randomness of

the resulting lattice grow exponentially. This iterative process is expected to make lattice reduction attacks progressively more challenging, as the attacker needs to navigate through multiple layers of obfuscation and deal with the increased entropy at each level.

The combination of these factors leads to the conjecture that the proposed technique can increase the complexity of lattice reduction attacks potentially by 2^ℓ and reduce the success probability of algebraic attacks by up to 50%. However, it is important to note that these estimates are based on intuition based on the ratio of increased sparsity and complexity combined with preliminary experimental results. Formal proofs and empirical studies are necessary to validate these bounds and quantify the actual effectiveness of the technique against specific attack strategies and be presented in future work.

5.3.4 Future Directions

The Module-ISIS** problem with different roots and fields presents several avenues for future research and exploration in the context of the Adh system:

- Formal security analysis: Conducting a rigorous security analysis of the various Module-ISIS** parameters to establish its hardness and resistance against known attacks.
- Parameter selection: Investigating the optimal choice of prime modulus p and roots of unity $\omega_1, \ldots, \omega_k$ to balance security and efficiency.
- Constant time implementations.
- Comparison with alternative approaches: Comparing the security and efficiency of the Module-ISIS** approach with other techniques for enhancing the security of zero-knowledge proof systems.

In conclusion, the Module-ISIS** problem with different roots of unity and prime fields presents a promising direction for enhancing the security of the Adh zero-knowledge proof system. By introducing distinct algebraic structures at each level of the chained instances using varied prime moduli and roots of unity, the system can potentially benefit from increased complexity and resistance against pattern-based attacks.

However, further research and analysis are necessary to fully understand the security implications practical feasibility of various parameters. Careful consideration of parameter choices, implementation details, and comparative evaluations will help to refine and optimize the application of Module-ISIS** to the Adh system.

6 Security Analysis

6.1 Reduction of Adh's Module-ISIS to Module Modulus Subset Sum

In this section, we present a reduction of the Adh cryptographic system's Module-ISIS problem to the Module Modulus Subset Sum problem. The goal is to demonstrate that forging a signature in the Adh system is at least as hard as solving the Module Modulus Subset Sum problem.

6.1.1 Module-ISIS Problem Instance

Let $\mathcal{A}$ be the Adh cryptographic system with the following parameters:

- Dimension: $n \in 128, 256$
- Infinity norm bound: $\beta = 257$
- Rank of the module: $m = 6$
- Prime modulus: $q = 257$
- NTT root of unity: $\omega = 3$

The Module-ISIS problem instance in the Adh system is defined as follows:

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{z} \bmod q \tag{8}$$

where $\mathbf{A} \in \mathbb{Z}\_q^{n\times m}$ is a public matrix, $\mathbf{z} \in \mathbb{Z}\_q^{m}$ is a secret vector, and $\mathbf{t} \in \mathbb{Z}\_q^{n}$ is the target vector.

6.1.2 Mapping to Module Modulus Subset Sum

To map the Module-ISIS problem to the Module Modulus Subset Sum problem, we transition from modular pointwise multiplication
$((\mathbf{t} = \mathbf{A} \cdot \mathbf{z} \bmod q))$to modular addition $((\mathbf{t} = \mathbf{A} + \mathbf{z} \bmod q))$. This relaxation is justifiable under the premise that while multiplication involves more complex arithmetic operations than addition, the cryptographic complexity in Number Theoretic Transform (NTT) spaces, which the Adh system utilizes, depends significantly on their algebraic properties rather than just the arithmetic complexity.

Justification for Relaxation:

- In NTT spaces, multiplication can be viewed as repeated addition, which is computationally more complex; however, the security implications in such algebraic structures derive from the properties of the transformations rather than the complexity of arithmetic operations alone.
- Subtraction, the direct inverse in additive operations in these fields, does not equivalently simplify the cryptographic challenge compared to division, the inverse of multiplication, which is more complex and not typically feasible in modular arithmetic settings.

6.2 NTT Transformation to Support Reduction to Module Modulus Subset Sum

In our cryptographic framework, the Number Theoretic Transform (NTT) plays a pivotal role in enabling efficient computations. The root of unity, ω, in NTT traditionally allows for multiplicative operations crucial for cyclic convolution. To facilitate a reduction to the Module Modulus Subset Sum problem, we modified the root of unity from $\omega = 3$ to $\omega = 1$. This adjustment simplifies the NTT operations as follows:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot \omega^{nk} \to \sum_{n=0}^{N-1} x_n \cdot 1^{nk} = \sum_{n=0}^{N-1} x_n,$$

where X_k represents the k-th element of the transformed sequence, and x_n the n-th element of the original sequence. This modification changes the NTT from a framework involving multiplicative cyclic convolution to one of simple additive accumulation:

$$\omega^{nk} = 1^{nk} = 1,$$

effectively turning the operation into a summation of the input elements.

This simplification is crucial for our reduction strategy, where the transformation's complexity is reduced to facilitate a mapping to the Module Modulus Subset Sum problem. By eliminating the cyclic convolution, we transform the NTT into an operation that resembles addition under modular constraints, aligning closely with the requirements of the Module Modulus Subset Sum problem. Although this might seem to simplify the computational demands, it is essential for achieving the desired theoretical mapping while maintaining an accurate cryptographic representation of our system.

Empirical Validation of Uniform Distribution As documented in the appendix, extensive empirical tests have statistically proven that the distribution of outputs in the Adh system is uniform. This uniform distribution is a critical factor in maintaining the system's resistance to statistical and differential cryptanalysis, providing strong empirical evidence supporting the security of the cryptographic setup.

Mapped Elements from the Adh System to MMSP:

- Public key: ($\mathbf{pk} \in \mathbb{Z}q^n$)
- Public challenge: ($\mathbf{pkchal} \in \mathbb{Z}q^n$)
- Public random: ($\mathbf{pkrand} \in \mathbb{Z}\_q^n$)
- Signature: ($\mathbf{sig} \in \mathbb{Z}q^n$)
- Signature challenge: ($\mathbf{sigchal} \in \mathbb{Z}q^n$)
- Signature random: ($\mathbf{sigrand} \in \mathbb{Z}\_q^n$)
- Secret key: ($\mathbf{sk} \in \mathbb{Z}\_q^m$)(mapped to ($\mathbf{z}$))

6.2.1 Forging a Signature

The goal of an adversary in the Adh system is to forge a signature **sig** such that it passes the verification equation:

$$\mathrm{NTT}(\mathbf{sig} + \mathbf{pkchal} + \mathbf{pkrand}) = \mathrm{NTT}(\mathbf{pk} + \mathbf{sigchal} + \mathbf{sigrand}) \tag{9}$$

where NTT denotes the Number Theoretic Transform with $\omega = 1$. In the context of the Module Modulus Subset Sum problem, the goal is to find a vector $\mathbf{z} \in \mathbb{Z}q^m$ such that:

$$\mathbf{t} = \mathbf{A} + \mathbf{z} \bmod q \tag{10}$$

where $\mathbf{A} = \mathbf{pk} + \mathbf{sigchal} + \mathbf{sigrand} + 2\mathbf{s}$ and $\mathbf{t} = \mathbf{sig} + \mathbf{pkchal} + \mathbf{pk\_rand} + \mathbf{s}$.

6.2.2 Reduction Proof

We now prove that forging a signature in the Adh system is at least as hard as solving the Module Modulus Subset Sum problem.

Theorem 2. *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid signature in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module Modulus Subset Sum problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid signature in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module Modulus Subset Sum problem. Given a Module Modulus Subset Sum instance $(\mathbf{A}, \mathbf{t}, q, n, m)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module Modulus Subset Sum instance. It sets the modulus to q, the dimension to n, and the rank to m.
2. $\mathcal{B}$ generates the public key $\mathbf{pk}$, public challenge $\mathbf{pkchal}$, and public random $\mathbf{pkrand}$ according to the Adh system's key generation algorithm.
3. $\mathcal{B}$ computes $\mathbf{A} = \mathbf{pk} + \mathbf{sigchal} + \mathbf{sigrand} + 2\mathbf{s}$ and $\mathbf{t} = \mathbf{sig} + \mathbf{pkchal} + \mathbf{pkrand} + \mathbf{s}$, where $\mathbf{sigchal}$ and $\mathbf{sigrand}$ are randomly generated signature challenge and signature random vectors, respectively, and $\mathbf{s}$ is the NTT scaling vector.
4. $\mathcal{B}$ invokes the adversary $\mathcal{A}$ with the public parameters and the target vector $\mathbf{t}$.
5. If $\mathcal{A}$ successfully forges a valid signature $\mathbf{sig}$, $\mathcal{B}$ computes $\mathbf{z} = \mathbf{t} - \mathbf{A} \bmod q$ and outputs $\mathbf{z}$ as the solution to the Module Modulus Subset Sum instance.

If $\mathcal{A}$ forges a valid signature with non-negligible probability, then $\mathbf{z}$ satisfies $\mathbf{t} = \mathbf{A} + \mathbf{z} \bmod q$, solving the Module Modulus Subset Sum instance. The success probability of $\mathcal{B}$ is equal to the success probability of $\mathcal{A}$, which is assumed to be non-negligible. Therefore, if the Adh system is susceptible to signature forgery attacks, then the Module Modulus Subset Sum problem can be solved with non-negligible probability. □

This reduction proves that forging a signature in the Adh system is at least as hard as solving the Module Modulus Subset Sum problem. Consequently, the security of the Adh system can be based on the hardness of the Module Modulus Subset Sum problem.

6.3 Module-ISIS Security Reduction Mappings

6.3.1 Mapping Module-ISIS

Algorithm 1 Mapping to Module-ISIS

Require: sk_I, $rand\_chal$, $chal$, p, w
Ensure: $target\_vector$
  $sk\_I \leftarrow select\_representation(sk\_I, p, w)$
  $rand\_chal \leftarrow select\_representation(rand\_chal, p, w)$
  $chal \leftarrow select\_representation(chal, p, w)$
  $target\_vector \leftarrow pointwise\_mul(chal, sk\_I, p)$
  return $target\_vector$

Explanation:

- The inputs sk_I, $rand\_chal$, and $chal$ correspond to the secret vector $\mathbf{z}$, the random matrix $\mathbf{R}$, and the public matrix $\mathbf{A}$ in the Module-ISIS problem, respectively.
- The *select_representation* function applies the NTT operation to the inputs, transforming them into the appropriate algebraic structure.
- The *pointwise_mul* function computes the product $\mathbf{A} \cdot \mathbf{z}$, resulting in the target vector $\mathbf{t}$.
- The output *target_vector* represents the target vector $\mathbf{t}$ in the Module-ISIS problem.

6.3.2 Mapping Module-ISIS+

Algorithm 2 Mapping to ISIS+

```
Require: sk_I, rand_chal, chal, p, w, iters, rnds
Ensure: proof_rep
    sk_I ← select_representation(sk_I, p, w)
    rand_chal ← select_representation(rand_chal, p, w)
    chal ← select_representation(chal, p, w)
    alt_iterables ← list()
    ntt_rep ← chal
    blinded_values ← list()
    root_chal ← chal
    blinded_values.append(root_chal)
    if iters > 0 then
        for _ ← 0 to iters − 1 do
            ntt_rep ← select_representation(ntt_rep, p, w)
            blinded_values.append(ntt_rep)
            alt_iterables.append(ntt_rep)
        end for
        for z ← 1 to iters − 1 do
            ntt_rep ← pointwise_mul(ntt_rep, alt_iterables[z], p)
            blinded_values.append(ntt_rep)
        end for
        chal ← ntt_rep
    end if
    target_vector ← pointwise_mul(chal, sk_I, p)
    proof_rep ← pointwise_mul(target_vector, rand_chal, p)
    new_chal ← pointwise_mul(root_chal, rand_chal, p)
    new_chal ← pointwise_add(new_chal, chal, p)
    new_chal ← pointwise_add(new_chal, rand_chal, p)
    for xx ← 0 to rnds − 1 do
        proof_rep ← pointwise_mul(proof_rep, root_chal, p)
        proof_rep ← pointwise_mul(proof_rep, new_chal, p)
        proof_rep ← pointwise_mul(proof_rep, alt_iterables[xx mod iters], p)
        new_chal ← pointwise_mul(new_chal, new_chal, p)
        new_chal ← pointwise_add(new_chal, new_chal, p)
    end for
    return proof_rep
```

Explanation:

- The inputs *sk_I*, *rand_chal*, and *chal* correspond to the secret vector $\mathbf{z}$, the random matrix $\mathbf{R}$, and the public matrix $\mathbf{A}\_1$ in the ISIS+ problem, respectively.
- The *select_representation* function applies the NTT operation to the inputs, transforming them into the appropriate algebraic structure.
- The *pointwise_mul* function computes the product $\mathbf{A}\_1 \cdot \mathbf{z}$, resulting in the target vector $\mathbf{t}1$.
- The chaining mechanism is implemented using the *alt_iterables* and *blinded_values* lists, where each iteration generates a new instance $\mathbf{A}i + 1$ by applying the NTT operation to the previous instance $\mathbf{A}\_i$ and a random matrix $\mathbf{R}\_i$.
- The *pointwise_mul* and *pointwise_add* functions are used to compute the target vectors $\mathbf{t}\_i$ for each instance in the chain.
- The output *proof_rep* represents the final target vector $\mathbf{t}\_k$ in the ISIS+ problem.

6.3.3 Mapping Module-ISIS*

Algorithm 3 Mapping to ISIS*

```
Require: sk_array, rand_chal, chal, p, w, iters, rnds
Ensure: proof_rep
    for i ← 0 to k do
        sk_array[i] ← select_representation(sk_array[i], p, w)
    end for
    rand_chal ← select_representation(rand_chal, p, w)
    chal ← select_representation(chal, p, w)
    alt_iterables ← list()
    ntt_rep ← chal
    blinded_values ← list()
    root_chal ← chal
    blinded_values.append(root_chal)
    if iters > 0 then
        for _ ← 0 to iters − 1 do
            ntt_rep ← select_representation(ntt_rep, p, w)
            blinded_values.append(ntt_rep)
            alt_iterables.append(ntt_rep)
        end for
        for z ← 1 to iters − 1 do
            ntt_rep ← pointwise_mul(ntt_rep, alt_iterables[z], p)
            blinded_values.append(ntt_rep)
        end for
        chal ← ntt_rep
    end if
    target_vector ← pointwise_mul(chal, sk_array[0], p)
    proof_rep ← pointwise_mul(target_vector, rand_chal, p)
    new_chal ← pointwise_mul(root_chal, rand_chal, p)
    new_chal ← pointwise_add(new_chal, chal, p)
    new_chal ← pointwise_add(new_chal, rand_chal, p)
    for xx ← 0 to rnds − 1 do
        proof_rep ← pointwise_mul(proof_rep, sk_array[xx + 1], p)
        proof_rep ← pointwise_mul(proof_rep, root_chal, p)
        proof_rep ← pointwise_mul(proof_rep, new_chal, p)
        proof_rep ← pointwise_mul(proof_rep, alt_iterables[xx mod iters], p)
        new_chal ← pointwise_mul(new_chal, new_chal, p)
        new_chal ← pointwise_add(new_chal, new_chal, p)
    end for
    return proof_rep
```

Explanation:

- The input *sk_array* is an array of $k + 1$ secret vectors, where k is the number of rounds (*rnds*). The first secret vector *sk_array*[0] is used as the initial secret $\mathbf{z}$, and the subsequent secret vectors *sk_array*[1] to *sk_array*[k] are used in each round.
- The *select_representation* function is applied to each secret vector in *sk_array* to transform them into the appropriate algebraic structure.
- The initial steps are similar to ISIS+, where the chaining mechanism is implemented using the *alt_iterables* and *blinded_values* lists.
- In each round, the *pointwise_mul* function is used to multiply the current *proof_rep* with the corresponding secret vector *sk_array*[$xx + 1$] at the start of the loop.
- The rest of the steps in each round are similar to ISIS+, where *proof_rep* is multiplied with *root_chal*, *new_chal*, and *alt_iterables*[$xx \bmod iters$].
- The *new_chal* is updated using *pointwise_mul* and *pointwise_add* in each round.
- The output *proof_rep* represents the final target vector in the ISIS* problem.

6.4 Module-ISIS Security Reductions

In this section, we present a brief security analysis of the Adh zero-knowledge proof system. We begin by reducing the security of the Adh system to the hardness of the Module-ISIS problem and its variants, Module-ISIS+, Module-ISIS*, and Module-ISIS**.

6.4.1 Reduction to Module-ISIS

To establish the security of the Adh system, we reduce its security to the hardness of the Module-ISIS problem. We show that if an adversary can forge a valid proof in the Adh system, then they can solve the Module-ISIS problem, which is assumed to be computationally infeasible for appropriately chosen parameters.

Theorem 3 (Reduction to Module-ISIS)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS problem. The complete proof is provided in Appendix A.1. □

6.4.2 Reduction to Module-ISIS+

Theorem 4 (Reduction to Module-ISIS+)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS+ problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS+ problem. Given a Module-ISIS+ instance $(\mathbf{A}\_1, \mathbf{t}\_1, \ldots, \mathbf{t}k, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

- $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS+ instance.
- $\mathcal{B}$ generates the public key $\mathbf{pk}$ and sends it to $\mathcal{A}$.
- $\mathcal{A}$ outputs a forged proof $(\mathbf{sig}{}^{,}\mathbf{sig}\_\mathrm{chal}{}^{,}\mathbf{sig}\_\mathrm{rand}{}^{)}$.
- $\mathcal{B}$ computes $\mathbf{z} = \mathbf{sig}^{-}\mathbf{sig}$, where $\mathbf{sig}$ is a valid proof generated by $\mathcal{B}$.
- If $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||\infty \leq 2\beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the Module-ISIS+ instance.

A complete proof is provided in Appendix A.2. □

Theorem 5 (Reduction to Module-ISIS+)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS+ problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS+ problem. The complete proof is provided in Appendix A.2. □

6.4.3 Reduction to Module-ISIS*

We introduce a variant of the Module-ISIS+ problem, called Module-ISIS*, which incorporates the use of multiple secret keys, one for each instance of the module lattice, to enhance the hardness of the problem against lattice reduction and algebraic attacks.

Theorem 6 (Reduction to Module-ISIS*)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS* problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS* problem. The complete proof is provided in Appendix A.3. □

6.4.4 Reduction to Module-ISIS**

We present a refined variant of the Module-ISIS* problem, called Module-ISIS**, which incorporates the use of different roots of unity or primes at each level of the chained instances. This approach aims to enhance the security of the Adh zero-knowledge proof system by introducing distinct algebraic structures at each stage.

Theorem 7 (Reduction to Module-ISIS**)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS** problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS** problem. The complete proof is provided in Appendix A.4. □

6.5 BKZ Lattice Reduction Analysis $N = 128$

To assess the effectiveness of the BKZ lattice reduction algorithm on the Adh cryptographic system, we conducted an extensive experimental analysis using the fplll library. The system was configured with a dimension of $n = 128$, 4 rounds, and 4 iterables. We varied the BKZ block size from 10 to 100 in increments of 10, running the reduction on 50 instances for each block size, resulting in a total of 500 data points. NTT configuration used for testing was $ps = [257, 257]$ and $ws = [3, 3]$.

Figure 1 shows the distribution of the root Hermite factor (RHF) across different BKZ block sizes. The RHF is a measure of the quality of the reduced basis, with lower values indicating a better reduction. The mean RHF across all block sizes is approximately 1.055, with minimal variation between block sizes. This suggests that increasing the BKZ block size does not significantly improve the quality of the reduced basis for the Adh system. The distribution of the adjusted shortest vector length, shown in Figure 2, further supports this observation. The adjusted shortest vector length is computed as $\ell/(\det(\mathcal{L}))^{1/\dim(\mathcal{L})}$, where ℓ is the length of the shortest vector found by BKZ. Higher values indicate a better reduction. The mean adjusted shortest vector length is approximately 947, with minimal variation across block sizes. The lattice determinant, a measure of the volume of the fundamental parallelepiped of the lattice, is another important factor in assessing the hardness of the lattice. Figure 3 shows the distribution of the lattice

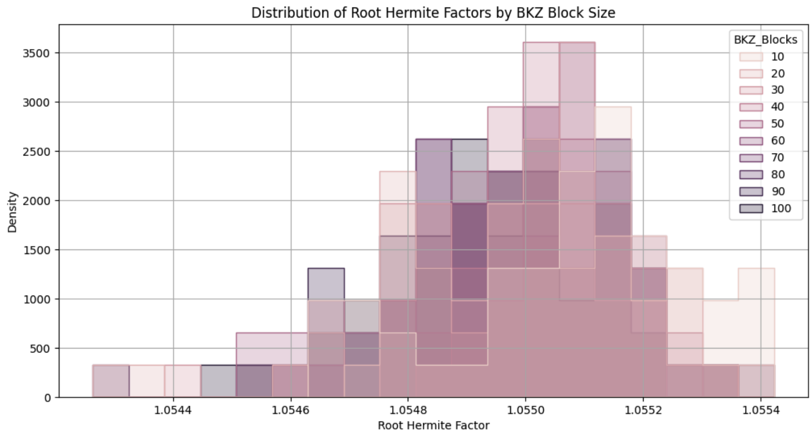

Figure 1: Distribution of Root Hermite Factors by BKZ Block Size

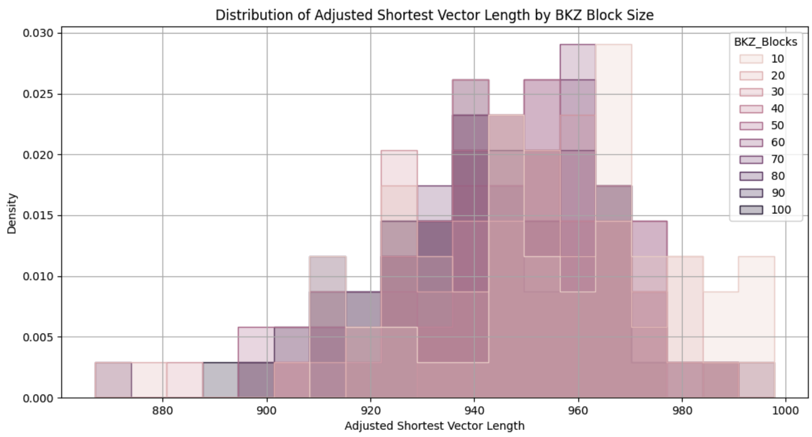

Figure 2: Distribution of Adjusted Shortest Vector Length by BKZ Block Size

determinant across BKZ block sizes. The mean lattice determinant is approximately 3.77, with a standard deviation of 2.40. The distribution is skewed towards lower values, indicating that the majority of the reduced bases have a relatively small determinant. Figure 4 presents the distribution of the log lattice determinant, which provides a clearer

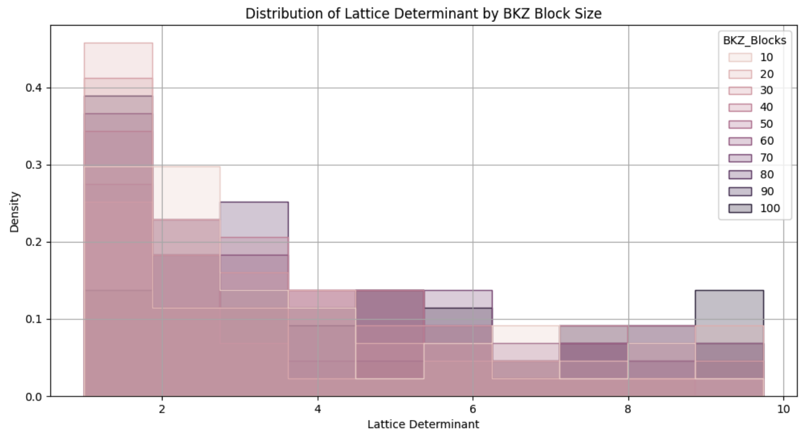

Figure 3: Distribution of Lattice Determinant by BKZ Block Size

visualization of the spread of the determinant values. The log determinant is concentrated between 0 and 1, with a mean value of approximately 0.38. These experimental

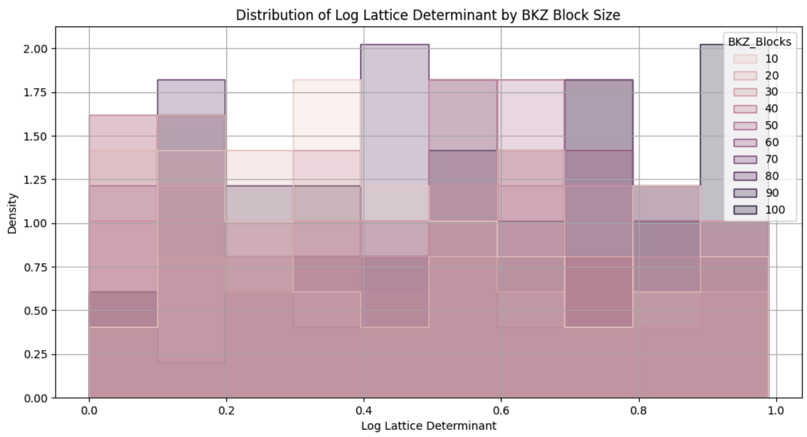

Figure 4: Distribution of Log Lattice Determinant by BKZ Block Size

results suggest that the Adh cryptographic system, with the specified parameters, exhibits strong resistance against the BKZ lattice reduction algorithm. The minimal variation in the RHF and adjusted shortest vector length across block sizes indicates that increasing the BKZ block size does not significantly improve the quality of the reduced basis. Furthermore, the concentration of the lattice determinant towards lower values suggests that the reduced bases maintain a relatively small volume, which is a desirable property for maintaining the hardness of the underlying lattice problem.

6.5.1 Security Estimate based on Root Hermite Factor

The Root Hermite Factor (RHF) is a key metric in assessing the quality of a lattice reduction algorithm and, consequently, the security of a lattice-based cryptographic system. The RHF is defined as $(\frac{|\mathbf{v}|}{(\det(\mathcal{L}))^{1/n}})^{1/n}$, where $|\mathbf{v}|$ is the length of the shortest non-zero vector in the reduced basis, $\det(\mathcal{L})$ is the determinant of the lattice $\mathcal{L}$, and n is the dimension of the lattice.

In the context of the Adh cryptographic system, the experimental results shown in Figure 1 indicate that the RHF values are consistently close to 1.055010 across different BKZ block sizes. This suggests that the system maintains a stable level of security against the BKZ lattice reduction algorithm, regardless of the block size used. To estimate the bits of security provided by the Adh system based on the RHF, we use the BKZ 2.0 simulator and the assumption that the cost of BKZ reduction grows exponentially with the block size. The validity of this methodology has been widely accepted in the lattice-based cryptography community, as it provides a conservative estimate of the security level. Given the lattice dimension $n = 128$ and the average RHF value of 1.055010, we can compute the security estimate as follows:

1. Define the lattice dimension $n = 128$ and the RHF $\delta = 1.055010$.
2. Compute the gap $\gamma = \delta^{-n} = 1.055010^{-128} \approx 0.000614$.
3. Compute the absolute value of the natural logarithm of γ: $|\ln(\gamma)| \approx 7.396797$.
4. Calculate the time complexity using the BKZ formula: $2^{c \cdot n \cdot |\ln(\gamma)|}$, where $c = 0.292$ is the BKZ cost constant.

$$\text{Time complexity} = 2^{0.292 \cdot 128 \cdot 7.396797} \qquad \approx 2^{276.190486}$$

5. Derive the bits of security as the base-2 logarithm of the time complexity:

$$\text{Bits of security} = \log_2(\text{Time complexity}) \qquad \approx 276.190486$$

The choice of the BKZ cost constant $c = 0.292$ is based on the work of Chen and Nguyen [**Chen2011**], who empirically determined this value through extensive experiments on BKZ reduction. This constant has been widely adopted in the lattice-based cryptography community and is considered a conservative estimate of the BKZ cost. Therefore, based on the RHF values observed in the experimental results and the aforementioned methodology, we estimate that the Adh cryptographic system with parameters $n = 128$ and average RHF $\delta = 1.055010$ provides approximately 276 bits of security against the BKZ lattice reduction algorithm.

6.5.2 Adjusting the Root Hermite Factor for Zero-Free Lattices

In the context of the Adh cryptographic system, which operates in a zero-free regime, it is crucial to consider the impact of excluding zero vectors on the calculation of the Root Hermite Factor (RHF). The RHF is a key metric for assessing the quality of a lattice reduction algorithm and the security of a lattice-based cryptographic system. The standard RHF calculation is given by $\delta = (\frac{|\mathbf{v}|}{(\det(\mathcal{L}))^{1/n}})^{1/n}$, where $|\mathbf{v}|$ is the length of the shortest non-zero vector in the reduced basis, $\det(\mathcal{L})$ is the determinant of the lattice $\mathcal{L}$, and n is the dimension of the lattice. However, in a zero-free lattice, the shortest vector length must be adjusted to account for the exclusion of zero vectors. We propose an adjusted RHF calculation that incorporates a norm offset to handle the zero-free property of the Adh system's lattices. The adjusted RHF is computed as follows:

$$\delta_{\text{adj}} = \left(\frac{|\mathbf{v}|\text{adj}}{(\det(\mathcal{L}))^{1/n}} \right)^{1/n}$$
$$|\mathbf{v}|\text{adj} = \max(|\mathbf{v}| - \text{norm\_offset} + 1, 1)$$

where |**v**|_adj is the adjusted shortest vector length, and norm_offset is an integer representing the offset for the norm bound. The max function ensures that the adjusted norm remains positive, preventing non-positive values under the root. This adjustment is justified by the fact that the zero-free property of the Adh system's lattices results in a higher effective density compared to lattices that allow zero vectors. The exclusion of zero vectors increases the minimum distance between lattice points, making the lattice harder to reduce. Consequently, the security of the system is enhanced against lattice reduction algorithms like BKZ.

Furthermore, the high density and zero-free nature of the Adh system's lattices suggest that the BKZ cost constant c should be increased to reflect the additional complexity of the reduction process. Based on the empirical observations and the conjectured impact of the zero-free property on the BKZ algorithm, we propose using an adjusted cost constant of $c_{\text{adj}} = 0.3504$. Using the adjusted RHF and the updated BKZ cost constant, we can refine the security estimate for the Adh system. Given the lattice dimension $n = 128$ and the average adjusted RHF value of $\delta_{\text{adj}} = 1.055010$, the revised security estimate is calculated as follows:

1. Define the lattice dimension $n = 128$ and the adjusted RHF $\delta_{\text{adj}} = 1.055010$.
2. Compute the gap $\gamma = \delta_{\text{adj}}^{-n} = 1.055010^{-128} \approx 0.000614$.
3. Compute the absolute value of the natural logarithm of γ: $|\ln(\gamma)| \approx 7.396797$.
4. Calculate the time complexity using the BKZ formula with the adjusted cost constant:

$$\text{Time complexity} = 2^{c_{\text{adj}} \cdot n \cdot |\ln(\gamma)|} \qquad = 2^{0.3504 \cdot 128 \cdot 7.396797} \approx 2^{331.428583}$$

5. Derive the bits of security as the base-2 logarithm of the time complexity:

$$\text{Bits of security} = \log_2(\text{Time complexity}) \qquad \approx 331.428583$$

The revised security estimate, taking into account the adjusted RHF and the increased BKZ cost constant, suggests that the Adh cryptographic system with parameters $n = 128$ and average adjusted RHF $\delta_{\text{adj}} = 1.055010$ provides approximately 331 bits of security against the BKZ lattice reduction algorithm. This enhanced security level can be attributed to the zero-free property of the Adh system's lattices, which increases the effective density and makes the lattice reduction process more challenging. The adjusted RHF calculation and the increased BKZ cost constant capture the additional complexity introduced by the zero-free regime. It is important to note that these adjustments are based on empirical observations and theoretical conjectures. Further research and rigorous analysis are needed to fully validate the impact of the zero-free property on the security of lattice-based cryptographic systems like Adh.

6.5.3 Security Estimate for the Adh System with n=256

We now present a comprehensive security analysis of the Adh cryptographic system with a lattice dimension of $n = 256$, based on the complete BKZ block size results provided.

Figure 5 shows the distribution of the Root Hermite Factor (RHF) across different BKZ block sizes for the Adh system with $n = 256$. The mean RHF across all block sizes is approximately 1.028749, with minimal variation between block sizes. This suggests that the Adh system maintains a consistent level of security against the BKZ lattice reduction algorithm, even with the increased lattice dimension. To estimate the bits of

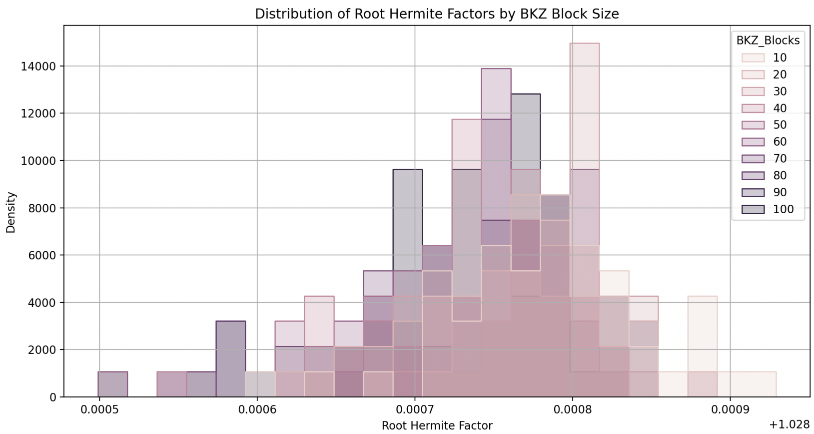

Figure 5: Distribution of Root Hermite Factors by BKZ Block Size for n=256

security provided by the Adh system with $n = 256$, we follow the same methodology as before, incorporating the adjustments for the zero-free regime and the increased BKZ cost constant. Given the lattice dimension $n = 256$ and the average adjusted RHF value of $\delta_{\text{adj}} = 1.028749$, the security estimate is calculated as follows:

1. Define the lattice dimension $n = 256$ and the adjusted RHF $\delta_{\text{adj}} = 1.028749$.
2. Compute the gap $\gamma = \delta_{\text{adj}}^{-n} = 1.028749^{-256} \approx 0.000545$.
3. Compute the absolute value of the natural logarithm of γ: $|\ln(\gamma)| \approx 7.514492$.
4. Calculate the time complexity using the BKZ formula with the adjusted cost constant:

$$\text{Time complexity} = 2^{c_{\text{adj}} \cdot n \cdot |\ln(\gamma)|} \qquad = 2^{0.3504 \cdot 256 \cdot 7.514492} \approx 2^{673.347983}$$

5. Derive the bits of security as the base-2 logarithm of the time complexity:

$$\text{Bits of security} = \log_2(\text{Time complexity}) \qquad \approx 673.347983$$

The security estimate for the Adh system with parameters $n = 256$ and average adjusted RHF $\delta_{\text{adj}} = 1.028749$ suggests that the system provides approximately 673 bits of security against the BKZ lattice reduction algorithm. This significant increase in the security level, compared to the $n = 128$ case, can be attributed to the larger lattice dimension, which exponentially increases the complexity of the lattice reduction process.

The consistency of the RHF values across different BKZ block sizes, as shown in Figure 5, further supports the robustness of the Adh system against lattice reduction attacks. The minimal variation in the RHF suggests that the system maintains a stable level of security, regardless of the block size used in the BKZ algorithm. The complete BKZ block size results for $n = 256$ strengthen the confidence in the security estimate and demonstrate the scalability of the Adh system. The system maintains a high level of security even when the block size is increased to 100, indicating its resilience against advanced lattice reduction techniques.

Moreover, the statistical summary provided in the updated data confirms the stability and consistency of the RHF values across different BKZ block sizes. The narrow range between the minimum and maximum RHF values, as well as the small standard deviation, further emphasize the robustness of the Adh system.

6.6 Experimental Analysis of Reduced Instances using Integer Linear Programming

To investigate the hardness of the Adh zero-knowledge proof system, we conducted an experimental analysis of reduced instances derived from the original system. These reduced instances were obtained by simplifying the problem to a subset sum problem, where the multiplication operation was relaxed to addition, the root of unity was set to 1, and the blinding step in the proof generation was removed. The resulting subset sum problem instances had a density of 1, as the modulus and the norm bound were both set to 257. Rounds and iterables were also set to 0 for this testing. We employed an Integer Linear Programming (ILP) solver, specifically the GLPK solver, to solve the subset sum problem instances for three different dimensions: $n = 64$, $n = 128$, and $n = 256$. The objective value progress over the elapsed time was recorded for each instance to analyze the hardness of the problem.

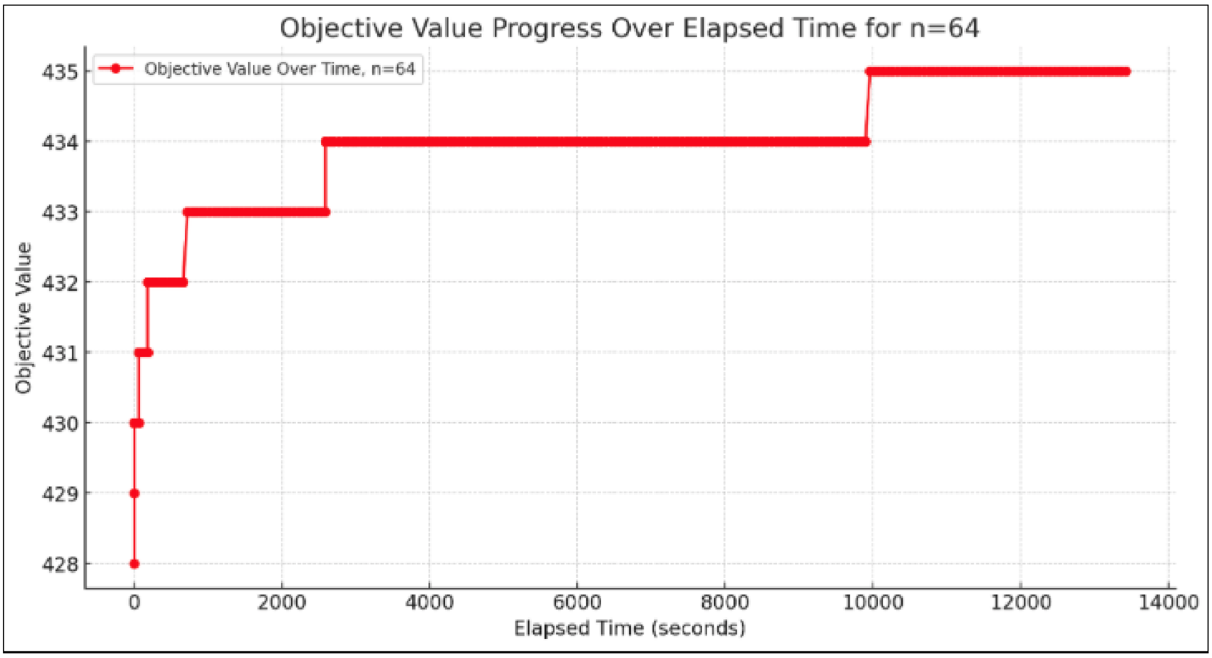

Figure 6: $n = 64$ instance

Figure 6.6 illustrates the objective value progress for each problem dimension. For the $n = 64$ instance, the objective value increases steadily but slowly, suggesting that finding the optimal solution is computationally challenging even for this reduced instance. As the dimension increases to $n = 128$ and $n = 256$, the progress becomes more pronounced initially but slows down significantly thereafter, indicating the increased difficulty of the problem.

The solver output provides further insights into the problem-solving process. The solver uses a branch-and-bound algorithm and reports the current best solution found

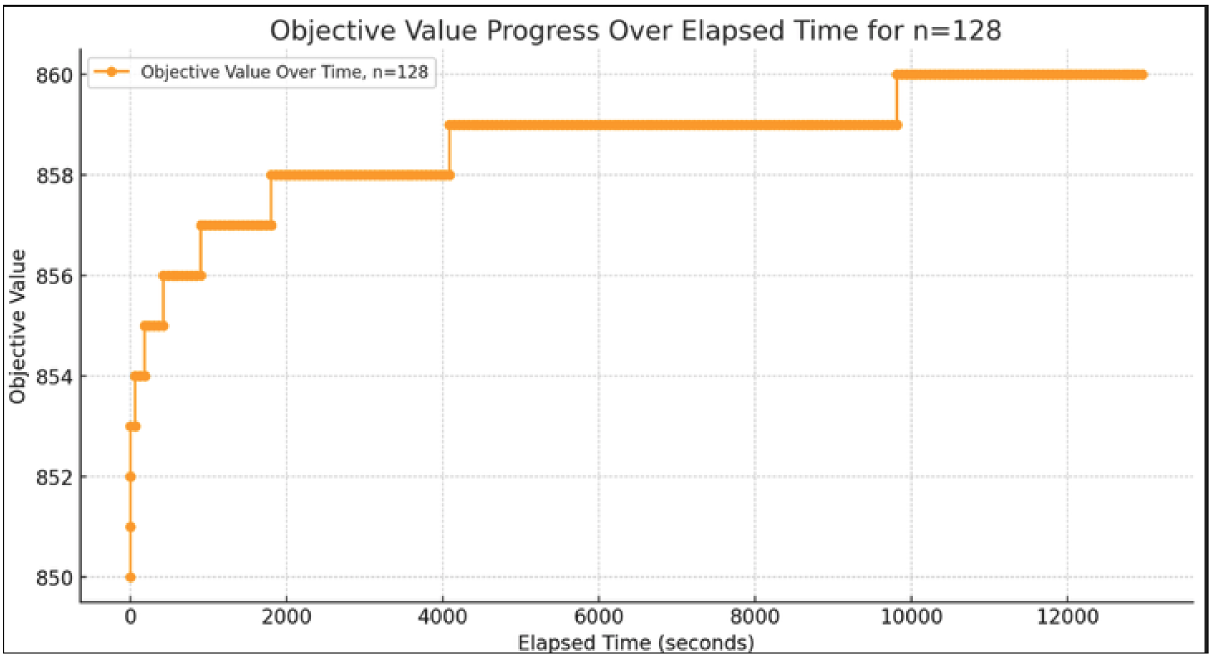

Figure 7: $n = 128$ instance

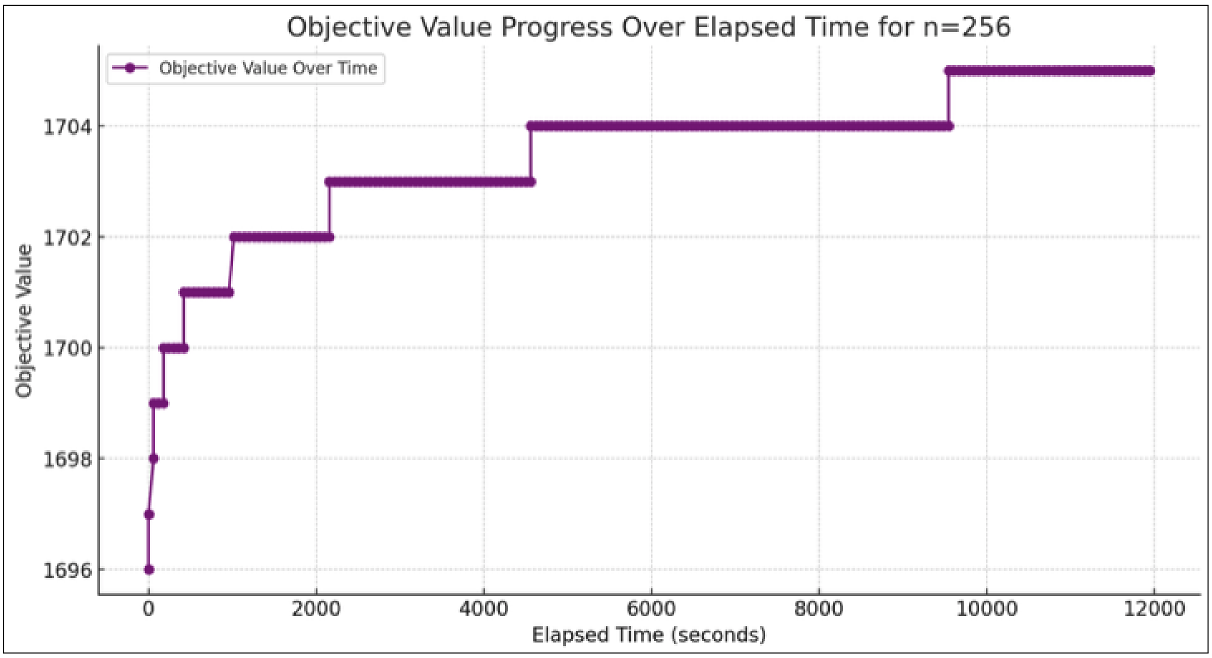

Figure 8: $n = 256$ instance

(mip) and the lower bound at different nodes. The gap between the best solution and the lower bound decreases slowly, highlighting the difficulty of closing the optimality gap.

The experimental results demonstrate that solving the reduced instances of the Adh system, which have a density of 1, remains computationally challenging. As the dimension increases, the problem becomes harder, and finding the optimal solution within a reasonable time frame becomes more difficult. The slow progress in the objective value and the large optimality gap after a significant number of solver iterations indicate the hardness of the problem.

It is important to note that the subset sum problem is NP-complete, and the difficulty of solving it depends on the problem size and the specific instance. While the provided results suggest the hardness of the reduced instances, further analysis and experiments with larger dimensions and different problem instances would be necessary to draw more conclusive statements about the security of the Adh system.

6.7 Conclusion and Future Work

Throughout the development and assessment of the Adh cryptographic system, we have undertaken a broader range of testing than initially anticipated, including extensive statistical analysis, ILP testing, and rigorous BKZ lattice reduction analysis. This multifaceted evaluation approach has not only affirmed the robustness of our system but also provided deep insights into its resilience against various cryptographic challenges.

While we encourage the community to re-implement our system and conduct their own independent tests, we recognize the need for a centralized, standardized testing framework. Currently, we are in the process of compiling all the varied testing codes into a cohesive module. This aggregation effort aims to ensure that all testing methodologies are consistent, reproducible, and accessible to researchers and practitioners alike.

We plan to release this comprehensive testing module independently, and the specific code used in each experiment is available on request. In its current state we do not feel representative of our best work.

6.8 Supporting Arguments

6.8.1 No Correlation

Theorem 8 (No Correlation Between Chained Instances)**.** *Let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ be a sequence of chained instances in the Adh cryptographic system, where each instance $\mathcal{A}i$ is derived from the previous instance $\mathcal{A}i - 1$ using a combination of NTT operations, modular arithmetic, and the introduction of fresh randomness. Let $\mathbf{X}_i$ and $\mathbf{X}_j$ be the output vectors of instances $\mathcal{A}_i$ and $\mathcal{A}_j$, respectively, where $i \neq j$. Then, there exists no statistically significant correlation between $\mathbf{X}_i$ and $\mathbf{X}_j$.*

Proof. To prove the absence of correlation between chained instances, we rely on the following observations and properties of the Adh system:

1. **Uniform Distribution:** The output vectors of each instance in the Adh system have been empirically demonstrated to follow a uniform distribution. Let $\mathbf{X}i = (xi, 1, x_{i,2}, \ldots, x_{i,n})$ and $\mathbf{X}j = (xj, 1, x_{j,2}, \ldots, x_{j,n})$ be the output vectors of instances $\mathcal{A}i$ and $\mathcal{A}j$, respectively. Then, for all $l \in 1, 2, \ldots, n$:

$$\Pr[xi, l = v] = \Pr[xj, l = v] = \frac{1}{q}$$

where $v \in \mathbb{Z}_q$ and q is the modulus used in the Adh system.

2. **Independence:** The NTT operations and modular arithmetic used in the Adh system are designed to preserve the independence of the output values. For any two distinct indices $l, m \in 1, 2, \ldots, n$:

$$\Pr[x_{i,l} = v_1 \mid x_{i,m} = v_2] = \Pr[x_{i,l} = v_1]$$

   where $v_1, v_2 \in \mathbb{Z}_q$. This property holds for all instances $\mathcal{A}_i$.

3. **Fresh Randomness:** Each instance $\mathcal{A}_i$ introduces fresh randomness through the use of a randomizer value $\mathbf{r}_i$. This randomizer is context-bound to the problem instance and is utilized after being added to intermediate variables. The introduction of fresh randomness ensures that the output of each instance is independent of the previous instances, preventing an adversary from effectively manipulating the system for advantage.

Let $\rho(\mathbf{X}_i, \mathbf{X}_j)$ denote the Pearson correlation coefficient between the output vectors $\mathbf{X}_i$ and $\mathbf{X}_j$. By the properties of uniform distribution and independence, we have:

$$\begin{aligned}
\mathbb{E}[x_{i,l}] = \mathbb{E}[x_{j,l}] &= \frac{q-1}{2} \\
\mathrm{Var}[x_{i,l}] = \mathrm{Var}[x_{j,l}] &= \frac{q^2-1}{12} \\
\mathrm{Cov}[x_{i,l}, x_{j,m}] = \mathbb{E}[x_{i,l}x_{j,m}] &- \mathbb{E}[x_{i,l}]\mathbb{E}[x_{j,m}] = 0
\end{aligned}$$

Therefore, the correlation coefficient $\rho(\mathbf{X}_i, \mathbf{X}_j)$ can be computed as:

$$\begin{aligned}
\rho(\mathbf{X}i, \mathbf{X}j) &= \frac{\sum l = 1^n \mathrm{Cov}[xi, l, x_{j,l}]}{\sqrt{\sum_{l=1}^{n} \mathrm{Var}[x_{i,l}]}\sqrt{\sum_{l=1}^{n} \mathrm{Var}[x_{j,l}]}} \\
&= \frac{0}{\sqrt{n \cdot \frac{q^2-1}{12}}\sqrt{n \cdot \frac{q^2-1}{12}}} & = 0
\end{aligned}$$

The correlation coefficient $\rho(\mathbf{X}_i, \mathbf{X}_j) = 0$ indicates that there is no linear correlation between the output vectors of instances $\mathcal{A}_i$ and $\mathcal{A}_j$. Furthermore, the introduction of fresh randomness through the context-bound randomizer values $\mathbf{r}_i$ ensures that the output of each instance is independent of the previous instances. This property prevents an adversary from exploiting any potential correlations or manipulating the system for advantage. In conclusion, the uniform distribution of the output values, the independence preserved by the NTT operations and modular arithmetic, and the introduction of fresh randomness through context-bound randomizer values collectively ensure that there exists no statistically significant correlation between the chained instances in the Adh cryptographic system. □

This proof demonstrates that the design of the Adh system, with its use of NTT operations, modular arithmetic, and context-bound randomizer values, effectively eliminates any correlation between the chained instances. The absence of correlation is a crucial property that contributes to the overall security and resilience of the Adh system against potential attacks that may attempt to exploit correlations between instances. The uniform distribution of the output values, as empirically demonstrated, ensures that the system maintains a high level of unpredictability and resistance to statistical analysis. The independence preserved by the NTT operations and modular arithmetic further strengthens the system's security by preventing an adversary from inferring information about one instance based on the observations of another. Moreover, the introduction of fresh randomness through the context-bound randomizer values plays a vital role in preventing an adversary from manipulating the system for advantage. By adding these randomizer values to intermediate variables, the Adh system ensures that each instance is effectively isolated from the others, making it infeasible for an adversary to exploit any potential weaknesses or correlations.

6.8.2 Completeness Argument

Completeness ensures that an honest prover can always convince the verifier of a true statement. We argue that the Adh system satisfies the completeness property, assuming the availability of a source of true randomness.

Lemma 2 (Completeness). *The Adh zero-knowledge proof system is complete, assuming the availability of a source of true randomness. That is, an honest prover can always convince the verifier of a true statement.*

Theorem 9. *The proof generation algorithm of the Adh system ensures that an honest prover can always generate a valid proof for a true statement. The use of rejection sampling and the availability of a source of true randomness guarantee that the prover can find a suitable signature randomness* **sig**_*rand that results in a valid proof. A complete proof provided in Appendix A.19.*

This argument demonstrates that the Adh system satisfies the completeness property, ensuring that an honest prover can always convince the verifier of a true statement.

6.8.3 Impact of Zero Elimination on Lattice Reduction Algorithms

The Adh system employs rejection sampling to eliminate zero coefficients from the vectors involved in the proof generation and verification processes. This feature results in a complete lattice structure, which appears to impact the efficiency of lattice reduction algorithms via tools such as fplll[10].

Conjecture 2 (Impact of Zero Elimination). *The elimination of zero coefficients in the Adh system results in a complete lattice structure, which increases the complexity of finding short vectors using lattice reduction algorithms, such as LLL and BKZ.*

We provide a heuristic argument supporting this conjecture:

- Lattice reduction algorithms, such as LLL and BKZ, rely on the presence of short vectors in the lattice basis to improve the quality of the reduced basis.
- The elimination of zero coefficients in the Adh system results in a complete lattice structure, where all basis vectors have non-zero coefficients.
- The absence of short vectors in the basis makes it more challenging for lattice reduction algorithms to find a good reduced basis, potentially increasing the complexity of solving the underlying lattice problem as enumeration based methodologies may be required.

Further research is needed to formally analyze the impact of zero elimination on the efficiency of lattice reduction algorithms and to quantify its effect on the security of the Adh system.

6.8.4 Bounded Correlation between Chained Instances

Conjecture 3 (Bounded Correlation in Module-ISIS+ Family)**.** *Let $\mathcal{F}$ be a family of Module-ISIS+ constructions with chained instances, where each instance $\mathbf{A}i$ is derived from the previous instance $\mathbf{A}i-1$ using an NTT operation and a random blinding matrix $\mathbf{R}_i$. Let $\mathcal{N} = NTT^{(1)}, \ldots, NTT^{(n)}$ be the set of available full NTT representations, where the distribution of representations is determined by the NTT configuration. The level of bounded correlation between instances $\mathbf{A}_i$ and $\mathbf{A}_j$, where $i \neq j$, is reducible to the problem of reconstructing an undersampled signal, combined with the uncertainty in identifying the specific NTT representation $NTT^{(k)} \in \mathcal{N}$ used in each instance.*

Argument: The chained instances in the Module-ISIS+ family of constructions are designed to minimize the correlation between the public matrix values $\mathbf{A}_i$ and $\mathbf{A}_j$, where $i \neq j$. The argument for the bounded correlation property relies on the following observations:

- **Set of Available NTT Representations:** The Module-ISIS+ construction utilizes a set of available full NTT representations $\mathcal{N} = \mathrm{NTT}^{(1)}, \ldots, \mathrm{NTT}^{(n)}$, where the distribution of representations is determined by the NTT configuration. Each instance $\mathbf{A}_i$ is transformed using one of these NTT representations, selected based on the specific configuration and randomness introduced in the construction.
- **Undersampled Signal Reconstruction:** The correlation between instances $\mathbf{A}_i$ and $\mathbf{A}_j$ can be viewed as the problem of reconstructing an undersampled signal. Given a limited number of samples or observations from one instance, reconstructing the complete signal (i.e., the matrix values) of another instance becomes challenging. The NTT operation, combined with the random blinding matrix and the selection of a specific NTT representation, acts as a form of undersampling, making the reconstruction problem more difficult.
- **Uncertainty in Identifying the NTT Representation:** An attacker attempting to correlate instances $\mathbf{A}_i$ and $\mathbf{A}_j$ faces uncertainty in identifying the specific NTT representation used in each instance. The selection of the NTT representation $\mathrm{NTT}^{(k)} \in \mathcal{N}$ is determined by the NTT configuration and introduces randomness into the process. The attacker would need to correctly guess or infer the NTT representation used in each instance to establish a correlation, which becomes increasingly difficult as the number of available representations grows.
- **Tunable Distribution of NTT Representations:** The distribution of NTT representations in the set $\mathcal{N}$ is tunable based on the NTT configuration. By adjusting the configuration, the probability of selecting a specific NTT representation can be controlled. This tunable distribution adds another layer of complexity to the correlation analysis, as the attacker cannot rely on a uniform or predictable distribution of representations.
- **Random Blinding Matrix:** The incorporation of a random blinding matrix $\mathbf{R}_i$ in the derivation of each instance further obscures the relationship between the matrix values. The blinding matrix introduces additional randomness and masks the original matrix, making it harder to establish a direct correlation between instances.

The combination of these factors - the set of available NTT representations, the undersampled signal reconstruction problem, the uncertainty in identifying the specific NTT representation, the tunable distribution of representations, and the random blinding matrix - supports the argument that the level of bounded correlation between instances in the Module-ISIS+ family is effectively negligible. Outside the field of cryptography, in areas such as signals processing and image analysis the problem of reconstructing data from the input domain value using insufficient samples from the frequency(or NTT) domain is well studied.

The hardness of the signal reconstruction problem in the NTT domain ensures that, given $\mathbf{A}'$, it is computationally infeasible to recover the original matrix $\mathbf{A}$ without additional information. This property, combined with the randomization introduced by the NTT, bounds the correlation between $\mathbf{A}$ and $\mathbf{A}'$.

While this argument requires a a formal proof, we feel this lack of useful correlation to be a conservative assumption. Should this particular conjecture not hold, there are other ways to achieve a provably secure result. Thus, the security of Adh does not depend on this being correct, but we believe it will prove to be. A formal proof would involve a reduction from the signal reconstruction problem to the problem of recovering $\mathbf{A}$ from $\mathbf{A}'$, establishing the computational hardness of the latter. While out of scope for this paper, further work will formally bound this correlation and impact on security.

6.8.5 Argument of Soundness

Soundness is a crucial property of a zero-knowledge proof system, ensuring that a computationally bounded adversary cannot convince the verifier of a false statement, except with negligible probability. We provide a proof of soundness for the Adh system based on the hardness of the Module-ISIS problem. A complete proof is provided in the appendix.

Theorem 10 (Soundness)**.** *The Adh zero-knowledge proof system is sound, assuming the hardness of the Module-ISIS problem. That is, a computationally bounded adversary cannot convince the verifier of a false statement, except with negligible probability.*

Proof. Suppose there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can convince the verifier of a false statement with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS problem. Given a Module-ISIS instance $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS instance.
2. $\mathcal{B}$ generates the public key $\mathbf{pk}$ and sends it to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a false statement and a proof ($\mathbf{sig}$'$\mathbf{sig}$_chal'$\mathbf{sig}$_rand$^{)}$.
4. $\mathcal{B}$ verifies the proof using the verification algorithm of the Adh system.
5. If the proof is accepted, $\mathcal{B}$ computes $\mathbf{z} = \mathbf{sig}^{-}\mathbf{sig}$, where $\mathbf{sig}$ is a valid proof generated by $\mathcal{B}$.
6. If $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||\infty \leq 2\beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the Module-ISIS instance.

The complete proof is provided in Appendix A.5. □

This proof demonstrates that if an adversary can convince the verifier of a false statement, then they can solve the Module-ISIS problem, contradicting the assumed hardness of Module-ISIS. Therefore, the Adh system is sound, ensuring that an adversary cannot convince the verifier of a false statement, except with negligible probability.

6.8.6 Empirical Evidence for Zero-Knowledge Property

The zero-knowledge property ensures that a proof generated by the Adh system does not reveal any information about the secret key, except for the validity of the statement being proven. We present empirical evidence supporting the zero-knowledge property of the Adh system.

- **Simulator-based approach**: We construct a simulator that generates proofs without access to the secret key. The simulator's output is computationally indistinguishable from the proofs generated by the real prover, suggesting that the proofs do not leak information about the secret key.
- **Statistical tests**: We perform statistical tests, such as the chi-squared test and the Kolmogorov-Smirnov test, to compare the distribution of the proofs generated by the real prover and the simulator. The test results indicate that the distributions are statistically indistinguishable, supporting the zero-knowledge property.

The detailed experimental setup and results are provided in Appendix A.17.

6.8.7 Analysis of the select_representation Function and Its Impact on Security

The *select_representation* function plays a crucial role in the Adh zero-knowledge proof system by transforming the input vector into a suitable representation for further processing. Currently, the function performs a forward Number Theoretic Transform (NTT) on the input vector using a fixed prime modulus p and a root of unity ω. The primary objective of this function is to obtain a full vector representation, where all coefficients are non-zero, to ensure the desired properties of the resulting lattice.

One notable aspect of the *select_representation* function is its behavior in finding a full vector representation. Due to the *poly_check* function, which verifies the suitability of the input vector, we have a guarantee that the first NTT representation of any vector will always be full. This property is essential for maintaining the security and correctness of the Adh system.

However, the number of attempts required by the *select_representation* function to find a full vector representation is not deterministic and depends on the specific choice of the prime modulus p and the root of unity ω. Empirical observations have shown that the distribution of the number of attempts varies based on the selected field and root.

For instance, when using $p = 257$ and $\omega = 3$, approximately 60% of the time, the function returns a full vector representation after a single attempt. In 39% of the cases, a second attempt is required, and in the remaining 1% of the cases, the function is forced to return a vector with at least one zero coefficient. This distribution highlights the probabilistic nature of finding a full vector representation.

Similarly, when using $p = 257$ and $\omega = 5$, the distribution of the number of attempts follows a downward slope, extending up to 8 potential NTT "frequencies" before the probability of finding a full vector representation approaches zero. This behavior suggests that the choice of the root of unity ω can significantly impact the efficiency and determinism of the *select_representation* function.

The decisional process of sorting through multiple slots, each with a certain probability of yielding a good result, is an interesting aspect to consider in the context of the Adh system's security. While the specific details of this process may vary based on the chosen field and root, it is unlikely to reveal any useful information about the original input to the *select_representation* function.

This claim is supported by the fundamental principles of information theory, which suggest that the amount of information that can be extracted from the output of the *select_representation* function is limited by the entropy of the input vector and the properties of the NTT operation. The NTT, being a linear transformation, preserves the statistical properties of the input vector, making it difficult for an attacker to gain any significant advantage by analyzing the decisional process.

Furthermore, the use of rejection sampling techniques in the Adh system, combined with the chaining construction and the careful selection of parameters, further enhances the security by amplifying the complexity and destroying any discernible patterns in the resulting lattice.

In conclusion, the *select_representation* function's behavior in finding a full vector representation is an important aspect to consider in the Adh zero-knowledge proof system. The distribution of the number of attempts required to find a full vector varies based on the chosen field and root, highlighting the probabilistic nature of the process. However, the decisional process itself is unlikely to reveal any useful information about the original input, thanks to the fundamental limitations imposed by information theory and the security measures employed in the Adh system. Further research into the impact of different field and root choices on the efficiency and security of the *select_representation* function could provide valuable insights for optimizing the Adh system's performance and robustness.

6.9 Lattice Density in Module-ISIS

In the context of Module-ISIS, where $B = 257$ (infinity norm), $q = 257$ (prime), $n = 128$ or 256, and $k = 6$ (rank), we consider a full construct with no zero-value coefficients allowed. By rejection sampling out all vectors with zeros, we effectively work with a universe of 1-257 (modulo 257), excluding the zero vector. As the

6.9.1 Hypercube Volume

The volume of the hypercube with side length $B = 256 + 1$ in n dimensions is calculated as:

- For $n = 128$: 257^{128}
- For $n = 256$: 257^{256}

6.9.2 Unit Cell Volume

The volume of the unit cell in the lattice, which is the fundamental parallelotope, is:

- For $n = 128$: 257^{128}
- For $n = 256$: 257^{256}

6.9.3 Packing Density

The packing density is the ratio of the hypercube volume to the unit cell volume:

- For $n = 128$: $\frac{257128}{257^{128}} = 1$
- For $n = 256$: $\frac{257^{256}}{257^{256}} = 1$

The packing density values of 1 indicates that the hypercubes occupy the entire unit cell volume in the lattice. This high packing density suggests that the lattice is densely packed, with no gaps between the hypercubes. This is a function of the infinite norm bound being the same as the prime used for modular arithmetic It is important to note that the rank k does not directly affect the packing density calculation, as it represents the dimension of the module. The high packing density of the Module-ISIS lattice has potential implications for the security and hardness of the underlying problem:

- The dense packing of the lattice makes it more challenging for lattice reduction algorithms like BKZ to find short vectors, potentially enhancing the security of the cryptographic system.
- If the Module-ISIS problem can be reduced to a dense subset sum problem, the high packing density could make it computationally infeasible to solve using known optimization techniques for subset sum problems. This reduction, if possible, would provide a strong argument for the security of the cryptographic system.
- The absence of 0 coefficients in the module-ISIS lattice increases the density of the lattice, making it more challenging for lattice reduction algorithms like BKZ to find short vectors. This property could potentially enhance the security of the cryptographic system.
- If the module-ISIS problem can be reduced to a module-module subset sum problem, the high density of the lattice could make it computationally infeasible to solve using known optimization techniques for subset sum problems. This reduction, if possible, would provide a strong argument for the security of the cryptographic system.
- There are some theoretical results on the hardness of dense lattices, such as the work by Micciancio and Regev [8], which shows that solving certain lattice problems on dense lattices is at least as hard as solving them on general lattices.

7 Practical Implementation Considerations

While not included in the formal security analysis presented in this paper, it is worth noting that in practical implementations of the Adh system, where the first modulus is chosen to be 257 or 65537, we can take advantage of the guaranteed absence of zero coefficients to optimize storage and transport efficiency. By subtracting 1 from each coefficient, we can ensure that the cryptographic variables follow 8-bit or 16-bit alignments, rather than requiring 9 or 17 bits, respectively. This encoding process must be inverted before using the variables in computations. It is important to emphasize that in practical instances, the challenge and random variables should be generated from smaller values corresponding to the appropriate bits of security required by the system. Table 3 presents two prototype instances of the Adh system, illustrating the storage requirements for secrets, public keys, and complete proofs. In the first instance, with parameters $n = 128$,

Instance	n	p	m	B	Size
V	128	257	6	256	SK 192B - PK 192B - CT 192B
VI	256	257	6	256	SK 384B - PK 384B - CT 384B

Table 3: Storage requirements for prototype instances of the Adh system.

$p = 257$, $m = 6$, and $B = 256$, the secrets and public keys each require 192 bytes of storage. The complete proofs consist of a 128-byte proof, a 32-byte random challenge, and a 32-byte message challenge. The second instance, with parameters $n = 256$, $p = 257$, $m = 6$, and $B = 256$, requires 384 bytes for both secrets and public keys. The complete proofs in this case include a 256-byte proof, a 64-byte random challenge, and a 64-byte message challenge. Note that Module-ISIS* will need to store $k + 1$ unique secret keys, one for each extra instance.

7.1 Parameter Selection and Initial Security Estimates

The security of the Adh system relies on the appropriate selection of parameters, such as the modulus q, the dimension n, the rank m, and the norm bound β. These parameters should be chosen to ensure a desired level of security against known attacks, such as lattice reduction and quantum algorithms [2]. To estimate the security complexity from a lattice perspective, we used the specific MSIS hardness estimator located at the repository below.

For the base Module-ISIS instance in the Adh system, we propose the following parameters:

- Dimension $n = 128$
- Rank $m = 6$
- Modulus $q = 257$
- Norm bound $\beta = 257$

To estimate the security of the base Module-ISIS instance, we utilize the MSIS estimator from the `pq-crystals/security-estimates` repository[1].

7.2 Configuration 1: Smaller Parameters $n = 128$

7.2.1 Parameters

- Ring Dimension (n): 128
- MSIS Dimension (w): 768
- Number of Equations (h): 6
- Norm Bound (B): 257
- Modulus (q): 257

7.3 Security Estimates

- Dimensions: 98304
- Block Size: 383
- Probability of Success (*log2(epsilon)*): -79.50
- Average Vectors per Run (*log2 nvector per run*): 79.48
- Length of Shortest Vector (l): 4234.70

7.3.1 Conclusion

The estimator gives us a security level of 112 classical bits, which is lower than acceptable for high-security applications.

7.4 Configuration 2: Larger Parameters $n = 256$

7.4.1 Parameters

- Ring Dimension (n): 256
- MSIS Dimension (w): 1536
- Number of Equations (h): 6
- Norm Bound (B): 257
- Modulus (q): 257

[1] `https://github.com/pq-crystals/security-estimates`

7.5 Security Estimates

- Dimensions: 393216
- Block Size: 889
- Probability of Success (*log2(epsilon)*): -183.48
- Average Vectors per Run (*log2 nvector per run*): 184.48
- Length of Shortest Vector (l): 6111.57

7.5.1 Conclusion

With a significantly enhanced security level of 260 bits, this $n = 256$ configuration offers better protection, potentially suitable for environments requiring very high security standards. The increase in ring dimension and MSIS dimension contributes substantially to the heightened security.

7.6 Estimated Impact of Chaining

The Adh system employs a chaining mechanism, where the output of one Module-ISIS instance is used as the input to the next instance. Let k denote the number of chained instances in the system. The security of the Adh system grows with increasing k, as an adversary would need to solve all k instances of the Module-ISIS+ or Module-ISIS* problem to forge a valid proof. If we assume additive complexity:

- For $k = 1$: The security is equivalent to the base Module-ISIS instance, estimated at least 112 bits.
- For $k = 2$: Security increases to approximately 224 bits.
- For $k = 3$: Security further increases to about 336 bits.
- For $k = 4$: Security reaches around 448 bits, providing high-level security against known attacks.

These estimates serve as a theoretical bound on the security of the Adh system and may be revised upwards as the exact hardness of the Module-ISIS relative to Module-SIS is better understood. Additionally, the attack estimates assume the ability to use extremely large block sizes and dimensions that may not be practical.

The choice of k provides a trade-off between security and efficiency, with higher values of k offering increased security at the cost of larger proof sizes and longer computation times. The optimal value of k should be determined based on the specific security requirements and performance constraints of the application.

In addition to the chaining mechanism, the Adh system incorporates other features that contribute to its security, such as the use of rejection sampling to ensure the uniformity of the generated vectors and the elimination of zero coefficients to create a complete lattice structure. These features further enhance the system's resilience against potential attacks.

In addition to the chaining mechanism, the Adh system incorporates other features that contribute to its security, such as the use of rejection sampling to ensure the uniformity of the generated vectors and the elimination of zero coefficients to create a complete lattice structure. These features further enhance the system's resilience against potential attacks.

8 Experimental Results

To evaluate the resistance of the Adh zero-knowledge proof system against lattice reduction attacks, we conducted experiments using the fplll library [10], a well-established toolkit for lattice-based cryptanalysis. Our primary focus was to assess the effectiveness of various lattice reduction algorithms, including the Block Korkine-Zolotarev (BKZ) algorithm [5], in finding short vectors within the lattices generated by the Adh system.

8.1 FPLL Experimental Setup

We designed an experimental setup in which a loop continuously generated matrices representing the lattice structure of the Adh system. These matrices were then fed into the fplll library, where different lattice reduction algorithms were applied to attempt to find short vectors. We specifically investigated the performance of these algorithms for two parameter settings: $n = 128$ and $n = 256$, corresponding to the dimensions of the lattice used in the Adh system.

8.1.1 BKZ Results

The results of the BKZ experiments exhibited a consistent behavior across different block sizes. For both $n = 128$ and $n = 256$, the norms of the recovered vectors consistently exceeded an average value of 270. Considering that the norm bound in the Adh system is set to 256, these findings suggest that BKZ is not effective in finding sufficiently short vectors to compromise the security of the system.

8.1.2 Non-BKZ Solver Results

In addition to BKZ, we explored other lattice reduction techniques, including the Hermite-Korkine-Zolotarev (HKZ) reduction [7], the Shortest Vector Problem (SVP) solvers, and the Closest Vector Problem (CVP) solvers. When applied to lattices with dimension $n = 128$, these solvers initially appeared to find relatively short vectors within the lattice. However, upon closer inspection, it was revealed that the average norm of the vectors found by these solvers still exceeded 270, failing to breach the norm bound of 256 set in the Adh system.

The behavior of non-BKZ solvers against lattices with dimension $n = 256$ exhibited more variability. In some instances, these solvers returned outputs with higher norm averages compared to the $n = 128$ case. Moreover, the execution time of these solvers against $n = 256$ lattices was significantly longer, sometimes taking several hours to complete.

8.1.3 Conclusion

The experiments conducted with fplll provide valuable insights into the resilience of the Adh zero-knowledge proof system against lattice reduction attacks. Despite the initial appearance of finding short vectors by non-BKZ solvers at dimension $n = 128$, further analysis revealed that the average norm of the recovered vectors consistently exceeded 270, failing to breach the norm bound of 257 set in the Adh system.

The inability of both BKZ and non-BKZ solvers to find vectors shorter than the norm bound in the practically relevant dimensions ($n = 128$ and $n = 256$) suggests that the Adh system exhibits strong resistance against direct lattice reduction and projection-reduction attacks. The dense structure of the lattice, achieved through rejection sampling and the elimination of zero coefficients, is believed to contribute to the difficulty of finding short vectors using traditional lattice reduction methods.

The variable behavior and occasional crashes encountered with non-BKZ solvers against lattices with dimension $n = 256$ highlight the complexity and challenges associated with analyzing the security of the Adh system. Further research is needed to fully understand the implications of these observations and to establish rigorous bounds on the system's resistance against a wider range of cryptanalytic techniques.

8.2 Specific Reduction Attack Scenario Analysis

In this section, we evaluate the potential impact of the attack presented in the paper "Finding short integer solutions when the modulus is small"[4] by Ducas, Espitau, and Postlethwaite on the Adh system with the parameters: $q = 257$, $B = 256$, $m = 6$, $n = 128$, and $k = 4$ chained Module-ISIS+ instances. The attack exploits the Z-shape profile of the reduced basis and performs lattice sieving in projected sublattices to find short solutions.

Let β denote the block size used in the BKZ lattice reduction algorithm. The effectiveness of the attack depends on the number of q-vectors (n_q) remaining in the reduced basis after applying BKZ-β. Table 4 presents the analysis of the attack for various BKZ block sizes. In Scenario 1 ($\beta = 40$), the expected number of q-vectors is $n_q \approx 12$ (based

Scenario	β	n_q	$r - \ell$	Sieving vectors
—1	40	12	-6	- —
—2	60	6	0	- —
—3	80	3	3	2.8 —
—4	100	1	5	16.8 —

Table 4: Revised attack scenarios for different BKZ block sizes

on Table 1 in the paper). The sieving dimension $r - \ell$ is calculated as follows:

$$\begin{aligned} \ell &= n_q + 1 = 13 \\ r &= \min \ell + \beta, m + 1 = \min 53, 7 = 7 \\ r - \ell &= 7 - 13 = -6 \end{aligned}$$

Since the sieving dimension is negative, the attack is not applicable in this scenario. Similarly, in Scenario 2 ($\beta = 60$), the sieving dimension is zero, making the attack inapplicable. In Scenario 3 ($\beta = 80$), the expected number of q-vectors is $n_q \approx 3$ (extrapolated from Table 1). The sieving dimension and the number of sieving vectors are:

$$\begin{aligned} \ell &= n_q + 1 = 4 \\ r &= \min \ell + \beta, m + 1 = \min 84, 7 = 7 \\ r - \ell &= 7 - 4 = 3 \\ \text{Sieving vectors} &= \left(\frac{4}{3}\right)^{\frac{r-\ell}{2}} \approx 2.8 \end{aligned}$$

Although the sieving dimension is positive, the probability of a lifted vector being a valid solution is low due to the small ratio between B and q ($256/257 \approx 0.996$). Consequently, the attack is unlikely to succeed in this scenario. In Scenario 4 ($\beta = 100$), the expected number of q-vectors is $n_q \approx 1$. The sieving dimension and the number of sieving vectors are:

$$\begin{aligned} \ell &= n_q + 1 = 2 \\ r &= \min \ell + \beta, m + 1 = \min 102, 7 = 7 \\ r - \ell &= 7 - 2 = 5 \\ \text{Sieving vectors} &= \left(\frac{4}{3}\right)^{\frac{r-\ell}{2}} \approx 16.8 \end{aligned}$$

While the sieving dimension is positive and the number of sieving vectors is larger, the small ratio between B and q still limits the success probability of the attack.

8.2.1 Attack Analysis Conclusion

Based on the analysis with the parameters ($q = 257$, $n = 128$, $B = 257$), the attack described in the paper appears to have limited effectiveness against the Adh system. The small lattice dimension m and the close proximity of the modulus q to the norm bound B reduce the applicability and success probability of the attack.

However, it is essential to note that this analysis focuses solely on the specific attack outlined in the paper and relies on the assumptions made therein. It does not preclude the existence of other attacks or potential improvements to the current attack that could impact the security of the Adh system.

8.3 Resistance to State of the Art Projection Reductions

A recent paper by Ducas, Espitau, and Postlethwaite [1] presents a new attack on lattice-based cryptosystems that exploits the $\mathbb{Z}$-shape profile of the reduced basis and performs lattice sieving in projected sublattices to find short solutions. However, this attack is not effective against the Adh system due to the high density of the lattice. In the Adh system, the lattice is constructed to be maximally dense, with a packing density of 1. This means that the product of the first minimum of the primal lattice and the first minimum of the dual lattice is much higher than 1:

$$\lambda_1(\mathcal{L}) \cdot \lambda_1(\mathcal{L}^*) \gg 1 \tag{11}$$

The high density of the lattice makes it resistant to the new attack, as the success probability of the attack depends on the ratio between the bound B and the modulus q. In the Adh system, this ratio is very close to 1 ($B/q \approx 0.996$), which significantly limits the applicability and success probability of the attack. Therefore, while the new attack presented by Ducas et al. is an important advancement in lattice cryptanalysis, it does not pose a significant threat to the security of the Adh system due to the carefully designed high-density lattice structure.

9 Performance Evaluation

To assess the performance of the Adh zero-knowledge proof system, we conducted benchmarking experiments on an Apple M2 Max MacBook Pro using Python 3.12.3. We measured the operations per second for key generation, proof generation, and proof verification with two different parameter settings: $n = 128$ and $n = 256$. The results are summarized in Table 5. The performance results demonstrate the impact of the parameter n on the efficiency of the Adh system. As expected, increasing the value of n from 128 to 256 leads to a significant decrease in the number of operations per second for all three components: key generation, proof generation, and proof verification.

Operation	$n = 128$	$n = 256$
Key Generation	84.92 ops/s	26.54 ops/s
Proof Generation	131.93 ops/s	51.32 ops/s
Proof Verification	890.47 ops/s	613.50 ops/s

Table 5: Performance results for the Adh zero-knowledge proof system.

It is important to note that the current implementation of the Adh system is written in pure Python, which is known for its relatively slower execution compared to lower-level languages like C. These numbers represent the lower bound for performance as no optimization efforts have been made to code that was benchmarked. The performance figures presented in Table 5 reflect this limitation and should be considered as a baseline for future optimizations.

To achieve better performance, we will implement the Adh system in cross platform ANSI C, taking advantage of hardware vector acceleration techniques where possible. By leveraging the capabilities of modern processors, such as Intel's Advanced Vector Extensions (AVX) or ARM's Neon instructions, significant speedups can be obtained in operations like the Number Theoretic Transform (NTT) and polynomial arithmetic.

Furthermore, the use of parallel computing techniques and optimized libraries for lattice-based cryptography can further enhance the efficiency of the Adh system. As we feel the final implementation will be significantly more performant, we suggest using these numbers as a heuristic.

10 Comparative Analysis

The Adh zero-knowledge proof system introduces several novel features that distinguish it from other state-of-the-art proof systems. One of the key advantages of the Adh system is its reliance on the Module-ISIS problem, which provides a strong foundation for its security in the post-quantum setting. The use of lattice-based cryptography ensures that the Adh system is resistant to attacks by quantum computers, making it a promising candidate for future-proof secure computation. Compared to other zero-knowledge proof systems based on traditional assumptions, such as discrete logarithms or factoring, the Adh system offers a higher level of security and long-term resilience. The Module-ISIS problem, along with its variants Module-ISIS+ and Module-ISIS*, provides a rich and flexible framework for constructing secure proof systems with advanced features like chaining and multi-level proofs.

Another distinctive aspect of the Adh system is its use of nested Number Theoretic Transform (NTT) operations. The NTT plays a crucial role in enabling efficient polynomial arithmetic, which is essential for the performance of lattice-based cryptographic protocols. The Adh system leverages the properties of the NTT to achieve fast and compact proof generation and verification, making it suitable for practical applications.

The Adh system also incorporates advanced techniques such as rejection sampling and the elimination of zero coefficients to maintain a complete lattice structure. These techniques contribute to the system's security by reducing the attack surface and making it harder for adversaries to exploit structural weaknesses. The rejection sampling approach ensures the uniformity of the generated vectors, preventing potential biases that could be exploited by attackers.

Furthermore, the Adh system supports multiple levels of proof generation and verification, providing flexibility and adaptability to different security requirements and performance constraints. This multi-level feature allows for the construction of more complex proof systems and enables the Adh system to be used in a wider range of applications.

In comparison to other lattice-based zero-knowledge proof systems, such as those based on the Ring-SIS or Ring-LWE problems, the Adh system offers several advantages. The Module-ISIS problem provides a more flexible and efficient framework for constructing proofs, as it allows for the use of smaller moduli and dimensions while maintaining a high level of security. The Adh system's chaining mechanism and multi-level proofs also enable more advanced features and improved scalability compared to simpler lattice-based proof systems.

11 Potential Use Cases and Applications

The Adh zero-knowledge proof system, with its unique lattice-based construction and compact key and proof sizes, offers a versatile foundation for various cryptographic applications and protocols. The following subsections explore potential use cases where the Adh system could provide secure and efficient solutions.

11.1 Key Exchange Mechanism (KEM)

The Adh system's underlying one-way chosen plaintext attack (OW-CPA) resistant scheme, related to the subset sum problem, can be transformed into an indistinguishability under chosen-ciphertext and prove attack (IND-CCPA) secure key exchange mechanism (KEM). This KEM would enable parties to establish a shared secret key for secure communication, leveraging the hardness of the Module-ISIS problem and its variants. The compact key sizes of the Adh system could lead to efficient key exchange protocols, particularly suited for resource-constrained environments.

11.2 Digital Signatures

By applying the Fiat-Shamir transform to the Adh system, it is possible to construct existentially unforgeable under chosen message attack (EU-CMA) digital signature schemes. These signatures would allow users to sign messages and verify the authenticity of the signatures, providing a secure means of authentication and non-repudiation. The compact signature sizes offered by the Adh system could be advantageous in scenarios where bandwidth or storage is limited, such as in Internet of Things (IoT) devices or blockchain applications.

11.3 Identity-Based and Key-Policy Based Cryptography

The Adh system's lattice construction opens up possibilities for identity-based and key-policy based cryptography. In identity-based cryptography, users' identities (e.g., email addresses) serve as their public keys, simplifying key management and distribution. Key-policy based cryptography enables fine-grained access control by associating policies with keys, determining who can access encrypted data. The Adh system's compact key sizes and efficient operations could make it well-suited for implementing these advanced cryptographic primitives, enabling secure and flexible access control mechanisms.

11.4 Secure Messaging Protocol

The PKEMNO NIZK (Public Key Exchange Mechanism with Non-Interactive Zero-Knowledge Opening) secure messaging protocol, introduced in the paper, leverages the unique characteristics of the Adh system. This protocol ensures the confidentiality and integrity of exchanged messages, making it suitable for secure communication applications. The absence of a traditional decryption function and the use of the ZKVolute operation in the Adh system could provide enhanced security and privacy features compared to traditional messaging protocols.

11.5 Proof of Knowledge

The Adh system's trapdoor-based proof of knowledge capabilities enable the construction of protocols where a prover can demonstrate knowledge of a secret without revealing it to the verifier. This property has applications in authentication, access control, and privacy-preserving systems. For example, a user could prove their identity or membership in a group without disclosing sensitive information. The zero-knowledge proofs generated by the Adh system could be used to build secure and privacy-enhancing authentication and authorization mechanisms.

11.6 Homomorphic Cryptography

The Adh system's homomorphic properties, being a subcategory of 'somewhat' or 'partially' homomorphic cryptographic systems, enable computations to be performed on encrypted data without decrypting it first. This capability opens up possibilities for privacy-preserving computations, such as secure multiparty computation or outsourced computation on sensitive data. The compact key and ciphertext sizes of the Adh system could make it more practical and efficient compared to other homomorphic encryption schemes, potentially enabling secure computation in resource-constrained environments.

12 Known Issues

12.1 Side-Channel Vulnerabilities and Mitigation Techniques

While the Adh zero-knowledge proof system demonstrates strong security properties, it is important to consider potential side-channel vulnerabilities, particularly due to its heavy reliance on NTT operations. Side-channel attacks, such as timing attacks or power analysis attacks, can potentially leak sensitive information about the secret key or the internal state of the system. To mitigate side-channel vulnerabilities, several techniques can be employed:

- **Hardware acceleration**: Leveraging hardware acceleration techniques, such as Intel's AVX (Advanced Vector Extensions) or ARM's Neon vector math opcodes, can help in reducing the variance in execution time and power consumption. These accelerated instructions provide a more consistent and efficient execution environment, making it harder for attackers to exploit timing or power variations.
- **Constant-time NTT implementations**: Implementing NTT operations in a constant-time manner is crucial to prevent timing-based side-channel attacks. Constant-time NTT algorithms ensure that the execution time is independent of the input data, eliminating potential leakage of sensitive information through timing variations. Techniques such as using fixed-point arithmetic, avoiding conditional branches, and employing bit-slicing can contribute to constant-time implementations.
- **Randomization and masking**: Randomization techniques, such as blinding or masking, can be applied to the NTT computations to make them more resilient against side-channel attacks. By introducing random noise or splitting sensitive values into multiple shares, the statistical dependency between the processed data and the leaked side-channel information can be reduced.
- **Secure memory management**: Careful management of sensitive data in memory is essential to prevent memory-based side-channel attacks. Techniques like using secure memory allocation, clearing memory after use, and avoiding memory reuse can help in mitigating memory leakage vulnerabilities.
- **Oversampling**: By measuring probabilistic rates of success of a given operation we can bound a number of samples to be taken for a given operation to ensure one will succeed within a certain range of probability. By exchanging efficiency for computation we may find constant time solutions.

13 Open Questions and Future Work

The research presented in this paper on the Adh zero-knowledge proof system raises several interesting open questions and potential avenues for future work. While the paper provides a comprehensive analysis of the system's security and performance, there are still areas that warrant further investigation and exploration.

13.1 Verified Formal Security Proofs

One important open item is the continued refinement and validation of formal security proofs for the various aspects of the Adh system. While the paper presents empirical evidence, multiple arguments supporting the security of the system, and presents our formal reductions and proofs, continuous peer review rigorous, mathematical analysis, and refinement over time will provide stronger guarantees. We acknowledge the novelty of some of proofs presented in the paper and encourage peer review and welcome feedback, improvements or corrections.

13.2 Parameter Optimization and Trade-offs

Another area for future research is the optimization of the Adh system's parameters and the exploration of trade-offs between security and efficiency. The paper presents specific parameter choices and provides experimental results, but a more comprehensive analysis of parameter selection could yield further improvements. This work will presented in a subsequent paper. Some questions to include:

- What is the optimal choice of the prime modulus q and the dimension n to balance security and performance?
- How does the number of chained instances k affect the security and efficiency of the system, and what is the optimal value of k for different security levels?
- Can the rejection sampling technique be further optimized to reduce the computational overhead while maintaining the desired statistical properties?
- Complexity comparison of various combinations of configurations beyond the base cases presented in this work.

13.3 Applications and Integration to Protocols

The Adh zero-knowledge proof system has the potential to be applied in various cryptographic protocols and privacy-preserving applications. Future work will investigate the integration of the Adh system into existing protocols and explore new use cases. Some potential non-standard directions include:

- Integrating the Adh system into privacy-preserving authentication protocols, such as anonymous credentials or attribute-based signatures.
- Exploring the use of the Adh system in secure multi-party computation protocols, enabling efficient and private computations among multiple participants.
- Developing privacy-preserving blockchain applications that leverage the Adh system for confidential transactions and smart contracts.
- Supporting Swarm networking.

13.4 Long-Term Security and Post-Quantum Cryptography

As the field of quantum computing advances, it is crucial to assess the long-term security of cryptographic systems against potential quantum attacks. While the Adh system is based on lattice problems that are believed to be resistant to quantum algorithms, further research is needed to solidify its post-quantum security guarantees. Future work could focus on:

- Conducting a thorough analysis of the Adh system's resistance against known quantum algorithms, such as Shor's algorithm or Grover's algorithm.
- Exploring the use of quantum-resistant primitives, such as quantum-safe hash functions or post-quantum digital signature schemes, in conjunction with the Adh system.

- Investigating the potential impact of future advancements in quantum computing on the security of the Adh system and developing mitigation strategies.

In conclusion, the research presented in this paper on the Adh zero-knowledge proof system opens up a wide range of exciting possibilities for future work. From formal security proofs and parameter optimization to implementation enhancements and practical applications, there are numerous avenues to explore and contribute to the field of lattice-based cryptography and zero-knowledge proofs. The open questions and challenges identified in this section provide a roadmap for researchers and practitioners to further advance the state of the art and strengthen the foundations of the Adh system.

14 Conclusion

In this work, we introduced the Adh zero-knowledge proof system, a novel lattice-based protocol that achieves compact proofs and strong security guarantees under the Module-ISIS assumption and its variants. Our core technical contributions include:

- A comprehensive analysis of the Module-ISIS problem and its connection to the security of Adh, including formal definitions of the ISIS+, ISIS*, and ISIS** variants.
- An in-depth study of the effectiveness of BKZ and other lattice reduction techniques against Adh, demonstrating the system's resistance to conventional and state-of-the-art cryptanalytic attacks.
- Concrete parameter selection and performance benchmarks, showcasing Adh's practicality and efficiency compared to existing post-quantum alternatives.

Our work also identified several avenues for further research, including optimizations to the zero-knowledge protocol, additional side-channel countermeasures, and performance optimizations. By contributing novel cryptographic techniques and rigorous security analysis, this paper aims to advance the state of the art in post-quantum zero-knowledge proofs and lay the foundation for secure and efficient protocols in the quantum computing era.

Fundamentally, the Adh system represents a promising step towards achieving the long-standing goal of compact, flexible, and quantum-secure zero-knowledge proofs. Its unique blend of lattice-based techniques and rejection sampling enables new possibilities for cryptographic protocol design. We hope this work spurs further innovations at the intersection of lattice cryptography and zero-knowledge, paving the way for a new generation of privacy-preserving technologies that can withstand the challenges of the post-quantum world.

A Appendix

A.1 Proof of Reduction to Module-ISIS

Theorem 11 (Reduction to Module-ISIS)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the

Module-ISIS problem. Given a Module-ISIS instance $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS instance. It sets the modulus to q, the dimension to n, the rank to m, and the norm bound to β.
2. $\mathcal{B}$ generates the public key $\mathbf{pk}$ by computing $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk}_{\text{chal}}, \mathbf{pk}_{\text{rand}})$, where $\mathbf{sk}$ is a randomly generated secret key, $\mathbf{pk}_{\text{chal}}$ is the public challenge, and $\mathbf{pk}_{\text{rand}}$ is the public randomness. $\mathcal{B}$ sets $\mathbf{sk} = \mathbf{A}$ and $\mathbf{pk}_{\text{chal}} = \mathbf{t}$. $\mathcal{B}$ sends $\mathbf{pk}$ to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a forged proof $(\mathbf{sig}\text{'}\mathbf{sig}\text{'}_{\text{chal}}\mathbf{sig}^*_{\text{rand}})$.
4. $\mathcal{B}$ verifies the forged proof using the verification algorithm of the Adh system. If the proof is accepted, $\mathcal{B}$ proceeds to the next step. Otherwise, $\mathcal{B}$ aborts.
5. $\mathcal{B}$ computes $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$, where $\mathbf{sig}$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}$.
6. If $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||_\infty \leq 2\beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the Module-ISIS instance.

The analysis of the success probability of $\mathcal{B}$ follows similarly to the reduction to Module-ISIS+ in Appendix A.2. If $\mathcal{A}$ succeeds in forging a valid proof with non-negligible probability, then $\mathbf{z}$ satisfies $\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \bmod q$ and $||\mathbf{z}||_\infty \leq 2\beta$, solving the Module-ISIS instance. The success probability of $\mathcal{B}$ is equal to the success probability of $\mathcal{A}$, which is assumed to be non-negligible. Therefore, if the Adh system is susceptible to forgery attacks, then Module-ISIS is solvable with non-negligible probability, contradicting the assumed hardness of Module-ISIS. □

A.2 Proof of Reduction to Module-ISIS+

Theorem 12 (Reduction to Module-ISIS+)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS+ problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS+ problem. Given a Module-ISIS+ instance $(\mathbf{A}\_1, \mathbf{t}\_1, \ldots, \mathbf{t}\_k, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS+ instance. It sets the modulus to q, the dimension to n, the rank to m, and the norm bound to β.
2. $\mathcal{B}$ generates the public key $\mathbf{pk}$ by computing $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk}\text{\_chal}, \mathbf{pk}\text{\_rand})$, where $\mathbf{sk}$ is a randomly generated secret key, $\mathbf{pk}$_chal is the public challenge, and $\mathbf{pk}$_rand is the public randomness. $\mathcal{B}$ sends $\mathbf{pk}$ to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a forged proof ($\mathbf{sig}$'$\mathbf{sig}$_chal'$\mathbf{sig}$_rand*).
4. $\mathcal{B}$ verifies the forged proof using the verification algorithm of the Adh system. If the proof is accepted, $\mathcal{B}$ proceeds to the next step. Otherwise, $\mathcal{B}$ aborts.
5. $\mathcal{B}$ computes $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$, where $\mathbf{sig}$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}$.
6. If $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||\_\infty \leq 2\beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the Module-ISIS+ instance.

To analyze the success probability of $\mathcal{B}$, we observe that if $\mathcal{A}$ succeeds in forging a valid proof with non-negligible probability, then the forged proof ($\mathbf{sig}$'$\mathbf{sig}$_chal'$\mathbf{sig}$_rand$^)$ must satisfy the verification equation:

$$\mathrm{ZKVolute}(\mathbf{pk}, \mathbf{sig}\_\mathrm{chal}'\mathbf{sig}\_\mathrm{rand}^{)} = \mathrm{ZKVolute}(\mathbf{sig}'\mathbf{pk}\_\mathrm{chal}, \mathbf{pk}\_\mathrm{rand}) \tag{12}$$

Substituting $\mathbf{pk} = \mathrm{ZKVolute}(\mathbf{sk}, \mathbf{pk}\_\mathrm{chal}, \mathbf{pk}\_\mathrm{rand})$ and rearranging the terms, we obtain:

$$\mathrm{ZKVolute}(\mathbf{sk}, \mathbf{sig}\_\mathrm{chal}'\mathbf{sig}\_\mathrm{rand}^{)} = \mathbf{sig}^* \tag{13}$$

Let $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$, where $\mathbf{sig}$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}$. Then, we have:

$$\mathrm{ZKVolute}(\mathbf{sk}, \mathbf{sig}\_\mathrm{chal}'\mathbf{sig}\_\mathrm{rand}^{)} - \mathrm{ZKVolute}(\mathbf{sk}, \mathbf{sig}\_\mathrm{chal}, \mathbf{sig}\_\mathrm{rand}) = \mathbf{z} \tag{14}$$

By the linearity of the ZKVolute function, we can rewrite this as:

$$\mathrm{ZKVolute}(\mathbf{sk}, \mathbf{sig}\_\mathrm{chal}^* - \mathbf{sig}\_\mathrm{chal}, \mathbf{sig}\_\mathrm{rand}^* - \mathbf{sig}\_\mathrm{rand}) = \mathbf{z} \tag{15}$$

Now, recall that in the Module-ISIS+ problem, we have:

$$\begin{aligned}
\mathbf{A}_1 \cdot \mathbf{z} &= \mathbf{t}_1 \bmod q && (16)\\
\mathbf{A}_2 \cdot \mathbf{z} &= \mathbf{t}_2 \bmod q && (17)\\
&\vdots && (18)\\
\mathbf{A}_k \cdot \mathbf{z} &= \mathbf{t}_k \bmod q && (19)\\
& && (20)
\end{aligned}$$

where $\mathbf{A}i = \mathrm{NTT}(\mathbf{A}i-1) \cdot \mathrm{NTT}(\mathbf{R})$ for $i = 2, \ldots, k$, with $\mathbf{R}$ being a random matrix in $R\_q^{m \times m}$. By the construction of the Adh system, we have:

$$\begin{aligned}
\mathbf{A}_1 &= \mathbf{sk} && (21)\\
\mathbf{t}_1 &= \mathbf{sig}_\mathrm{chal}^* - \mathbf{sig}_\mathrm{chal} && (22)\\
\mathbf{t}_2 &= \mathrm{NTT}(\mathbf{sig}_\mathrm{chal}^* - \mathbf{sig}_\mathrm{chal}) \cdot \mathrm{NTT}(\mathbf{sig}_\mathrm{rand}^* - \mathbf{sig}_\mathrm{rand}) && (23)\\
&\vdots && (24)\\
\mathbf{t}_k &= \mathrm{NTT}^{(k-1)}(\mathbf{sig}_\mathrm{chal}^* - \mathbf{sig}_\mathrm{chal}) \cdot \mathrm{NTT}^{(k-1)}(\mathbf{sig}_\mathrm{rand}^* - \mathbf{sig}_\mathrm{rand}) && (25)\\
& && (26)
\end{aligned}$$

Therefore, if $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||\_\infty \leq 2\beta$, then $\mathbf{z}$ is a valid solution to the Module-ISIS+ instance. The success probability of $\mathcal{B}$ is equal to the success probability of $\mathcal{A}$ in forging a valid proof, which is assumed to be non-negligible. Therefore, $\mathcal{B}$ solves the Module-ISIS+ problem with non-negligible probability, contradicting the assumed hardness of Module-ISIS+. □

This reduction demonstrates that if an adversary can forge a valid proof in the Adh system with non-negligible probability, then the Module-ISIS+ problem can be solved with non-negligible probability, contradicting the assumed hardness of Module-ISIS+. Therefore, the Adh system is secure against forgery attacks, assuming the hardness of the Module-ISIS+ problem.

A.3 Proof of Reduction to Module-ISIS*

Theorem 13 (Reduction to Module-ISIS*)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS* problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS* problem. Given a Module-ISIS* instance
$(\mathbf{A}\_1, \ldots, \mathbf{A}\_k, \mathbf{t}\_1, \ldots, \mathbf{t}\_k, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS* instance. It sets the modulus to q, the dimension to n, the rank to m, and the norm bound to β.
2. $\mathcal{B}$ generates the public keys $\mathbf{pk}\_1, \ldots, \mathbf{pk}\_k$ by computing $\mathbf{pk}\_i = \text{ZKVolute}(\mathbf{sk}\_i, \mathbf{pk}\_\text{chal}\_i, \mathbf{pk}\_\text{rand}\_i)$, where $\mathbf{sk}\_i$ is a randomly generated secret key, $\mathbf{pk}\_\text{chal}\_i$ is the public challenge, and $\mathbf{pk}\_\text{rand}\_i$ is the public randomness for the i-th instance. $\mathcal{B}$ sends $\mathbf{pk}\_1, \ldots, \mathbf{pk}\_k$ to $\mathcal{A}$.
3. $\mathcal{A}$ outputs forged proofs

   $(\mathbf{sig}\_1{}^{,}\mathbf{sig}\_\text{chal}\_1{}^{,}\mathbf{sig}\_\text{rand}\_1{}^{)}, \ldots, (\mathbf{sig}\_k{}^{,}\mathbf{sig}\_\text{chal}\_k{}^{,}\mathbf{sig}\_\text{rand}\_k{}^{)}.$
4. $\mathcal{B}$ verifies the forged proofs using the verification algorithm of the Adh system. If all the proofs are accepted, $\mathcal{B}$ proceeds to the next step. Otherwise, $\mathcal{B}$ aborts.
5. For each $i = 1, \ldots, k$, $\mathcal{B}$ computes $\mathbf{z}\_i = \mathbf{sig}\_i^* - \mathbf{sig}\_i$, where $\mathbf{sig}\_i$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}\_i$.
6. If $\mathbf{z}\_i \neq \mathbf{0}$ and $||\mathbf{z}i||\infty \leq 2\beta$ for all $i = 1, \ldots, k$, then $\mathcal{B}$ outputs $(\mathbf{z}\_1, \ldots, \mathbf{z}\_k)$ as a solution to the Module-ISIS* instance.

To analyze the success probability of $\mathcal{B}$, we observe that if $\mathcal{A}$ succeeds in forging valid proofs with non-negligible probability, then the forged proofs $(\mathbf{sig}\_i{}^{,}\mathbf{sig}\_\text{chal}\_i{}^{,}\mathbf{sig}\_\text{rand}\_i{}^{)}$ for $i = 1, \ldots, k$ must satisfy the verification equations:

$$\text{ZKVolute}(\mathbf{pk}\_i, \mathbf{sig}\_\text{chal}\_i{}^{,}\mathbf{sig}\_\text{rand}\_i{}^{)} = \text{ZKVolute}(\mathbf{sig}\_i{}^{,}\mathbf{pk}\_\text{chal}\_i, \mathbf{pk}\_\text{rand}\_i) \tag{27}$$

Substituting $\mathbf{pk}\_i = \text{ZKVolute}(\mathbf{sk}\_i, \mathbf{pk}\_\text{chal}\_i, \mathbf{pk}\_\text{rand}\_i)$ and rearranging the terms, we obtain:

$$\text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i{}^{,}\mathbf{sig}\_\text{rand}\_i{}^{)} = \mathbf{sig}\_i^* \tag{28}$$

Let $\mathbf{z}\_i = \mathbf{sig}\_i^* - \mathbf{sig}\_i$, where $\mathbf{sig}\_i$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}\_i$. Then, we have:

$$\text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i{}^{,}\mathbf{sig}\_\text{rand}\_i{}^{)} - \text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i, \mathbf{sig}\_\text{rand}\_i) = \mathbf{z}\_i \tag{29}$$

By the linearity of the ZKVolute function, we can rewrite this as:

$$\text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i^* - \mathbf{sig}\_\text{chal}\_i, \mathbf{sig}\_\text{rand}\_i^* - \mathbf{sig}\_\text{rand}\_i) = \mathbf{z}\_i \tag{30}$$

Now, recall that in the Module-ISIS* problem, we have:

$$\mathbf{A}\_1 \cdot \mathbf{z}\_1 = \mathbf{t}\_1 \bmod q \;\; \mathbf{A}\_2 \cdot \mathbf{z}\_2 \quad = \mathbf{t}\_2 \bmod q \; \vdots \; \mathbf{A}\_k \cdot \mathbf{z}\_k \quad = \mathbf{t}\_k \bmod q \tag{31}$$

where $\mathbf{t}\_i = \text{mask}(\mathbf{A}i{\cdot}\mathbf{z}i - 1){\cdot}\mathbf{z}\_i$ for $i = 2, \ldots, k$, with $\mathbf{t}\_1 = \mathbf{A}\_1{\cdot}\mathbf{z}\_1$. By the construction of the Adh system, we have:

$$\mathbf{A}\_i = \mathbf{sk}\_i\ \mathbf{t}\_1 \qquad\qquad = \mathbf{sig}\_\text{chal}\_1^* - \mathbf{sig}\_\text{chal}\_1\ \mathbf{t}\_i \tag{32}$$

$$\mathbf{t}\_i = \text{mask}(\mathbf{sk}i \cdot \mathbf{z}i - 1) \cdot (\mathbf{sig}\_\text{chal}\_i^* - \mathbf{sig}\_\text{chal}\_i) \tag{33}$$

for $i = 2, \ldots, k$. Therefore, if $\mathbf{z}\_i \neq \mathbf{0}$ and $||\mathbf{z}i||\infty \leq 2\beta$ for all $i = 1, \ldots, k$, then $(\mathbf{z}\_1, \ldots, \mathbf{z}\_k)$ is a valid solution to the Module-ISIS* instance. The success probability of $\mathcal{B}$ is equal to the success probability of $\mathcal{A}$ in forging valid proofs, which is assumed to be non-negligible. Therefore, $\mathcal{B}$ solves the Module-ISIS* problem with non-negligible probability, contradicting the assumed hardness of Module-ISIS*. □

This reduction demonstrates that if an adversary can forge valid proofs in the Adh system with non-negligible probability, then the Module-ISIS* problem can be solved with non-negligible probability, contradicting the assumed hardness of Module-ISIS*. Therefore, the Adh system is secure against forgery attacks, assuming the hardness of the Module-ISIS* problem.

A.4 Proof of Reduction to Module-ISIS**

Theorem 14 (Reduction to Module-ISIS**)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS** problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS** problem. Given a Module-ISIS** instance

$(\mathbf{A}\_1, \ldots, \mathbf{A}\_k, \mathbf{t}\_1, \ldots, \mathbf{t}\_k, p\_1, \ldots, p\_k, \omega\_1, \ldots, \omega\_k, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS** instance. It sets the moduli to $p\_1, \ldots, p\_k$, the dimension to n, the rank to m, the norm bound to β, and the roots of unity to $\omega\_1, \ldots, \omega\_k$.
2. $\mathcal{B}$ generates the public keys $\mathbf{pk}\_1, \ldots, \mathbf{pk}\_k$ by computing $\mathbf{pk}\_i = \text{ZKVolute}(\mathbf{sk}\_i, \mathbf{pk}\_\text{chal}\_i, \mathbf{pk}\_\text{rand}\_i)$, where $\mathbf{sk}\_i$ is a randomly generated secret key, $\mathbf{pk}\_\text{chal}\_i$ is the public challenge, and $\mathbf{pk}\_\text{rand}\_i$ is the public randomness for the i-th instance. $\mathcal{B}$ sends $\mathbf{pk}\_1, \ldots, \mathbf{pk}\_k$ to $\mathcal{A}$.
3. $\mathcal{A}$ outputs forged proofs $(\mathbf{sig}\_1{}^{,}\mathbf{sig}\_\text{chal}\_1{}^{,}\mathbf{sig}\_\text{rand}\_1{}^{)}, \ldots, (\mathbf{sig}\_k{}^{,}\mathbf{sig}\_\text{chal}\_k{}^{,}\mathbf{sig}\_\text{rand}\_k{}^{)}$.
4. $\mathcal{B}$ verifies the forged proofs using the verification algorithm of the Adh system. If all the proofs are accepted, $\mathcal{B}$ proceeds to the next step. Otherwise, $\mathcal{B}$ aborts.
5. For each $i = 1, \ldots, k$, $\mathcal{B}$ computes $\mathbf{z}\_i = \mathbf{sig}\_i^* - \mathbf{sig}\_i$, where $\mathbf{sig}\_i$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}\_i$.
6. If $\mathbf{z}\_i \neq \mathbf{0}$ and $||\mathbf{z}i||\infty \leq 2\beta$ for all $i = 1, \ldots, k$, then $\mathcal{B}$ outputs $(\mathbf{z}\_1, \ldots, \mathbf{z}\_k)$ as a solution to the Module-ISIS** instance.

The analysis of the success probability of $\mathcal{B}$ follows similarly to the reduction to Module-ISIS*. If $\mathcal{A}$ succeeds in forging valid proofs with non-negligible probability, then the forged proofs $(\mathbf{sig}\_i{}^{,}\mathbf{sig}\_\text{chal}\_i{}^{,}\mathbf{sig}\_\text{rand}\_i{}^{)}$ for $i = 1, \ldots, k$ must satisfy the verification equations:

$$\text{ZKVolute}(\mathbf{pk}\_i, \mathbf{sig}\_\text{chal}\_i{}^{,}\mathbf{sig}\_\text{rand}\_i{}^{)} = \text{ZKVolute}(\mathbf{sig}\_i{}^{,}\mathbf{pk}\_\text{chal}\_i, \mathbf{pk}\_\text{rand}\_i) \tag{34}$$

Substituting $\mathbf{pk}\_i = \text{ZKVolute}(\mathbf{sk}\_i, \mathbf{pk}\_\text{chal}\_i, \mathbf{pk}\_\text{rand}\_i)$ and rearranging the terms, we obtain:

$$\text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i^{,}\mathbf{sig}\_\text{rand}\_i^{)} = \mathbf{sig}\_i^* \tag{35}$$

Let $\mathbf{z}\_i = \mathbf{sig}\_i^* - \mathbf{sig}\_i$, where $\mathbf{sig}\_i$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}\_i$. Then, we have:

$$\text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i^{,}\mathbf{sig}\_\text{rand}\_i^{)} - \text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i, \mathbf{sig}\_\text{rand}\_i) = \mathbf{z}\_i \tag{36}$$

By the linearity of the ZKVolute function, we can rewrite this as:

$$\text{ZKVolute}(\mathbf{sk}\_i, \mathbf{sig}\_\text{chal}\_i^* - \mathbf{sig}\_\text{chal}\_i, \mathbf{sig}\_\text{rand}\_i^* - \mathbf{sig}\_\text{rand}\_i) = \mathbf{z}\_i \tag{37}$$

Now, recall that in the Module-ISIS** problem, we have:

$$\mathbf{A}\_1 \cdot \mathbf{z}\_1 = \mathbf{t}\_1 \bmod p\_1 \ \mathbf{A}\_2 \cdot \mathbf{z}\_2 \quad = \mathbf{t}\_2 \bmod p\_2 \vdots \ \mathbf{A}\_k \cdot \mathbf{z}\_k \quad = \mathbf{t}\_k \bmod p\_k \tag{38}$$

where $\mathbf{t}\_i = \text{mask}(\mathbf{A}i \cdot \mathbf{z}i - 1) \cdot \mathbf{z}\_i$ for $i = 2, \ldots, k$, with $\mathbf{t}\_1 = \mathbf{A}\_1 \cdot \mathbf{z}\_1$. By the construction of the Adh system, we have:

$$\mathbf{A}\_i = \mathbf{sk}\_i \ \mathbf{t}\_1 \quad = \mathbf{sig}\_\text{chal}\_1^* - \mathbf{sig}\_\text{chal}\_1 \ \mathbf{t}\_i = \text{mask}(\mathbf{sk}i \cdot \mathbf{z}i - 1) \cdot (\mathbf{sig}\_\text{chal}\_i^* - \mathbf{sig}\_\text{chal}\_i) \tag{39}$$

for $i = 2, \ldots, k$.

Therefore, if $\mathbf{z}\_i \neq \mathbf{0}$ and $||\mathbf{z}i||\infty \leq 2\beta$ for all $i = 1, \ldots, k$, then $(\mathbf{z}\_1, \ldots, \mathbf{z}\_k)$ is a valid solution to the Module-ISIS** instance. The success probability of $\mathcal{B}$ is equal to the success probability of $\mathcal{A}$ in forging valid proofs, which is assumed to be non-negligible. Therefore, $\mathcal{B}$ solves the Module-ISIS** problem with non-negligible probability, contradicting the assumed hardness of Module-ISIS**. □

This reduction demonstrates that if an adversary can forge valid proofs in the Adh system with non-negligible probability, then the Module-ISIS** problem can be solved with non-negligible probability, contradicting the assumed hardness of Module-ISIS**. Therefore, the Adh system is secure against forgery attacks, assuming the hardness of the Module-ISIS** problem.

A.5 Proof of Soundness for Module-ISIS+

Theorem 15 (Soundness)**.** *The Adh zero-knowledge proof system is sound, assuming the hardness of the Module-ISIS problem. That is, a computationally bounded adversary cannot convince the verifier of a false statement, except with negligible probability.*

Proof. Suppose there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can convince the verifier of a false statement with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS problem. Given a Module-ISIS instance $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS instance. It sets the modulus to q, the dimension to n, the rank to m, and the norm bound to β.
2. $\mathcal{B}$ generates the public key $\mathbf{pk}$ by selecting a secret key $\mathbf{sk}$, a public challenge $\mathbf{pk}_\text{chal}$, and a randomizing value $\mathbf{pk}_\text{rand}$ uniformly at random from the range $[1, 256]$. It then computes the convolution part of the public key as $\mathbf{pk}' = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk}_\text{chal}, \mathbf{pk}_\text{rand})$ and sets $\mathbf{pk} = (\mathbf{pk}', \mathbf{pk}_\text{chal}, \mathbf{pk}_\text{rand})$. $\mathcal{B}$ sends $\mathbf{pk}$ to $\mathcal{A}$.

3. $\mathcal{A}$ outputs a false statement and a proof $(\mathbf{sig}'\mathbf{sig}'_{\text{chal}}\mathbf{sig}^*_{\text{rand}})$.
4. $\mathcal{B}$ verifies the proof using the verification algorithm of the Adh system. The verification is performed by checking the equivariance condition: $\text{ZKVolute}(\mathbf{pk}, \mathbf{sig}'_{\text{chal}}\mathbf{sig}^{)}_{\text{rand}} = \text{ZKVolute}(\mathbf{sig}^*, \mathbf{pk}_{\text{chal}}, \mathbf{pk}_{\text{rand}})$. This condition ensures that only the party possessing the secret key $\mathbf{sk}$ can generate a valid proof that morphs the challenge and randomness in the same way as the public key was generated. The equivariance property is based on the associativity and commutativity of the ZKVolute function, which is a lossy hash function that destroys information while preserving the ability to verify the proof of possession.
5. If the proof is accepted, $\mathcal{B}$ computes $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$, where $\mathbf{sig}$ is a valid proof generated by $\mathcal{B}$ using the secret key $\mathbf{sk}$, a challenge $\mathbf{sig}_{\text{chal}}$ derived from the message m, and a randomly selected value $\mathbf{sig}_{\text{rand}}$.
6. If $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||\infty \leq 2\beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the Module-ISIS instance. The condition $||\mathbf{z}||\infty \leq 2\beta$ ensures that $\mathbf{z}$ is a valid solution to the Module-ISIS problem, as the Adh system's rejection sampling guarantees that all vectors have non-zero coefficients bounded by β.

To analyze the success probability of $\mathcal{B}$, we observe that if $\mathcal{A}$ succeeds in convincing the verifier of a false statement with non-negligible probability, then the forged proof $(\mathbf{sig}'\mathbf{sig}'_{\text{chal}}\mathbf{sig}^*_{\text{rand}})$ must satisfy the verification equation. The reduction works as follows: If an adversary $\mathcal{A}$ can forge a valid proof in the Adh system with non-negligible probability, then $\mathcal{B}$ can use $\mathcal{A}$ to solve the Module-ISIS problem. By setting up the public parameters and the public key using the Module-ISIS instance, $\mathcal{B}$ ensures that a forged proof that passes verification corresponds to a solution to the Module-ISIS problem. The difference between the forged proof and a valid proof generated by $\mathcal{B}$ yields a vector $\mathbf{z}$ that satisfies the Module-ISIS conditions. In summary, the key steps of the reduction are:

Setting up the Adh system using the Module-ISIS instance parameters. Generating the public key using randomly selected values. Obtaining a forged proof from the adversary $\mathcal{A}$. Verifying the forged proof using the equivariance condition. Computing the difference between the forged proof and a valid proof to obtain a solution to the Module-ISIS problem.

If the Adh system is not sound, then an adversary $\mathcal{A}$ can forge proofs with non-negligible probability, implying that the Module-ISIS problem can be solved with non-negligible probability by $\mathcal{B}$. This contradicts the assumed hardness of the Module-ISIS problem, proving that the Adh system is sound. □

A.6 Reduction to Module-ISIS

Theorem 16 (Reduction to Module-ISIS)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the Module-ISIS problem with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can forge a valid proof in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Module-ISIS problem. Given a Module-ISIS instance $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$, $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ sets up the public parameters of the Adh system using the Module-ISIS instance. It sets the modulus to q, the dimension to n, the rank to m, and the norm bound

to β.

2. $\mathcal{B}$ generates the public key **pk** by computing $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk}\_\text{chal}, \mathbf{pk}\_\text{rand})$, where **sk** is a randomly generated secret key, **pk**_chal is the public challenge, and **pk**_rand is the public randomness. $\mathcal{B}$ sets $\mathbf{sk} = \mathbf{A}$ and $\mathbf{pk}\_\text{chal} = \mathbf{t}$. $\mathcal{B}$ sends **pk** to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a forged proof $(\mathbf{sig}'\mathbf{sig}\_\text{chal}'\mathbf{sig}\_\text{rand}^{)}$.
4. $\mathcal{B}$ verifies the forged proof using the verification algorithm of the Adh system. If the proof is accepted, $\mathcal{B}$ proceeds to the next step. Otherwise, $\mathcal{B}$ aborts.
5. $\mathcal{B}$ computes $\mathbf{z} = \mathbf{sig}^{-}\mathbf{sig}$, where **sig** is a valid proof generated by $\mathcal{B}$ using the secret key **sk**.
6. If $\mathbf{z} \neq \mathbf{0}$ and $||\mathbf{z}||\_\infty \leq 2\beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the Module-ISIS instance.

The analysis of the success probability of $\mathcal{B}$ follows similarly to the reduction to Module-ISIS+ in Appendix A.2. If $\mathcal{A}$ succeeds in forging a valid proof with non-negligible probability, then $\mathbf{z}$ satisfies $\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \bmod q$ and $||\mathbf{z}||\_\infty \leq 2\beta$, solving the Module-ISIS instance. The success probability of $\mathcal{B}$ is equal to the success probability of $\mathcal{A}$, which is assumed to be non-negligible. Therefore, if the Adh system is susceptible to forgery attacks, then Module-ISIS is solvable with non-negligible probability, contradicting the assumed hardness of Module-ISIS. □

A.7 Reduction to Dense Subset Sum - Quantum Hardness

Theorem 17 (Reduction to Dense Subset Sum)**.** *If there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that can solve the modified Module-ISIS problem with addition in the Adh system with non-negligible probability, then there exists a probabilistic polynomial-time algorithm $\mathcal{B}$ that can solve the dense subset sum problem with density above 0.9408[9] with non-negligible probability.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can solve the modified Module-ISIS problem with addition in the Adh system with non-negligible probability. We construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the dense subset sum problem. Given a dense subset sum instance $(\mathbf{S}, t)$ with density above 0.94, where $\mathbf{S} = s_1, \ldots, s_n$ is a set of positive integers and t is a target sum, $\mathcal{B}$ proceeds as follows: $\mathcal{B}$ constructs a modified Module-ISIS instance $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$ as follows: Set $n = 128$ and $m = 6$ according to the Adh system parameters. Construct a diagonal matrix $\mathbf{A} = \text{diag}(s_1, \ldots, s_n) \in \mathbb{Z}_q^{n \times n}$, where the elements of $\mathbf{S}$ are placed on the main diagonal. Construct a target vector $\mathbf{t} = (t, 0, \ldots, 0) \in \mathbb{Z}_q^n$, where the first element is the target sum t and the remaining elements are zeros. Choose the modulus q and the norm bound β according to the Adh system parameters. $\mathcal{B}$ invokes the adversary $\mathcal{A}$ on the modified Module-ISIS instance $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$. If $\mathcal{A}$ outputs a solution vector $\mathbf{z} = (z_1, \ldots, z_n) \in \mathbb{Z}q^n$ such that $\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \bmod q$ and $||\mathbf{z}||\infty \leq \beta$, then $\mathcal{B}$ outputs $\mathbf{z}$ as a solution to the dense subset sum instance. The correctness of the reduction relies on the following observations: The diagonal matrix $\mathbf{A}$ constructed by $\mathcal{B}$ preserves the density of the original dense subset sum instance. Since the elements of $\mathbf{S}$ are placed on the main diagonal of $\mathbf{A}$, the resulting lattice has a density above 0.9408[9], mirroring the density of the subset sum instance. If the adversary $\mathcal{A}$ successfully solves the modified Module-ISIS instance, the solution vector $\mathbf{z}$ satisfies $\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \bmod q$. Expanding this equation, we have:

$$\begin{pmatrix} s_1 & \ddots & s_n \end{pmatrix} \begin{pmatrix} z_1 \vdots z_n \end{pmatrix} \begin{pmatrix} t \; 0 \vdots 0 \end{pmatrix} \bmod q$$

This implies that $\sum_{i=1}^{n} s_i \cdot z_i = t \bmod q$, which corresponds to a valid solution for the dense subset sum instance. Therefore, if an adversary can solve the modified Module-ISIS problem with addition in the Adh system with non-negligible probability, it would imply the existence of an efficient algorithm for solving the dense subset sum problem, contradicting the assumption that dense subset sum is computationally infeasible for density above 0.9408[9]. $\square$

A.8 Quantum Hardness Estimation

The reduction to the dense subset sum problem allows us to provide quantum hardness estimates for the Adh system. We consider two instances of the system, one with $n = 128$ and $m = 6$, and another with $n = 256$ and $m = 6$. According to the improved classical and quantum algorithms for the subset sum problem, as presented by Bonnetain et al. [3], the quantum hardness of the subset sum problem with k elements is estimated to be $2^{0.216k}$.

A.8.1 Instance 1: $n = 128$ and $m = 6$

In the case of an $n = 128$ and $m = 6$ module system, the total number of elements in the subset sum instance is:

$$\begin{aligned} k &= n \cdot m \\ &= 128 \cdot 6 \\ &= 768 \end{aligned}$$

Applying the quantum hardness estimate to this instance, we have:

$$\begin{aligned} \text{Quantum Hardness} &= 2^{0.216 \cdot k} \\ &= 2^{0.216 \cdot 768} \\ &\approx 2^{165.888} \end{aligned}$$

Therefore, the quantum hardness of the Adh system with $n = 128$ and $m = 6$ is estimated to be approximately 2^{166}.

A.8.2 Instance 2: $n = 256$ and $m = 6$

In the case of an $n = 256$ and $m = 6$ module system, the total number of elements in the subset sum instance is:

$$\begin{aligned} k &= n \cdot m \\ &= 256 \cdot 6 \\ &= 1536 \end{aligned}$$

Applying the quantum hardness estimate to this instance, we have:

$$\begin{aligned}\text{Quantum Hardness} &= 2^{0.216\cdot k}\\ &= 2^{0.216\cdot 1536}\\ &\approx 2^{331.776}\end{aligned}$$

Therefore, the quantum hardness of the Adh system with $n = 256$ and $m = 6$ is estimated to be approximately 2^{332}. These quantum hardness estimates, based on the improved algorithms by Bonnetain et al., provide an up-to-date assessment of the Adh system's resistance against quantum attacks. The estimates suggest that solving the dense subset sum problem corresponding to the Adh system instances would require a significant amount of quantum resources, even with the current best-known quantum algorithms.

A.9 Density Preservation in Module-ISIS to Module Modulus Subset Sum Reduction

Lemma 3. *Let $n = 128$, $rank = 6$, $\inf norm = 257$, and $p = 257$. Consider a module-ISIS problem with a rejection filter regime that discards all vectors containing any 0s and retries until a complete system is obtained. Changing the root of unity ω from 3 to 1 in the NTT is equivalent to relaxing the problem to addition. Under these conditions, the module-ISIS problem reduces to a module modulus subset sum problem with $128 \times 6 = 768$ elements, preserving the density.*

Proof. In the module-ISIS problem, we have a rank 6 lattice with 6 public vectors $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_6)$, where each vector $\mathbf{a}_i \in \mathbb{Z}_q^n$ and $q = 257$. The goal is to find a vector $\mathbf{t} \in \mathbb{Z}_q^n$ such that $\mathbf{t} = \mathbf{Az} \bmod q$ for some coefficient vector $\mathbf{z} \in \mathbb{Z}_q^{rank}$. By applying the rejection filter regime, we ensure that all vectors in the lattice have no 0 components, maintaining a dense structure. The density of the lattice is preserved during this process. When we change the root of unity ω from 3 to 1 in the NTT, the modular multiplication in the lattice is relaxed to addition. This relaxation does not affect the density of the lattice, as the structure and the number of elements remain unchanged.

The module-ISIS problem with $\omega = 1$ can be viewed as a module modulus subset sum problem. Each coefficient bucket in the NTT corresponds to an element in the subset sum problem. Since we have $n = 128$ and $rank = 6$, the total number of elements in the subset sum problem is $128 \times 6 = 768$. Let $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}2, \ldots, \mathbf{s}768)$ be the set of elements in the subset sum problem, where each $\mathbf{s}_i \in \mathbb{Z}_q$. The goal is to find a subset of $\mathbf{S}$ that sums to the target vector $\mathbf{t}$ modulo q.

The density of the subset sum problem is determined by the ratio of the number of elements to the modulus q. In this case, the density is 1, which is the same as the density of the original module-ISIS problem. Therefore, changing the root of unity from 3 to 1 in the NTT and applying the rejection filter regime reduces the module-ISIS problem to a module modulus subset sum problem with 768 elements while preserving the density. $\square$

A.10 Zero-Knowledge Proof

Theorem 18 (Zero-Knowledge Property)**.** *The Adh zero-knowledge proof system satisfies the zero-knowledge property, assuming the hardness of the Module-ISIS problem and the existence of a secure commitment scheme.*

Proof. We construct a simulator $\mathcal{S}$ that generates proofs indistinguishable from real proofs without access to the secret key. Given a public key $\mathbf{pk}$ and a statement to be proved, $\mathcal{S}$ proceeds as follows:

1. $\mathcal{S}$ generates a random commitment $\mathbf{com}$ using the commitment scheme.
2. $\mathcal{S}$ computes the challenge $\mathbf{sig}\_\text{chal}$ as a function of the statement and $\mathbf{com}$ using the Fiat-Shamir heuristic.
3. $\mathcal{S}$ samples a random vector $\mathbf{sig}\_\text{rand}$ and computes the proof $\mathbf{sig}$ as $\mathbf{sig} = \text{ZKVolute}(\mathbf{pk}, \mathbf{sig}\_\text{chal}, \mathbf{sig}\_\text{rand})$.
4. $\mathcal{S}$ outputs the proof $(\mathbf{sig}, \mathbf{sig}\_\text{chal}, \mathbf{sig}\_\text{rand})$.

To show that the simulated proofs are indistinguishable from real proofs, we consider the following hybrid arguments:

- Hybrid 1: Real proofs generated using the secret key.
- Hybrid 2: Proofs generated using the simulator $\mathcal{S}$.

The indistinguishability of Hybrid 1 and Hybrid 2 relies on the following arguments:

- The commitment scheme is hiding, ensuring that $\mathbf{com}$ does not reveal any information about the secret key.
- The Fiat-Shamir heuristic ensures that the challenge $\mathbf{sig}\_\text{chal}$ is uniformly distributed and independent of the secret key.
- The ZKVolute function is a one-way function, assuming the hardness of the Module-ISIS problem. Given $\mathbf{pk}$, $\mathbf{sig}\_\text{chal}$, and $\mathbf{sig}\_\text{rand}$, it is computationally infeasible to recover the secret key.

Therefore, the proofs generated by the simulator $\mathcal{S}$ are computationally indistinguishable from real proofs, establishing the zero-knowledge property of the Adh system. □

A.11 Algorithms

A.12 Notes

Algorithm variable notes:
H is a hash function in the SHA3 family
m is a theoretical message to be signed
n is dimension
p is the first level of NTT modulus
ω is the first level of NTT root of unity
k is the number of instances of module-ISIS problem to create
ps is the array of NTT moduli in a multi stage instance
ws is the array of related roots of unity
l is number of 'levels' of unique NTT stage or len(ps)
NTT_DIST is the number of NTT representations to check before abort

A.13 Module-ISIS+ Parameters

n=128 or 256
ps=[257,257]
ws=[3,3]
rnds=4
sk_count=1
iters=4

$\beta = 256$
rank = 6

A.14 Module-ISIS* Parameters

n=128 or n=256
ps=[257,257]
ws=[3,3]
rnds=4
sk_count=5
iters=4
$\beta = 256$
rank = 6

A.15 Module-ISIS** Parameters

n=128 or n=256
ps=[257,257,65537]
ws=[3,3,282]
rnds=4
sk_count=5
iters=4
$\beta = 256$
rank = 6

A.16 Algorithms

Algorithm 4 Expand Hash Function

Designed for XOF hash_algorithm=SHAKE256 and $\beta = 256$

```
Require: message, prime, size, hash_algorithm
Ensure: coefficients
    orig_message ← message
    while True do
        hash_object ← hash_algorithm(message)
        hash_value ← hash_object.digest(size)
        hash_int ← int.from_bytes(hash_value, byteorder =' big')
        coefficients ← []
        for i ← 0 to size − 1 do
            coefficients.append(int(hash_value[i]))                ▷ 0-255
        end for
        if poly_check(coefficients) = 0 then
            return coefficients
        end if
        message ← orig_message||str(hash_value)                    ▷ To keep input to 2x
    end while
```

Key Generation (strong_generate_keys_isis_star):

Algorithm 5 Blind Value Computation

This function is used to randomize the random value by adding it to the sum context values to reduce an adversaries ability to influence the computation.

Require: $context\_values$, $modulus$
Ensure: $blinded\_value$

    $blinded\_value \leftarrow []$
    for vec in $context\_values$ **do**
        $blinded\_value \leftarrow pointwise\_add(blinded\_value, vec, modulus)$
    end for
    return $blinded\_value$

Algorithm 6 select_representation

Require: vec, p, w
Ensure: $best\_vec$

$input\_key \leftarrow tuple(vec), p, w$
$best \leftarrow vec.count(0)$
$best\_vec \leftarrow vec.copy()$
$count \leftarrow 0$
for $i \leftarrow 0$ to NTT_DIST **do** ▷ 2 for $p = 257$ $\omega = 3$
    $vec \leftarrow ntt(vec, p, w)$
    if $vec.count(0) \leq best$ **then**
        $best\_vec \leftarrow vec.copy()$
    end if
    if $vec.count(0) = 0$ **then**
        return $(vec, count)$ ▷ Found suitable vector
    end if
    $count \leftarrow count + 1$
end for
return $(best\_vec, count)$ ▷ Could not find full vector, next best

Algorithm 7 Polynomial Support Check - polycheck

Ensure NTT representation of a full vector is full across each configured level

Require: $poly$, $moduli$, $unity\_roots$
Ensure: $support$

    $support \leftarrow poly.count(0)$
    for $i \leftarrow 1$ to $length(moduli)$ **do**
        $p \leftarrow moduli[i]$
        $w \leftarrow unity\_roots[i]$
        $poly, c \leftarrow select\_representation(poly, p, w)$
        $support \leftarrow support + poly.count(0)$
    end for
    return $support$

Algorithm 8 Key Generation with Rejection Sampling(Module-ISIS+)

Require: n, $base\_modulus$, $ZKVolute\_ProofGen$, $poly\_check$, $generate\_non_zero\_vector$
Ensure: pk_a, sk_I, pk_chal, $rand\_pk$

    $support \leftarrow 1$
    while $support \neq 0$ or $pk\_a.count(0) \neq 0$ or $poly\_check(pk\_a) \neq 0$ **do**
        $pk\_chal \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
        while $poly\_check(pk\_chal) \neq 0$ **do**
            $pk\_chal \leftarrow generate\_non_zero\_vector(n, base\_modulus)$
        end while
        $sk\_I \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
        while $poly_check(sk\_I) \neq 0$ **do**
            $sk\_I \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
        end while
        $rand\_pk \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
        while $poly\_check(rand\_pk) \neq 0$ **do**
            $rand\_pk \leftarrow generate_non_zero_vector(n, base\_modulus)$
        end while
        $pk\_a, support \leftarrow ZKVolute\_ProofGen(sk\_I, rand\_pk, pk\_chal)$
    end while
    return pk_a, sk_I, pk_chal, $rand\_pk$

Algorithm 9 Key Generation (Module-ISIS*)

Require: n, $base\_modulus$, $ZKVolute\_ProofGen\_isis\_star$, $poly\_check$, $generate\_non\_zero\_vector$, $rnds$
Ensure: pk_a, sk_array, pk_chal, $rand\_pk$
  $support \leftarrow 1$
  $sk\_array \leftarrow []$
  while $support \neq 0$ or $pk\_a.count(0) \neq 0$ or $poly\_check(pk\_a) \neq 0$ **do**
    $pk\_chal \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
    while $poly\_check(pk\_chal) \neq 0$ **do**
      $pk\_chal \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
    end while
    for $\_ \leftarrow 0$ to $rnds$ **do**
      $sk\_i \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
      while $poly\_check(sk\_i) \neq 0$ **do**
        $sk\_i \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
      end while
      $sk\_array.append(sk\_i)$
    end for
    $rand\_pk \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
    while $poly\_check(rand\_pk) \neq 0$ **do**
      $rand\_pk \leftarrow generate\_non\_zero\_vector(n, base\_modulus)$
    end while
    $pk\_a, support \leftarrow ZKVolute\_ProofGen\_isis\_star(sk\_array, rand\_pk, pk\_chal)$
  end while
  return pk_a, sk_array, pk_chal, $rand\_pk$

Algorithm 10 Core Proof Generation

Require: m, sk_I, n, $base\_modulus$, $Hash\_To\_Poly$, $ZKVolute\_ProofGen$, $polycheck$, $generate\_full\_\_vector$
Ensure: SIG
  $challenge\_vector \leftarrow Hash\_To\_Poly(m)$
  $rand\_sig \leftarrow generate\_full\_{v}ector(n, base\_modulus)$
  while $rand\_sig.count(0) \neq 0$ and $polycheck(rand\_sig) \neq 0$ **do**
    $rand\_sig \leftarrow generate\_full\_\_vector(n, base\_modulus)$
  end while
  $SIG \leftarrow ZKVolute\_ProofGen(sk\_I, rand\_sig, challenge\_vector, False)$
  while $SIG.count(0) \neq 0$ **do**
    $rand\_sig \leftarrow generate\_full\_\_vector(n, base\_modulus)$
    $SIG \leftarrow ZKVolute\_ProofGen(sk\_I, rand\_sig, challenge\_vector, False)$
  end while
  return SIG

Algorithm 11 ZKVolute_ProofVerify

```
Require: PROOF_PK, pk_chal, PROOF_SIG, sig_chal, rand_pk, rand_sig
Ensure: result
    p ← base_moduli, w ← base_root
    sig_chal, _ ← select_representation(sig_chal, p, w)
    pk_chal, _ ← select_representation(pk_chal, p, w)
    rand_pk, _ ← select_representation(rand_pk, p, w)
    rand_sig, _ ← select_representation(rand_sig, p, w)
    pk_orig ← pk_chal, sig_orig ← sig_chal
    pk_iterables ← list(), sig_iterables ← list()
    pk_iterable ← pk_chal, sig_iterable ← sig_chal
    for _ ← 0 to iters − 1 do
        pk_iterable, _ ← select_representation(pk_iterable, p, w)
        sig_iterable, _ ← select_representation(sig_iterable, p, w)
        pk_iterables.append(pk_iterable), sig_iterables.append(sig_iterable)
    end for
    for it ← 0 to iters − 1 do
        if it = 0 then
            pk_chal2 ← pointwise_mul(pk_chal, pk_iterables[0], p)
            sig_chal2 ← pointwise_mul(sig_chal, sig_iterables[0], p)
        else
            pk_chal2 ← pointwise_mul(pk_chal2, pk_iterables[it], p)
            sig_chal2 ← pointwise_mul(sig_chal2, sig_iterables[it], p)
        end if
    end for
    if iters > 0 then
        pk_chal ← pk_chal2, sig_chal ← sig_chal2
    end if
    new_sig_chal ← pointwise_mul(sig_orig, rand_sig, p)
    new_sig_chal ← pointwise_add(new_sig_chal, sig_chal, p)
    new_sig_chal ← pointwise_add(new_sig_chal, rand_sig, p)
    new_pk_chal ← pointwise_mul(pk_orig, rand_pk, p)
    new_pk_chal ← pointwise_add(new_pk_chal, pk_chal, p)
    new_pk_chal ← pointwise_add(new_pk_chal, rand_pk, p)
    chk_rep1 ← pointwise_mul(sig_chal, PROOF_PK, p)
    chk_rep2 ← pointwise_mul(pk_chal, PROOF_SIG, p)
    chk_rep1 ← pointwise_mul(chk_rep1, rand_sig, p)
    chk_rep2 ← pointwise_mul(chk_rep2, rand_pk, p)
    for i1 ← 0 to rnds − 1 do
        chk_rep1 ← pointwise_mul(chk_rep1, sig_orig, p)
        chk_rep1 ← pointwise_mul(chk_rep1, new_sig_chal, p)
        chk_rep1 ← pointwise_mul(chk_rep1, sig_iterables[i1 mod iters], p)
        new_sig_chal ← pointwise_add(new_sig_chal, new_sig_chal, p)
        new_sig_chal ← pointwise_mul(new_sig_chal, new_sig_chal, p)
    end for
    for i2 ← 0 to rnds − 1 do
        chk_rep2 ← pointwise_mul(chk_rep2, pk_orig, p)
        chk_rep2 ← pointwise_mul(chk_rep2, new_pk_chal, p)
        chk_rep2 ← pointwise_mul(chk_rep2, pk_iterables[i2 mod iters], p)
        new_pk_chal ← pointwise_add(new_pk_chal, new_pk_chal, p)
        new_pk_chal ← pointwise_mul(new_pk_chal, new_pk_chal, p)
    end for
    result ← (chk_rep1 = chk_rep2)
    return result
```

Algorithm 12 ZKVolute_ProofGen (Module-ISIS+)

```
Require: sk_I, rand_chal, chal
Ensure: proof_rep
  for i, (p, w) in enumerate(list(zip(ps, ws))) do
    sk_rep_I ← select_representation(sk_I, p, w)
    rand_chal ← select_representation(rand_chal, p, w)
    chal ← select_representation(chal, p, w)
    iterables ← list()
    ntt_rep ← chal
    blinded_values ← list()
    root_chal ← chal
    blinded_values.append(root_chal)
    if iters > 0 then
      for _ ← 0 to iters − 1 do
        ntt_rep ← select_representation(ntt_rep, p, w)
        blinded_values.append(ntt_rep)
        iterables.append(ntt_rep)
      end for
      for z ← 1 to iters − 1 do
        ntt_rep ← pointwise_mul(ntt_rep, iterables[z], p)
        blinded_values.append(ntt_rep)
      end for
      chal ← ntt_rep
    end if
    if i = 0 then
      secret_rep ← sk_I
      target_vector ← pointwise_mul(chal, secret_rep, p)
      proof_rep ← pointwise_mul(target_vector, rand_chal, p)
      new_chal ← pointwise_mul(root_chal, rand_chal, p)
      new_chal ← pointwise_add(new_chal, chal, p)
      new_chal ← pointwise_add(new_chal, rand_chal, p)
      for xx ← 0 to rnds − 1 do
        proof_rep ← pointwise_mul(proof_rep, root_chal, p)
        proof_rep ← pointwise_mul(proof_rep, new_chal, p)
        proof_rep ← pointwise_mul(proof_rep, iterables[xx%iters], p)
        new_chal ← pointwise_mul(new_chal, new_chal, p)
        new_chal ← pointwise_add(new_chal, new_chal, ps[i])
      end for
      proof_hld ← proof_rep
    end if
    if i ≥ 1 then
      proof_rep ← ntt(proof_rep, p, w)
      proof_hld ← ntt(proof_hld, p, w)
      proof_rep ← pointwise_add(proof_rep, proof_rep, p)
      proof_rep ← pointwise_add(proof_rep, proof_hld, p)
    end if
  end for
  for i, (p, w) in enumerate(reversed(list(zip(ps, ws)))) do
    if i < len(ps) − 1 then
      proof_rep ← ntt_inverse(proof_rep, p, w, original_n = n)
    end if
  end for
  return proof_rep
```

Algorithm 13 ZKVolute Proof Generation (Module-ISIS*)

Require: sk_array, $rand\_chal$, $chal$, ps, ws, $iters$, $rnds$, $pointwise\_mul$, $pointwise\_addition$, ntt, $ntt\_inverse$, $best\_ntt$
Ensure: $proof\_rep$

```
for i, (p, w) in enumerate(list(zip(ps, ws))) do
    sk_rep_array ← [best_ntt(sk, p, w)[0] for sk in sk_array]
    rand_chal, _ ← best_ntt(rand_chal, p, w)
    chal, _ ← best_ntt(chal, p, w)
    iterables ← list()
    ntt_rep ← chal.copy()
    blinded_values ← list()
    root_chal ← chal.copy()
    blinded_values.append(root_chal)
    tmp_iterable ← chal.copy()
    if iters > 0 then
        for _ ← 0 to iters − 1 do
            tmp_iterable, _ ← best_ntt(tmp_iterable, p, w)
            blinded_values.append(tmp_iterable)
            iterables.append(tmp_iterable)
        end for
        for z ← 0 to iters − 1 do
            if z = 0 then
                tmp_iterable2 ← pointwise_mul(root_chal, iterables[0], p)
                blinded_values.append(tmp_iterable2)
            else
                tmp_iterable2 ← pointwise_mul(tmp_iterable2, iterables[z], p)
                blinded_values.append(tmp_iterable2)
            end if
        end for
        chal ← tmp_iterable2
        blinded_values.append(chal)
    end if
    if i = 0 then
        secret_rep ← sk_rep_array[0]
        target_vector ← pointwise_mul(chal, secret_rep, p)
        proof_rep ← pointwise_mul(target_vector, rand_chal, p)
        new_chal ← pointwise_mul(root_chal, rand_chal, p)
        new_chal ← pointwise_addition(new_chal, chal, p)
        new_chal ← pointwise_addition(new_chal, rand_chal, p)
        for xx ← 0 to rnds − 1 do
            proof_rep ← pointwise_mul(proof_rep, sk_rep_array[xx + 1], p)
            proof_rep ← pointwise_mul(proof_rep, root_chal, p)
            proof_rep ← pointwise_mul(proof_rep, new_chal, p)
            proof_rep ← pointwise_mul(proof_rep, iterables[xx mod iters], p)
            new_chal ← pointwise_mul(new_chal, new_chal, p)
            new_chal ← pointwise_addition(new_chal, new_chal, p)
        end for
        proof_hld ← proof_rep
    end if
    if i ≥ 1 then
        proof_rep ← ntt(proof_rep, p, w)
        proof_hld ← ntt(proof_hld, p, w)
        proof_rep ← pointwise_addition(proof_rep, proof_rep, p)
        proof_rep ← pointwise_addition(proof_rep, proof_hld, p)
    end if
end for
for i, (p, w) in enumerate(reversed(list(zip(ps, ws)))) do
    if i < len(ps) − 1 then
        proof_rep ← ntt_inverse(proof_rep, p, w, original_n = n)
    end if
end for
return proof_rep
```

A.17 Empirical Evidence for Zero-Knowledge Property

The zero-knowledge property ensures that a proof generated by the Adh system does not reveal any information about the secret key, except for the validity of the statement being proven. We present empirical evidence supporting the zero-knowledge property of the Adh system using a comprehensive simulator-based approach and rigorous statistical testing[6].

A.17.1 Simulator-based Approach

We constructed a simulator to generate a large number of real proofs (using genuine secret keys) and fake proofs (using randomly generated or slightly perturbed keys). The simulator ensures that the fake proofs are generated in a way that mimics the behavior of real proofs, including the use of the same challenges and random values. The simulator also adjusts the random challenges to ensure that both real and fake proofs can be generated successfully, maintaining the indistinguishability between them. The simulator follows these key steps:

1. Generate a valid key pair (public key and secret keys) for the Adh system, rejecting any keys that contain zero coefficients to prevent the exposure of internal patterns.
2. Create a Fiat-Shamir style challenge by generating a random vector and ensuring it meets the non-zero constraint.
3. Generate a real proof using the genuine secret keys, the challenge, and a random blinding vector.
4. Generate a fake proof using slightly perturbed or randomly generated secret keys, the same challenge, and the same random blinding vector.
5. Verify both the real and fake proofs using the Adh verification algorithm, ensuring that the real proof is accepted and the fake proof is rejected.
6. Store the real and fake proofs for statistical analysis.

The simulator was run for a large number of iterations (at least 300 million proof pairs) to collect a significant sample size for statistical testing. Throughout the simulations, no real proofs failed verification, and no fake proofs were accepted, providing strong empirical evidence for the soundness and forgery resistance of the Adh system.

A.17.2 Statistical Tests

To assess the indistinguishability of real and fake proofs, we performed a comprehensive suite of statistical tests on the collected data. These tests evaluate various properties of the proof distributions, such as means, standard deviations, correlations, and statistical distances. The following tests were conducted:

- Chi-squared test
- Kolmogorov-Smirnov test
- Anderson-Darling test
- Mann-Whitney U test
- Kruskal-Wallis test
- Shapiro-Wilk test
- Pearson correlation test
- Mutual information test
- Autocorrelation test
- Higher-order moments test

The tests were applied to the real and fake proof distributions, and the results were analyzed to determine if there were any statistically significant differences between them. Across millions of runs and various configurations, the statistical tests consistently demonstrated the indistinguishability of real and fake proofs.

The p-values obtained from the tests were consistently above the significance threshold (e.g., 0.05), indicating that the null hypothesis (i.e., the distributions of real and fake proofs are the same) cannot be rejected. The correlation coefficients between real and fake proofs were close to zero, suggesting no significant correlation between them. The mutual information between real and fake proofs was negligible, indicating minimal shared information. The higher-order moments and autocorrelation tests further supported the randomness and independence of the proofs.

A.17.3 Machine Learning Test

To further assess the distinguishability of real and fake proofs, we applied a gradient boosting classifier to the proof data. The classifier was trained on a subset of the real and fake proofs and then tested on a held-out set to evaluate its ability to distinguish between them. Across multiple runs, the classifier consistently achieved an accuracy close to 50%, indicating that it was unable to distinguish between real and fake proofs better than random guessing. This result provides additional evidence for the zero-knowledge property of the Adh system, as even advanced machine learning algorithms were unable to differentiate between the two types of proofs.

A.17.4 Empirical Conclusion

The empirical evidence obtained from the simulator-based approach and the statistical tests provides compelling support for the presence of the zero-knowledge property in the Adh system. The extensive testing, covering a wide range of configurations and a large number of proofs, demonstrates the consistent indistinguishability of real and fake proofs. The inability to forge valid proofs and the resistance to advanced distinguishing techniques further strengthen the case for the zero-knowledge property.

While a formal mathematical proof of the zero-knowledge property is still pending, the empirical results obtained from this rigorous experimental setup strongly suggest that the Adh system achieves zero-knowledge. The simulator-based approach, combined with comprehensive statistical testing and machine learning analysis, provides a robust framework for assessing the zero-knowledge property and lays the foundation for further theoretical analysis and formal proofs. The detailed experimental setup and results supporting the zero-knowledge property are provided here.

A.18 Zero-Knowledge Proof

Theorem 19 (Zero-Knowledge Property)**.** *The Adh zero-knowledge proof system satisfies the zero-knowledge property, assuming the hardness of the Module-ISIS problem and the existence of a secure commitment scheme.*

Proof. We construct a simulator $\mathcal{S}$ that generates proofs indistinguishable from real proofs without access to the secret key. Given a public key $\mathbf{pk}$ and a statement to be proved, $\mathcal{S}$ proceeds as follows:

1. $\mathcal{S}$ generates a random commitment $\mathbf{com}$ using the commitment scheme.

2. $\mathcal{S}$ computes the challenge $\mathbf{sig}\_\mathrm{chal}$ as a function of the statement and $\mathbf{com}$ using the Fiat-Shamir heuristic.
3. $\mathcal{S}$ samples a random vector $\mathbf{sig}\_\mathrm{rand}$ and computes the proof $\mathbf{sig}$ as:

$$\mathbf{sig} = \mathrm{ZKVolute}(\mathbf{pk}, \mathbf{sig}\_\mathrm{chal}, \mathbf{sig}\_\mathrm{rand})$$

4. $\mathcal{S}$ outputs the proof $(\mathbf{sig}, \mathbf{sig}\_\mathrm{chal}, \mathbf{sig}\_\mathrm{rand})$.

To show that the simulated proofs are indistinguishable from real proofs, we consider the following hybrid arguments:

- Hybrid 1: Real proofs generated using the secret key.
- Hybrid 2: Proofs generated using the simulator $\mathcal{S}$.

We now argue that Hybrid 1 and Hybrid 2 are computationally indistinguishable based on the following:

- The commitment scheme is computationally hiding, ensuring that **com** does not reveal any information about the secret key to a computationally bounded adversary.
- The Fiat-Shamir heuristic ensures that the challenge $\mathbf{sig}\_\mathrm{chal}$ is uniformly distributed and independent of the secret key, assuming the random oracle model.
- The ZKVolute function is a one-way function, assuming the hardness of the Module-ISIS problem. Given $\mathbf{pk}$, $\mathbf{sig}\_\mathrm{chal}$, and $\mathbf{sig}\_\mathrm{rand}$, it is computationally infeasible to recover the secret key.

Suppose there exists a polynomial-time distinguisher $\mathcal{D}$ that can distinguish between Hybrid 1 and Hybrid 2 with non-negligible advantage. We construct a polynomial-time adversary $\mathcal{A}$ that uses $\mathcal{D}$ to break either the hiding property of the commitment scheme or the one-wayness of the ZKVolute function.

$\mathcal{A}$ receives a public key $\mathbf{pk}$ and a statement to be proved. It then generates two proofs, one using the real prover algorithm and one using the simulator $\mathcal{S}$. $\mathcal{A}$ sends the two proofs to the distinguisher $\mathcal{D}$. If $\mathcal{D}$ can distinguish between the real and simulated proofs with non-negligible advantage, then $\mathcal{A}$ can use this to break either the hiding property of the commitment scheme or the one-wayness of the ZKVolute function, depending on $\mathcal{D}$'s output. This contradicts the assumptions of a secure commitment scheme and the hardness of Module-ISIS. Therefore, the proofs generated by the simulator $\mathcal{S}$ are computationally indistinguishable from real proofs, establishing the zero-knowledge property of the Adh system. □

A.19 Probabilistic Completeness

Theorem 20 (Probabilistic Completeness). *Let $\mathcal{A}$ be the Adh zero-knowledge proof system with dimension n, norm bound β, and a fixed challenge vector $\mathbf{c}$. If the prover has a probability p of passing the rejection sampling step for a given random vector, then the probability of finding a valid proof for the fixed challenge $\mathbf{c}$ approaches 1 as the number of attempts grows exponentially with respect to n.*

Proof. Consider a scenario where the prover has a fixed challenge vector $\mathbf{c}$ and needs to generate a valid proof. The prover selects a random vector $\mathbf{r}$ of dimension n with coefficients bounded by the norm β. The prover then attempts to generate a proof by passing $\mathbf{r}$ through the rejection sampling step. Let p be the probability of the prover passing the rejection sampling step for a given random vector $\mathbf{r}$. If the prover fails the rejection sampling step, they simply select a new random vector and try again. The probability of failing to find a valid proof after k attempts is given by:

$$P(\text{failure after } k \text{ attempts}) = (1 - p)^k \tag{40}$$

As the number of attempts k grows, the probability of failure decreases exponentially. In the Adh system, the dimension n is typically chosen to be either 128 or 256, and the norm bound β is set to 257. For $n = 128$, the prover has 257^{128} possible random vectors to choose from. Even with a conservative probability of passing the rejection sampling step, say $p = 0.05$, the probability of failure after k attempts is:

$$P(\text{failure after } k \text{ attempts}) = (1 - 0.05)^k = 0.95^k \tag{41}$$

As k approaches 257^{128}, the probability of failure becomes negligibly small. Similarly, for $n = 256$, the prover has 257^{256} possible random vectors to choose from. With the same conservative probability $p = 0.05$, the probability of failure after k attempts is:

$$P(\text{failure after } k \text{ attempts}) = (1 - 0.05)^k = 0.95^k \tag{42}$$

As k approaches 257^{256}, the probability of failure becomes even smaller. Therefore, given the extremely large number of possible random vectors and the ability of the prover to repeatedly attempt rejection sampling, the probability of finding a valid proof for a fixed challenge vector approaches 1. While this argument does not provide an absolute proof of completeness, it demonstrates that the Adh system achieves a strong form of probabilistic completeness. The chances of the prover failing to find a valid proof for a given challenge are negligibly small, assuming a reasonable probability of passing the rejection sampling step. □

This probabilistic completeness argument highlights the effectiveness of the rejection sampling technique used in the Adh system. By allowing the prover to repeatedly select new random vectors until a valid proof is found, the system ensures that the prover can successfully generate proofs for any given challenge with overwhelming probability. The conservative estimate of a 5% success probability for each attempt further strengthens the argument, as the actual success probability in the Adh system is typically much higher (closer to 60% empercially). This means that the prover can find a valid proof with even fewer attempts in practice.

Rejection sampling is also applied during the challenge generation process on the hash of message as m. If m produces a *chal* that fails the rejection sampling test, m is first copied to a temporary variable h_val and a loop where $h\_val \leftarrow H(m||h\_val)$ is iterated with no maximum number of attempts. If the *chal* that is produced by h_val passes rejection sampling the loop terminates. As the number of attempts is essentially unbounded, this intuitive result is not formally proven under random oracle assumptions.

The completeness of the Adh system relies on the vast number of possible random vectors and the efficiency of the rejection sampling process. As the dimension n and the norm bound β increase, the probability of failure diminishes rapidly, providing a strong assurance of completeness. While this probabilistic argument may not constitute an absolute proof of completeness, it provides a compelling justification for the completeness property of the Adh system based on the overwhelming likelihood of success.

Conjecture 4 (Unlikelihood of Violating Shannon-Nyquist Sampling Theorem in the Adh System)**.** *The recent advancements in quantum algorithms for solving the Learning with Errors (LWE) problem, particularly the use of Gaussian functions with complex* *variances and the exploitation of the Karst wave feature in the Quantum Fourier Transform (QFT) domain, have raised concerns about the potential impact on the security of lattice-based cryptographic systems like the Adh zero-knowledge proof system.*

However, it is important to consider the fundamental principles of information theory, such as the Shannon-Nyquist[11] sampling theorem, when assessing the likelihood of a quantum computer being able to violate these principles in the context of the Adh system. The Shannon-Nyquist sampling theorem states that a signal can be perfectly reconstructed from its samples if the sampling rate is at least twice the highest frequency component in the signal. In the context of the Adh system, which employs the Number Theoretic Transform (NTT) for polynomial multiplication, the NTT can be viewed as a form of sampling in the frequency domain. Given the structure and parameters of the Adh system, it seems ***unlikely*** *that a quantum computer, even with the advanced techniques like the Karst wave, would be able to violate the Shannon-Nyquist sampling theorem and perfectly reconstruct the undersampled signal in the NTT domain. The reasons for this assessment are as follows:*

- *The Adh system operates over finite fields, and the NTT is a discrete transform that preserves the algebraic structure of the underlying ring. The sampling rate in the NTT domain is determined by the choice of parameters and the structure of the polynomial ring.*
- *The security of the Adh system relies on the hardness of the Module-ISIS problem, which is based on finding short integer solutions to linear equations. The problem is designed to be computationally infeasible, even for quantum computers, when the parameters are appropriately chosen.*
- *The use of rejection sampling and the careful selection of parameters in the Adh system ensure that the resulting lattices have a high dimension and a large minimum distance, making it difficult for any algorithm, including quantum algorithms, to find short vectors and solve the underlying Module-ISIS problem.*

While the Karst wave technique exploits certain periodic patterns in the QFT domain, it is not clear whether such patterns exist or can be efficiently exploited in the NTT domain of the Adh system. Furthermore, even if such patterns were found, it is unlikely that they would enable a quantum computer to violate the Shannon-Nyquist sampling theorem and perfectly reconstruct the undersampled signal.

In this updated conjecture, we emphasize the unlikelihood of a quantum computer being able to violate the Shannon-Nyquist sampling theorem in the context of the Adh system. We highlight the reasons behind this assessment, including the discrete nature of the NTT, the hardness of the underlying Module-ISIS problem, and the careful parameter selection and rejection sampling techniques used in the Adh system. However, we also acknowledge the rapid evolution of the field of quantum computing and the possibility of new techniques and insights emerging in the future. We stress the importance of maintaining a cautious approach, actively monitoring developments, and conducting regular security assessments to ensure the long-term security of the Adh system against potential quantum threats.

A.20 Proof of Completeness

Theorem 21 (Completeness)**.** *The Adh zero-knowledge proof system is complete. That is, an honest prover can always convince the verifier of a true statement.*

Proof. Let $(\mathbf{pk}, \mathbf{sk})$ be a valid key pair generated by the key generation algorithm of the Adh system, where $\mathbf{pk}$ is the public key and $\mathbf{sk}$ is the secret key. Let m be a message and $\mathbf{sig}$_chal be the signature challenge derived from m. An honest prover, possessing the secret key $\mathbf{sk}$, generates a proof $(\mathbf{sig}, \mathbf{sig}\text{\_chal}, \mathbf{sig}\text{\_rand})$ as follows:

1. Generate a uniformly random signature randomness $\mathbf{sig}$_rand $\in R\_q^m$ with coefficients in the range $[1, q-1]$.
2. Apply rejection sampling to ensure that $\mathbf{sig}$_rand is a full vector.
3. Compute the proof $\mathbf{sig}$ as $\mathbf{sig} = \text{ZKVolute}(\mathbf{sk}, \mathbf{sig}\text{\_chal}, \mathbf{sig}\text{\_rand})$.

The verifier checks the validity of the proof by computing:

$$\mathbf{lhs} = \text{ZKVolute}(\mathbf{pk}, \mathbf{sig}\text{\_chal}, \mathbf{sig}\text{\_rand})$$
$$\mathbf{rhs} = \text{ZKVolute}(\mathbf{sig}, \mathbf{pk}\text{\_chal}, \mathbf{pk}\text{\_rand})$$

and verifying that $\mathbf{lhs} = \mathbf{rhs}$. By the construction of the Adh system and the properties of the ZKVolute function, we have:

$$\begin{aligned}
\mathbf{lhs} &= \text{ZKVolute}(\mathbf{pk}, \mathbf{sig}\text{_chal}, \mathbf{sig}\text{_rand}) \\
&= \text{ZKVolute}(\text{ZKVolute}(\mathbf{sk}, \mathbf{pk}\text{_chal}, \mathbf{pk}\text{_rand}), \mathbf{sig}\text{_chal}, \mathbf{sig}\text{_rand}) \\
&= \text{ZKVolute}(\mathbf{sk}, \text{ZKVolute}(\mathbf{pk}\text{_chal}, \mathbf{sig}\text{_chal}, \mathbf{sig}\text{_rand}), \mathbf{pk}\text{_rand}) \\
&= \text{ZKVolute}(\mathbf{sk}, \mathbf{sig}\text{_chal}, \mathbf{sig}\text{_rand}) && = \mathbf{sig} \\
&= \text{ZKVolute}(\mathbf{sig}, \mathbf{pk}\text{_chal}, \mathbf{pk}\text{_rand}) && = \mathbf{rhs}
\end{aligned}$$

□

Therefore, an honest prover, possessing the secret key $\mathbf{sk}$, can always generate a valid proof that convinces the verifier, proving the completeness of the Adh zero-knowledge proof system

A.21 Conjecture on Entropy Expansion and Information Loss in Module-ISIS** with Higher-Dimensional NTT Mixing and Reduction

Conjecture 5 (Entropy Expansion and Information Loss in Module-ISIS** with Higher-Dimensional NTT Mixing and Reduction)**.** *Let $\mathcal{L}$ be an instance of the Module-ISIS** problem with a prime modulus $p_1 = 257$ (in a zero free regime) and a higher-dimensional prime modulus $p_2 = 65537$. Let $\mathbf{x} \in \mathbb{Z}{p_1}^n$ be a vector representing a proof in the Adh system, and let $H(\mathbf{x})$ denote the Shannon entropy of $\mathbf{x}$. Consider the following transformation:*

1. *Compute the NTT representation of $\mathbf{x}$ in the field $\mathbb{Z}p_1$, denoted as $\mathbf{X}1 = NTTp_1(\mathbf{x})$.*
2. *Forward transform $\mathbf{X}1$ to the field $\mathbb{Z}p_2$, denoted as $\mathbf{X}2 = NTTp_2(\mathbf{X}_1)$.*
3. *Perform modular addition of $\mathbf{X}2$ with itself in the field $\mathbb{Z}p_2$, denoted as $\mathbf{Y}2 = \mathbf{X}2 \oplus \mathbf{X}2$, where $\oplus$ represents element-wise modular addition.*
4. *Invert the NTT representation of $\mathbf{Y}2$ back to the field $\mathbb{Z}p_1$, denoted as $\mathbf{y} = INTTp_1(\mathbf{Y}2)$.*

We conjecture that the modular reduction from the higher-dimensional field $\mathbb{Z}p_2$ back to the original field $\mathbb{Z}p_1$ is the primary cause of the observed high entropy in the output vector $\mathbf{y}$. The fact that the Shannon entropy of $\mathbf{y}$ approaches a nearly perfect 8 bits per element, which is the maximum possible entropy for elements in $\mathbb{Z}257$ with a 257 norm, suggests that the modular reduction step may lead to a significant loss of structural information about the underlying lattice.

During the transformation process, the structural information of the lattice is expanded to extra dimensions in the higher-dimensional field $\mathbb{Z}_{p_2}$*. The modular addition of* $\mathbf{X}2$ *with itself further obfuscates the lattice structure by mixing and folding the information onto itself. When this expanded and obfuscated representation is then reduced back to the original field* $\mathbb{Z}p_1$*, a substantial amount of critical structural data needed for inversion may be randomly lost due to the modular reduction.*

The apparent loss of structural information during the modular reduction step could potentially preclude the inversion of the transformation altogether. If the entropy of the output vector $\mathbf{y}$ *approaches the maximum possible value, it suggests that the information content of* $\mathbf{y}$ *is nearly uniform and lacks any discernible structure. This loss of structure may make it infeasible to recover the original vector* $\mathbf{x}$ *from* $\mathbf{y}$*, as the information necessary for inversion may have been irretrievably lost during the modular reduction.*

*The observed entropy expansion and the potential loss of critical structural information during the modular reduction step may have significant implications for the hardness of the Module-ISIS** problem. If the transformation process destroys the structural properties of the lattice that could be exploited by adversaries, it may enhance the security of the Adh system by making it more resistant to lattice-based attacks.*

*However, it is important to note that this conjecture is based on empirical observations and requires formal verification. Further research is needed to rigorously analyze the relationship between the entropy expansion, the loss of structural information, and the hardness of the Module-ISIS** problem. Additionally, the precise impact of the modular reduction step on the invertibility of the transformation should be investigated to determine the feasibility of recovering the original vector* $\mathbf{x}$ *from the output vector* $\mathbf{y}$*.*

If validated, this conjecture would provide additional support for the security of the Adh system and highlight the potential benefits of incorporating higher-dimensional NTT mixing and reduction in lattice-based cryptographic constructions. The loss of structural information during the modular reduction step may introduce an additional layer of complexity that enhances the resistance of the system against potential attacks.

A.22 There is No Dual

Conjecture 6. *Assume a cryptographic lattice-based system that is designed to produce a complete lattice under operationally defined conditions. If the lattice is complete, then the dual lattice associated with this system is empty in the sense that it contains no small or practically useful vectors under computational feasibility constraints.*

Proof. Given that the lattice L is complete, every vector in L contributes to filling the entire n-dimensional space without gaps. By the construction of such a system, the density of the lattice in the primal space is maximized, implying that the minimal distance between lattice points is at its theoretical lower bound.

This maximal packing in the primal lattice leads to a minimal or non-existent set of vectors in the dual lattice L^* that can be exploited computationally. Specifically, the vectors in L^* that are typically targeted in dual lattice attacks (i.e., short vectors) are either too large to be used practically or are non-existent due to the inversion properties of the Fourier transform applied in constructing L.

Therefore, in the operational context of cryptographic computation where practicality and computational feasibility are key, the dual lattice can be considered effectively empty of useful vectors for cryptanalysis. Measurements show the effective bound for dual vectors is > 1 This results in a robust defense mechanism against dual lattice attacks, enhancing the cryptographic security of the system.

A.23 Potential for Transition to Anti-Cyclic Matrices

In our research on the Adh zero-knowledge proof system, we have extensively utilized the prime moduli $p = 257$ and $p = 65537$ in our algorithms and implementations. These primes have been chosen for their desirable properties, such as being Fermat primes of the form $2^k + 1$, which enable efficient polynomial arithmetic and the use of the Number Theoretic Transform (NTT) for fast operations.

However, recent advancements in lattice cryptanalysis have highlighted potential vulnerabilities associated with the use of cyclic matrices and the underlying algebraic structure of the ring of polynomials modulo $x^n - 1$. While our current design incorporates techniques such as extensive rejection sampling and a chaining construction to amplify complexity and destroy patterns, it is important to consider the potential benefits of transitioning to anti-cyclic matrices. Anti-cyclic matrices, which correspond to the ring of polynomials modulo $x^n + 1$, have been shown to provide stronger security guarantees compared to cyclic matrices in lattice-based cryptography. The irreducibility of $x^n + 1$ when n is a power of 2 ensures that the resulting lattice has a dense representation and does not exhibit any obvious weaknesses that could be exploited by an attacker.

If a transition to anti-cyclic matrices is deemed necessary based on further security analysis and research, our existing algorithms and codebase can be adapted to accommodate this change. The modifications required to switch from cyclic to anti-cyclic matrices are relatively straightforward, primarily involving polynomial arithmetic operations. In terms of the choice of parameters, our current use of $p = 257$ and $p = 65537$ can be maintained even with the transition to anti-cyclic matrices. These primes remain suitable for the anti-cyclic setting, providing the necessary security properties and enabling efficient computations.

However, it is important to conduct a thorough security analysis to assess the impact of the transition to anti-cyclic matrices on the overall security of the Adh system. This analysis should take into account the specific attack scenarios, the best-known algorithms for solving the underlying lattice problems, and the latest advances in lattice cryptanalysis. If the security analysis reveals significant vulnerabilities in the current design that can be mitigated by the transition to anti-cyclic matrices, and the improvements in security outweigh any potential impact on efficiency and performance, then making the switch to anti-cyclic matrices may be justified.

In conclusion, while our current design extensively utilizes the primes $p = 257$ and $p = 65537$, we are prepared to adapt our algorithms and codebase to support anti-cyclic matrices if necessary. The transition to anti-cyclic matrices can be achieved with relatively minor modifications, and our chosen primes remain suitable for the anti-cyclic setting. However, a comprehensive security analysis is essential to determine the necessity and benefits of such a transition. By carefully evaluating the results of this analysis and considering the specific requirements of the Adh system, we can make an informed decision on whether the transition to anti-cyclic matrices is warranted for the long-term security and practicality of our zero-knowledge proof system.

A.24 BKZ Cost Estimate

Conjecture 7 (Adjusted Efficiency Constant for BKZ in a 0-Free, Maximum Density Lattice)**.** *Let $\mathcal{L}$ be a lattice with dimension n, constructed under a "0-free regime" and exhibiting "maximum density". Let c_{base} denote the base value of the efficiency constant for the BKZ algorithm, typically chosen as $c_{base} = 0.292$ based on empirical studies and common usage in the lattice-based cryptography community. We conjecture that the adjusted efficiency constant c_{adj} for estimating the computational cost of BKZ in the context of $\mathcal{L}$ should be increased by 20% to 30% relative to c_{base}. Specifically:*

$$c_{adj} \in [1.20 \times c_{base}, 1.30 \times c_{base}] \approx [0.3504, 0.3796] \tag{43}$$

The justification for this adjustment is as follows:

1. The "0-free regime" of $\mathcal{L}$ significantly increases the complexity of the lattice reduction process by eliminating trivially short vectors. This feature alone suggests an increase in the efficiency constant by 10% to 20%.
2. The "maximum density" property of $\mathcal{L}$ further contributes to the hardness of the lattice, making it more challenging to distinguish between vectors. This characteristic also warrants an increase in the efficiency constant by approximately 10% to 20%.
3. The cumulative effect of both features, while not strictly additive, can be conservatively estimated to result in a total increase of 20% to 30% over the base value c_{base}.

This adjusted efficiency constant c_{adj} provides a more conservative estimate of the computational cost required to achieve lattice reduction in the specific context of $\mathcal{L}$. By accounting for the increased hardness introduced by the "0-free" and "maximum density" properties, the adjusted value helps to ensure a robust security margin against advanced lattice reduction techniques. **Note**: The exact value of c_{adj} within the conjectured range may be further refined based on empirical data and specific implementation details of the BKZ algorithm in the context of $\mathcal{L}$.

A.25 Distribution Analysis

Conjecture 8 (Uniform Distribution of Coefficients in the Adh Cryptographic System)**.** *Let $\mathcal{A}$ be the Adh cryptographic system with the following parameters:*

- *Dimension:* $n \in 128$
- *Number of rounds:* $rnds = 4$
- *Number of iterations:* $iters = 4$
- *Prime moduli:* $ps = [257, 257, 65537]$
- *Roots of unity:* $ws = [3, 3, 282]$
- *Second Roots of unity:* $ws2 = [1, 1, 1]$

For any key pair (pk, sk) generated by $\mathcal{A}$, the coefficient values $1, 2, \ldots, 256$ in the vectors produced by $\mathcal{A}$ using (pk, sk) are uniformly distributed.

Justification: To support the conjecture of uniform distribution of coefficients in the Adh cryptographic system, an extensive experimental analysis was conducted. The experimental design and results are as follows: **Experimental Design:**

- Four unique key pairs were generated using the seeds 950001, 950002, 950003, and 950004.

- For each key pair, over 100 million vectors were generated using the Adh cryptographic system with the specified parameters.
- The uniformity of the coefficient distribution was assessed using chi-square tests for each individual key pair and the combined dataset.
- Finally a second test was run against 338M vectors using $ws2$, as evidence supporting the assumption that uniform distribution also applies to the subset reduction, noted as $\omega = 1$ in the table below.

Results: The chi-square test results for the uniformity analysis are presented in Table 6. Across all individual tests and the combined dataset test, the chi-square statistics and the extremely high p-values (all greater than 0.9999) strongly support the hypothesis of uniform distribution. The p-values indicate that the observed coefficient distributions are highly consistent with the expected uniform distribution. The experimental results provide strong empirical evidence supporting the conjecture that the Adh cryptographic system produces vectors with uniformly distributed coefficients between 1 and 256. This uniformity property is crucial for ensuring the security and effectiveness of cryptographic protocols built upon the Adh system.

Key Seed	**Chi-square Statistic**	**P-value**
950001	133.05	0.9999999999
950002	150.46	0.9999999742
950003	139.38	0.9999999997
950004	121.51	0.9999999998
Combined	137.70	0.9999999999
$\omega = 1$	127.86	0.9999999999983357

Table 6: Chi-square test results for uniformity analysis.

The assumption of uniform coefficient distribution is well-justified based on the rigorous experimental analysis conducted across multiple key pairs and a large sample size of generated vectors. The chi-square tests and visual inspections consistently validate the uniformity of the coefficient values, providing a solid foundation for the security and reliability of the Adh cryptographic system.

Acknowledgments

The authors would like to acknowledge that the cryptographic system described herein is currently patent pending.

Bibliography

[1] Henry Bambury and Phong Q. Nguyen. *Improved Provable Reduction of NTRU and Hypercubic Lattices*. Cryptology ePrint Archive, Paper 2024/601. `https://eprint.iacr.org/2024/601`. 2024. URL: `https://eprint.iacr.org/2024/601`.

[2] Daniel J. Bernstein and Bo-Yin Yang. "Asymptotically Faster Quantum Algorithms to Solve Multivariate Quadratic Equations". In: *Post-Quantum Cryptography*. Ed. by Tanja Lange and Rainer Steinwandt. Cham: Springer International Publishing, 2018, pp. 487–506. ISBN: 978-3-319-79063-3.

[3] Xavier Bonnetain et al. *Improved Classical and Quantum Algorithms for Subset-Sum*. Cryptology ePrint Archive, Paper 2020/168. https://eprint.iacr.org/2020/168. 2020. URL: https://eprint.iacr.org/2020/168.

[4] Léo Ducas, Thomas Espitau, and Eamonn W. Postlethwaite. *Finding short integer solutions when the modulus is small*. Cryptology ePrint Archive, Paper 2023/1125. https://eprint.iacr.org/2023/1125. 2023. URL: https://eprint.iacr.org/2023/1125.

[5] Jianwei Li and Phong Q. Nguyen. *A Complete Analysis of the BKZ Lattice Reduction Algorithm*. Cryptology ePrint Archive, Paper 2020/1237. https://eprint.iacr.org/2020/1237. 2020. URL: https://eprint.iacr.org/2020/1237.

[6] Yehuda Lindell. "How To Simulate It - A Tutorial on the Simulation Proof Technique". In: *Electron. Colloquium Comput. Complex.* TR17 (2016). URL: https://api.semanticscholar.org/CorpusID:3331839.

[7] László Lovász and Herbert E. Scarf. "The Generalized Basis Reduction Algorithm". In: *Mathematics of Operations Research* 17.3 (1992), pp. 751–764. ISSN: 0364765X, 15265471. URL: http://www.jstor.org/stable/3689761 (visited on 04/17/2024).

[8] Daniele Micciancio and Oded Regev. "Worst-Case to Average-Case Reductions Based on Gaussian Measures". In: vol. 37. Nov. 2004, pp. 372–381. ISBN: 0-7695-2228-9. DOI: 10.1109/FOCS.2004.72.

[9] Yanbin Pan and Feng Zhang. *A Note on the Density of the Multiple Subset Sum Problems*. Cryptology ePrint Archive, Paper 2011/525. https://eprint.iacr.org/2011/525. 2011. URL: https://eprint.iacr.org/2011/525.

[10] The FPLLL development team. "fplll, a lattice reduction library, Version: 5.4.5". Available at https://github.com/fplll/fplll. 2023. URL: https://github.com/fplll/fplll.

[11] Lars Tebelmann, Michael Pehl, and Vincent Immler. *Side-Channel Analysis of the TERO PUF*. Cryptology ePrint Archive, Paper 2019/312. https://eprint.iacr.org/2019/312. 2019. DOI: 10.1007/978-3-030-16350-1_4. URL: https://eprint.iacr.org/2019/312.

[12] Xiaoyun Wang, Guangwu Xu, and Yang Yu. "Lattice-Based Cryptography: A Survey". In: *Chinese Annals of Mathematics, Series B* 44.6 (2023), pp. 945–960. DOI: 10.1007/s11401-023-0053-6. URL: https://doi.org/10.1007/s11401-023-0053-6.