

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

Samuel Lavery

sam@trustlessprivacy.com

April 30, 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

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Slicing into the Variants of the Module-ISIS Problem: A Pie Analogy</b>	<b>7</b>
<b>3</b>	<b>Preliminaries</b>	<b>8</b>
3.1	Notation and Definitions . . . . .	8
3.2	Unique Features . . . . .	8
3.3	Module-ISIS Problem . . . . .	9
3.4	Number Theoretic Transform (NTT) . . . . .	9

<b>4</b>	<b>The Adh Zero-Knowledge Proof System</b>	<b>9</b>
4.1	Overview . . . . .	10
4.2	Assumptions . . . . .	10
4.3	Key Generation . . . . .	10
4.4	Proof Generation . . . . .	11
4.5	Verification . . . . .	11
<b>5</b>	<b>Problem Definitions</b>	<b>11</b>
5.1	Module-ISIS+ definition . . . . .	11
5.1.1	Module-ISIS* Problem and Its Application to the Adh System . .	12
5.2	Definition of Module-ISIS* . . . . .	12
5.2.1	Hardness of Module-ISIS* . . . . .	13
5.2.2	Application to the Adh System . . . . .	13
5.2.3	Future Directions . . . . .	14
5.3	Definition of Module-ISIS** . . . . .	14
5.3.1	Application to the Adh System . . . . .	15
5.3.2	Experimental Observations . . . . .	15
5.3.3	Security Considerations . . . . .	16
5.3.4	Future Directions . . . . .	17
<b>6</b>	<b>Security Analysis</b>	<b>17</b>
6.1	Reduction of Adh's Module-ISIS to Module Modulus Subset Sum . . . .	17
6.1.1	Module-ISIS Problem Instance . . . . .	17
6.1.2	Mapping to Module Modulus Subset Sum . . . . .	18
6.2	NTT Transformation to Support Reduction to Module Modulus Subset Sum	18
6.2.1	Forging a Signature . . . . .	19
6.2.2	Reduction Proof . . . . .	19
6.3	Module-ISIS Security Reduction Mappings . . . . .	20
6.3.1	Mapping Module-ISIS . . . . .	20
6.3.2	Mapping Module-ISIS+ . . . . .	21
6.3.3	Mapping Module-ISIS* . . . . .	22
6.4	Module-ISIS Security Reductions . . . . .	23
6.4.1	Reduction to Module-ISIS . . . . .	23
6.4.2	Reduction to Module-ISIS+ . . . . .	23
6.4.3	Reduction to Module-ISIS* . . . . .	24
6.4.4	Reduction to Module-ISIS** . . . . .	24
6.5	BKZ Lattice Reduction Analysis $N = 128$ . . . . .	24
6.5.1	Security Estimate based on Root Hermite Factor . . . . .	27
6.5.2	Adjusting the Root Hermite Factor for Zero-Free Lattices . . . . .	27
6.5.3	Security Estimate for the Adh System with $n=256$ . . . . .	28
6.6	Experimental Analysis of Reduced Instances using Integer Linear Program- ming . . . . .	30
6.7	Conclusion and Future Work . . . . .	32
6.8	Supporting Arguments . . . . .	32
6.8.1	No Correlation . . . . .	32
6.8.2	Completeness Argument . . . . .	34
6.8.3	Impact of Zero Elimination on Lattice Reduction Algorithms . . .	34
6.8.4	Bounded Correlation between Chained Instances . . . . .	35

6.8.5	Argument of Soundness . . . . .	36
6.8.6	Empirical Evidence for Zero-Knowledge Property . . . . .	36
6.8.7	Analysis of the select_representation Function and Its Impact on Security . . . . .	37
6.9	Lattice Density in Module-ISIS . . . . .	38
6.9.1	Hypercube Volume . . . . .	38
6.9.2	Unit Cell Volume . . . . .	38
6.9.3	Packing Density . . . . .	38
<b>7</b>	<b>Practical Implementation Considerations</b>	<b>39</b>
7.1	Parameter Selection and Initial Security Estimates . . . . .	40
7.2	Configuration 1: Smaller Parameters $n = 128$ . . . . .	40
7.2.1	Parameters . . . . .	40
7.3	Security Estimates . . . . .	40
7.3.1	Conclusion . . . . .	40
7.4	Configuration 2: Larger Parameters $n = 256$ . . . . .	40
7.4.1	Parameters . . . . .	40
7.5	Security Estimates . . . . .	41
7.5.1	Conclusion . . . . .	41
7.6	Original Estimated Impact of Chaining . . . . .	41
7.7	New Conjecture on Impact of Chaining . . . . .	42
<b>8</b>	<b>Experimental Results</b>	<b>43</b>
8.1	FPLL Experimental Setup . . . . .	43
8.1.1	BKZ Results . . . . .	43
8.1.2	Non-BKZ Solver Results . . . . .	43
8.1.3	Conclusion . . . . .	43
8.2	Specific Reduction Attack Scenario Analysis . . . . .	44
8.2.1	Attack Analysis Conclusion . . . . .	45
8.3	Resistance to State of the Art Projection Reductions . . . . .	45
<b>9</b>	<b>Performance Evaluation</b>	<b>46</b>
<b>10</b>	<b>Comparative Analysis</b>	<b>46</b>
<b>11</b>	<b>Potential Use Cases and Applications</b>	<b>47</b>
11.1	Key Exchange Mechanism (KEM) . . . . .	47
11.2	Digital Signatures . . . . .	47
11.3	Identity-Based and Key-Policy Based Cryptography . . . . .	48
11.4	Secure Messaging Protocol . . . . .	48
11.5	Proof of Knowledge . . . . .	48
11.6	Homomorphic Cryptography . . . . .	48
<b>12</b>	<b>Known Issues</b>	<b>49</b>
12.1	Side-Channel Vulnerabilities and Mitigation Techniques . . . . .	49

<b>13 Open Questions and Future Work</b>	<b>49</b>
13.1 Verified Formal Security Proofs . . . . .	49
13.2 Parameter Optimization and Trade-offs . . . . .	50
13.3 Applications and Integration to Protocols . . . . .	50
13.4 Long-Term Security and Post-Quantum Cryptography . . . . .	50
<b>14 Conclusion</b>	<b>51</b>
<b>A Appendix</b>	<b>51</b>
A.1 Proof of Reduction to Module-ISIS . . . . .	51
A.2 Proof of Reduction to Module-ISIS+ . . . . .	52
A.3 Proof of Reduction to Module-ISIS* . . . . .	54
A.4 Proof of Reduction to Module-ISIS** . . . . .	55
A.5 Proof of Soundness for Module-ISIS+ . . . . .	56
A.6 Reduction to Module-ISIS . . . . .	57
A.7 Reduction to Dense Subset Sum - Quantum Hardness . . . . .	58
A.8 Quantum Hardness Estimation . . . . .	59
A.8.1 Instance 1: $n = 128$ and $m = 6$ . . . . .	59
A.8.2 Instance 2: $n = 256$ and $m = 6$ . . . . .	59
A.9 Density Preservation in Module-ISIS to Module Modulus Subset Sum Reduction . . . . .	60
A.10 Zero-Knowledge Proof . . . . .	60
A.11 Algorithms . . . . .	61
A.12 Notes . . . . .	61
A.13 Module-ISIS+ Parameters . . . . .	61
A.14 Module-ISIS* Parameters . . . . .	62
A.15 Module-ISIS** Parameters . . . . .	62
A.16 Algorithms . . . . .	62
A.17 Empirical Evidence for Zero-Knowledge Property . . . . .	68
A.17.1 Simulator-based Approach . . . . .	68
A.17.2 Statistical Tests . . . . .	68
A.17.3 Machine Learning Test . . . . .	69
A.17.4 Empirical Conclusion . . . . .	69
A.18 Zero-Knowledge Proof . . . . .	69
A.19 Probabilistic Completeness . . . . .	70
A.20 Proof of Completeness . . . . .	72
A.21 Conjecture on Entropy Expansion and Information Loss in Module-ISIS** with Higher-Dimensional NTT Mixing and Reduction . . . . .	73
A.22 There is No Dual . . . . .	74
A.23 Potential for Transition to Anti-Cyclic Matrices . . . . .	75
A.24 Consideration of Ajita’s Bound in the Adh System . . . . .	76
A.25 Formal Analysis of Short Key Implications under Ajita’s Bound . . . . .	76
A.25.1 Theoretical Framework . . . . .	76
A.25.2 Implications for Adh System . . . . .	76
A.25.3 Practical Considerations . . . . .	76
A.25.4 Conclusion . . . . .	77
A.26 BKZ Cost Estimate . . . . .	77
A.27 Distribution Analysis . . . . .	77

# 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

38 bit security remains an open research question. Nonetheless, this work showcases the  
 39 effectiveness of the full lattice structure and the chaining mechanism employed in the  
 40 Adh system.

Metric	Adh-128	Adh-256	ML-KEM1	ML-KEM5	ML-DSA3	ML-DSA5
<i>PK</i>	192B	384B	736B	1440B	1472B	2592B
<i>SK</i>	192B	384B	1632B	3168B	4000B	4864B
<i>CT/SIG</i>	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.

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

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

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

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

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

71 **Thesis 1.** *The Adh zero-knowledge proof system is secure under the hardness assump-*  
 72 *tion of the Module-ISIS problem, providing soundness, completeness, and zero-knowledge*  
 73 *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

- 121 • Use Cases and Applications
- 122 • Comparative Analysis, Known Problems, Conclusion and Future Work
- 123 • Detailed Appendix

## 124 3 Preliminaries

### 125 3.1 Notation and Definitions

126 Throughout this paper, we use the following notation:

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

134 We also define the following terms:

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

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

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

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

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

### 145 3.2 Unique Features

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

- 148 • **Nested NTT Calls:** The ZKVolute function used in the Adh system employs  
149 recursive NTT operations, allowing for efficient polynomial arithmetic, maintaining  
150 the necessary algebraic structure, while allowing for a diffusive dimensional shift  
151 and mix operation.
- 152 • **Rejection Sampling:** The rejection sampling technique is used throughout the  
153 Adh system to ensure that all the vectors involved are full vectors, eliminating the  
154 presence of zero coefficients and maintaining a complete lattice structure.
- 155 • **Chaining Functions:** Adh implements multiple WOTS+ like chaining function  
156 using number theoretic primitives to amplify hardness of the core module-ISIS  
157 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  $\beta$ , find a non-zero vector  $\mathbf{z} \in R_q^n$  such that:

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

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  $(q, n, m, \beta)$ . 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 decisionally 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  $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk\_chal}, \mathbf{pk\_rand})$ , where  $\text{ZKVolute}$  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.

## 239 4.4 Proof Generation

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

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

250 The proof generation algorithm ensures that the signature challenge and signature ran-  
 251 domness are full vectors, maintaining the complete lattice structure throughout the com-  
 252 putation.

## 253 4.5 Verification

254 The verification algorithm of the Adh system takes as input the public key  $\mathbf{pk}$ , the proof  
 255  $\mathbf{sig}$ , the signature challenge  $\mathbf{sig\_chal}$ , and the signature randomness  $\mathbf{sig\_rand}$ . It proceeds  
 256 as follows:

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

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

# 264 5 Problem Definitions

## 265 5.1 Module-ISIS+ definition

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

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

268 and a set of target vectors

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

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

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

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

$$\vdots \tag{6}$$

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

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

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

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

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

- 288 1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS+ instance.
- 289 2.  $\mathcal{B}$  generates the public key  $\mathbf{pk}$  and sends it to  $\mathcal{A}$ .
- 290 3.  $\mathcal{A}$  outputs a forged proof  $(\mathbf{sig}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$ .
- 291 4.  $\mathcal{B}$  computes  $\mathbf{z} = \mathbf{sig}^{-1} \mathbf{sig}$ , where  $\mathbf{sig}$  is a valid proof generated by  $\mathcal{B}$ .
- 292 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+  
 293 instance.

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

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

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

### 305 5.1.1 Module-ISIS\* Problem and Its Application to the Adh System

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

## 311 5.2 Definition of Module-ISIS\*

312 **Definition 8** (Module-ISIS\* Problem). *Let  $k$  be a positive integer denoting the number*  
 313 *of chained instances. Given uniformly random matrices  $\mathbf{A}_1, \dots, \mathbf{A}_k \in R_q^{m \times m}$  and a set of*

314 target vectors  $\mathbf{t}_1, \dots, \mathbf{t}_k \in R_q^m$ , find non-zero vectors  $\mathbf{z}_1, \dots, \mathbf{z}_k \in R_q^m$  such that:

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

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

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

### 323 5.2.1 Hardness of Module-ISIS\*

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

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

### 336 5.2.2 Application to the Adh System

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

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

349 However, it is crucial to carefully analyze the impact of using Module-ISIS\* on the  
350 security of the Adh system. Further research is needed to ensure that the use of multiple

351 secret keys does not introduce any unforeseen vulnerabilities or weaknesses that could be  
 352 exploited by an adversary.

### 353 5.2.3 Future Directions

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

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

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

## 369 5.3 Definition of Module-ISIS\*\*

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

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

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

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

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

### 386 5.3.1 Application to the Adh System

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

- 393 • Select compatible non-decreasing prime modulus  $p$  values for each level  $i$ .
- 394 • Assign a different root of unity  $\omega_i$  to each level  $i$ .
- 395 • Perform the NTT operations and pointwise multiplications at the first level. Lev-  
 396 els beyond the first perform pointwise addition at each level transformed by the  
 397 corresponding root of unity  $\omega_i$  and the prime modulus  $p_i$ .

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

402 For example  $ps = [257, 257, 257]$  with  $ws = [3, 2, 251]$  is a valid configuration. Other  
 403 commonly used examples are  $ps = [257, 257]$ ,  $ws = [3, 3]$ ,  $ps = [257, 65537]$  and  $ws =$   
 404  $[3, 282]$  or  $ps = [257, 257, 65537]$ ,  $ws = [3, 3, 501]$ .

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

### 409 5.3.2 Experimental Observations

410 The Module-ISIS\*\* problem with different roots of unity and different  $p$  values has been  
 411 observed to increase the Shannon entropy of the output proof values consistently and  
 412 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  $\mathbf{t} = \mathbf{A}\mathbf{s} \bmod q$ . If for proofs  $\pi_0$  and  $\pi_1$  generated by  $\mathcal{P}$  the distributions*

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

413 *are computationally indistinguishable (denoted by  $\approx_c$ ) and the entropy of  $\pi_0$  is higher*  
 414 *than the entropy of  $\pi_1$ , then it is computationally harder for an adversary to break the*  
 415 *soundness of  $\mathcal{L}$ .*

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

422 This phenomenon appears to be primarily due to the multi-stage transformation pro-  
 423 cess within the Number Theoretic Transform (NTT) domains. Initially, data is repre-  
 424 sented in lower-dimensional NTT spaces, which is then projected or transformed into a

425 larger, more complex NTT structure. This expanded representation is subsequently inte-  
426 grated through modular addition, before undergoing an NTT inversion operation. Such  
427 modular reductions likely amalgamate and obfuscate the dimensional structure and ac-  
428 tual information content, potentially enhancing the security against attempts to reverse-  
429 engineer the original input.

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

### 440 5.3.3 Security Considerations

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

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

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



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

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

### 482 5.3.4 Future Directions

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

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

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

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

## 503 6 Security Analysis

### 504 6.1 Reduction of Adh’s Module-ISIS to Module Modulus Sub- 505 set Sum

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

#### 510 6.1.1 Module-ISIS Problem Instance

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

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

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

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{z} \bmod q \quad (8)$$

518 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  
519 target vector.

### 520 6.1.2 Mapping to Module Modulus Subset Sum

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

#### 528 **Justification for Relaxation:**

- 529 • In NTT spaces, multiplication can be viewed as repeated addition, which is com-  
530 putationally more complex; however, the security implications in such algebraic  
531 structures derive from the properties of the transformations rather than the com-  
532 plexity of arithmetic operations alone.
- 533 • Subtraction, the direct inverse in additive operations in these fields, does not equiv-  
534 alently simplify the cryptographic challenge compared to division, the inverse of  
535 multiplication, which is more complex and not typically feasible in modular arith-  
536 metic settings.

## 537 6.2 NTT Transformation to Support Reduction to Module Mod- 538 ulus Subset Sum

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

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

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

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

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

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

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

560 **Mapped Elements from the Adh System to MMSP:**

- 561 • Public key: ( $\mathbf{pk} \in \mathbb{Z}q^n$ )
- 562 • Public challenge: ( $\mathbf{pkchal} \in \mathbb{Z}q^n$ )
- 563 • Public random: ( $\mathbf{pkrand} \in \mathbb{Z}q^n$ )
- 564 • Signature: ( $\mathbf{sig} \in \mathbb{Z}q^n$ )
- 565 • Signature challenge: ( $\mathbf{sigchal} \in \mathbb{Z}q^n$ )
- 566 • Signature random: ( $\mathbf{sigrand} \in \mathbb{Z}q^n$ )
- 567 • Secret key: ( $\mathbf{sk} \in \mathbb{Z}q^m$ )(mapped to ( $\mathbf{z}$ ))

### 568 6.2.1 Forging a Signature

569 The goal of an adversary in the Adh system is to forge a signature  $\mathbf{sig}$  such that it passes  
570 the verification equation:

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

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

$$\mathbf{t} = \mathbf{A} + \mathbf{z} \text{ mod } q \quad (10)$$

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

### 574 6.2.2 Reduction Proof

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

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

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

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

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

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

## 604 6.3 Module-ISIS Security Reduction Mappings

### 605 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**  $target\_vector$

---

606 Explanation:

- 607 • The inputs  $sk_I, rand\_chal$ , and  $chal$  correspond to the secret vector  $\mathbf{z}$ , the random  
608 matrix  $\mathbf{R}$ , and the public matrix  $\mathbf{A}$  in the Module-ISIS problem, respectively.
- 609 • The  $select\_representation$  function applies the NTT operation to the inputs, trans-  
610 forming them into the appropriate algebraic structure.
- 611 • The  $pointwise\_mul$  function computes the product  $\mathbf{A} \cdot \mathbf{z}$ , resulting in the target  
612 vector  $\mathbf{t}$ .
- 613 • The output  $target\_vector$  represents the target vector  $\mathbf{t}$  in the Module-ISIS problem.

---

**Algorithm 2** Mapping to ISIS+

---

**Require:**  $sk_I, rand\_chal, chal, p, w, iters, rnds$   
**Ensure:**  $proof\_rep$

```

 $sk_I \leftarrow select\_representation(sk_I, p, w)$ 
 $rand\_chal \leftarrow select\_representation(rand\_chal, p, w)$ 
 $chal \leftarrow select\_representation(chal, p, w)$ 
 $alt\_iterables \leftarrow list()$ 
 $ntt\_rep \leftarrow chal$ 
 $blinded\_values \leftarrow list()$ 
 $root\_chal \leftarrow chal$ 
 $blinded\_values.append(root\_chal)$ 
if  $iters > 0$  then
  for  $\_ \leftarrow 0$  to  $iters - 1$  do
     $ntt\_rep \leftarrow select\_representation(ntt\_rep, p, w)$ 
     $blinded\_values.append(ntt\_rep)$ 
     $alt\_iterables.append(ntt\_rep)$ 
  end for
  for  $z \leftarrow 1$  to  $iters - 1$  do
     $ntt\_rep \leftarrow pointwise\_mul(ntt\_rep, alt\_iterables[z], p)$ 
     $blinded\_values.append(ntt\_rep)$ 
  end for
   $chal \leftarrow ntt\_rep$ 
end if
 $target\_vector \leftarrow pointwise\_mul(chal, sk_I, p)$ 
 $proof\_rep \leftarrow pointwise\_mul(target\_vector, rand\_chal, p)$ 
 $new\_chal \leftarrow pointwise\_mul(root\_chal, rand\_chal, p)$ 
 $new\_chal \leftarrow pointwise\_add(new\_chal, chal, p)$ 
 $new\_chal \leftarrow pointwise\_add(new\_chal, rand\_chal, p)$ 
for  $xx \leftarrow 0$  to  $rnds - 1$  do
   $proof\_rep \leftarrow pointwise\_mul(proof\_rep, root\_chal, p)$ 
   $proof\_rep \leftarrow pointwise\_mul(proof\_rep, new\_chal, p)$ 
   $proof\_rep \leftarrow pointwise\_mul(proof\_rep, alt\_iterables[xx \bmod iters], p)$ 
   $new\_chal \leftarrow pointwise\_mul(new\_chal, new\_chal, p)$ 
   $new\_chal \leftarrow pointwise\_add(new\_chal, new\_chal, p)$ 
end for
return  $proof\_rep$ 

```

---

615 Explanation:

- 616 • The inputs  $sk_I$ ,  $rand\_chal$ , and  $chal$  correspond to the secret vector  $\mathbf{z}$ , the random  
617 matrix  $\mathbf{R}$ , and the public matrix  $\mathbf{A}_1$  in the ISIS+ problem, respectively.
- 618 • The  $select\_representation$  function applies the NTT operation to the inputs, trans-  
619 forming them into the appropriate algebraic structure.
- 620 • The  $pointwise\_mul$  function computes the product  $\mathbf{A}_1 \cdot \mathbf{z}$ , resulting in the target  
621 vector  $\mathbf{t}_1$ .
- 622 • The chaining mechanism is implemented using the  $alt\_iterables$  and  $blinded\_values$   
623 lists, where each iteration generates a new instance  $\mathbf{A}_{i+1}$  by applying the NTT  
624 operation to the previous instance  $\mathbf{A}_i$  and a random matrix  $\mathbf{R}_i$ .
- 625 • The  $pointwise\_mul$  and  $pointwise\_add$  functions are used to compute the target  
626 vectors  $\mathbf{t}_i$  for each instance in the chain.
- 627 • The output  $proof\_rep$  represents the final target vector  $\mathbf{t}_k$  in the ISIS+ problem.

---

**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

```

---

629 Explanation:

- 630 • The input *sk\_array* is an array of  $k + 1$  secret vectors, where  $k$  is the number of  
631 rounds (*rnds*). The first secret vector *sk\_array*[0] is used as the initial secret  $\mathbf{z}$ , and  
632 the subsequent secret vectors *sk\_array*[1] to *sk\_array*[ $k$ ] are used in each round.  
633 • The *select\_representation* function is applied to each secret vector in *sk\_array* to  
634 transform them into the appropriate algebraic structure.  
635 • The initial steps are similar to ISIS+, where the chaining mechanism is implemented  
636 using the *alt\_iterables* and *blinded\_values* lists.  
637 • In each round, the *pointwise\_mul* function is used to multiply the current *proof\_rep*  
638 with the corresponding secret vector *sk\_array*[ $xx + 1$ ] at the start of the loop.  
639 • The rest of the steps in each round are similar to ISIS+, where *proof\_rep* is multi-  
640 plied with *root\_chal*, *new\_chal*, and *alt\_iterables*[ $xx \bmod iters$ ].  
641 • The *new\_chal* is updated using *pointwise\_mul* and *pointwise\_add* in each round.  
642 • 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.  $\square$

### 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  $(\mathbf{A}_1, \mathbf{t}_1, \dots, \mathbf{t}_k, q, n, m, \beta)$ ,  $\mathcal{B}$  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  $(\mathbf{sig}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$ .
- $\mathcal{B}$  computes  $\mathbf{z} = \mathbf{sig}^{-1} \mathbf{sig}$ , 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.  $\square$

**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.  $\square$

### 682 6.4.3 Reduction to Module-ISIS\*

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

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

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

### 693 6.4.4 Reduction to Module-ISIS\*\*

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

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

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

## 705 6.5 BKZ Lattice Reduction Analysis $N = 128$

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

712 Figure 1 shows the distribution of the root Hermite factor (RHF) across different  
713 BKZ block sizes. The RHF is a measure of the quality of the reduced basis, with lower  
714 values indicating a better reduction. The mean RHF across all block sizes is approxi-  
715 mately 1.055, with minimal variation between block sizes. This suggests that increasing  
716 the BKZ block size does not significantly improve the quality of the reduced basis for the  
717 Adh system. The distribution of the adjusted shortest vector length, shown in Figure  
718 2, further supports this observation. The adjusted shortest vector length is computed  
719 as  $\ell/(\det(\mathcal{L}))^{1/\dim(\mathcal{L})}$ , where  $\ell$  is the length of the shortest vector found by BKZ. Higher  
720 values indicate a better reduction. The mean adjusted shortest vector length is approxi-  
721 mately 947, with minimal variation across block sizes. The lattice determinant, a measure  
722 of the volume of the fundamental parallelepiped of the lattice, is another important fac-  
723 tor in assessing the hardness of the lattice. Figure 3 shows the distribution of the lattice



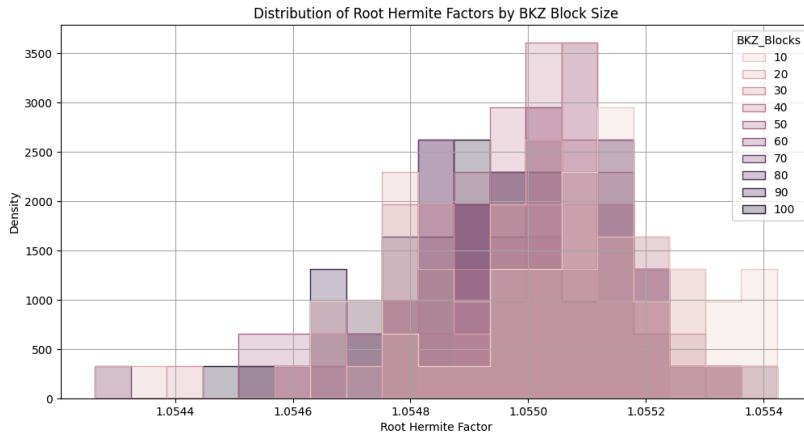


Figure 1: Distribution of Root Hermite Factors by BKZ Block Size

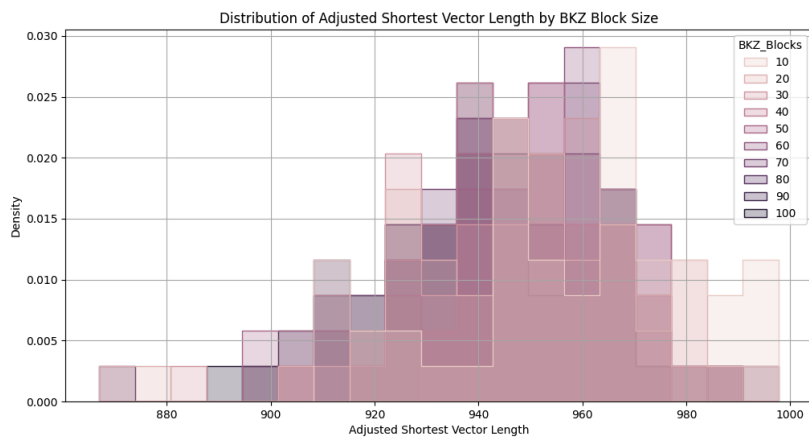


Figure 2: Distribution of Adjusted Shortest Vector Length by BKZ Block Size

724 determinant across BKZ block sizes. The mean lattice determinant is approximately  
 725 3.77, with a standard deviation of 2.40. The distribution is skewed towards lower values,  
 726 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

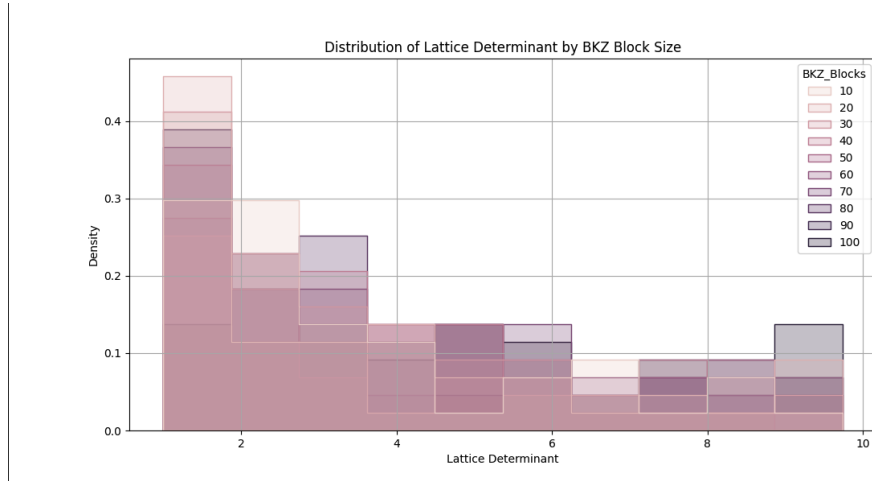


Figure 3: Distribution of Lattice Determinant by BKZ Block Size

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

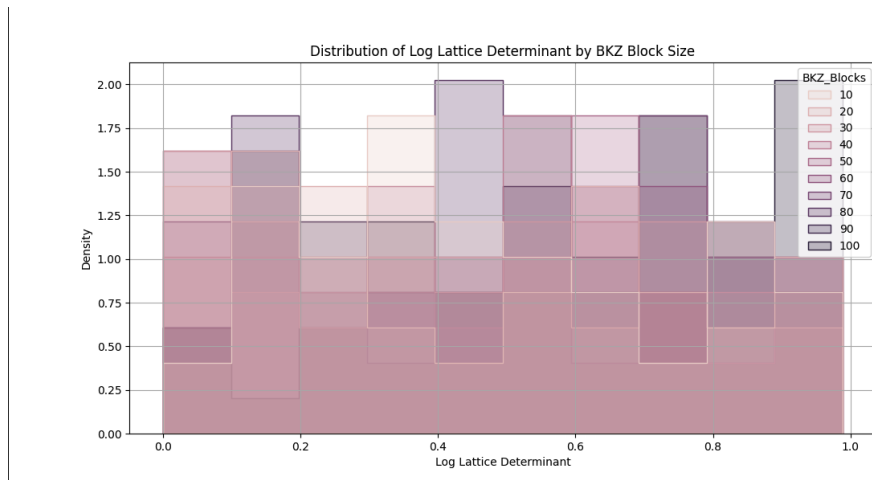


Figure 4: Distribution of Log Lattice Determinant by BKZ Block Size

729 results suggest that the Adh cryptographic system, with the specified parameters, exhibits  
 730 strong resistance against the BKZ lattice reduction algorithm. The minimal variation in  
 731 the RHF and adjusted shortest vector length across block sizes indicates that increas-  
 732 ing the BKZ block size does not significantly improve the quality of the reduced basis.  
 733 Furthermore, the concentration of the lattice determinant towards lower values suggests  
 734 that the reduced bases maintain a relatively small volume, which is a desirable property  
 735 for maintaining the hardness of the underlying lattice problem.  
 736

### 737 6.5.1 Security Estimate based on Root Hermite Factor

738 The Root Hermite Factor (RHF) is a key metric in assessing the quality of a lattice reduc-  
 739 tion algorithm and, consequently, the security of a lattice-based cryptographic system.  
 740 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  
 741 vector in the reduced basis,  $\det(\mathcal{L})$  is the determinant of the lattice  $\mathcal{L}$ , and  $n$  is the  
 742 dimension of the lattice.

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

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

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

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

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

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

### 767 6.5.2 Adjusting the Root Hermite Factor for Zero-Free Lattices

768 In the context of the Adh cryptographic system, which operates in a zero-free regime,  
 769 it is crucial to consider the impact of excluding zero vectors on the calculation of the  
 770 Root Hermite Factor (RHF). The RHF is a key metric for assessing the quality of a  
 771 lattice reduction algorithm and the security of a lattice-based cryptographic system. The  
 772 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  
 773 shortest non-zero vector in the reduced basis,  $\det(\mathcal{L})$  is the determinant of the lattice  
 774  $\mathcal{L}$ , and  $n$  is the dimension of the lattice. However, in a zero-free lattice, the shortest  
 775 vector length must be adjusted to account for the exclusion of zero vectors. We propose  
 776 an adjusted RHF calculation that incorporates a norm offset to handle the zero-free

777 property of the Adh system’s lattices. The adjusted RHF is computed as follows:

$$\delta_{\text{adj}} = \left( \frac{|\mathbf{v}|_{\text{adj}}}{(\det(\mathcal{L}))^{1/n}} \right)^{1/n}$$

$$|\mathbf{v}|_{\text{adj}} = \max(|\mathbf{v}| - \text{norm\_offset} + 1, 1)$$

778 where  $|\mathbf{v}|_{\text{adj}}$  is the adjusted shortest vector length, and `norm_offset` is an integer rep-  
 779 resenting the offset for the norm bound. The `max` function ensures that the adjusted  
 780 norm remains positive, preventing non-positive values under the root. This adjustment  
 781 is justified by the fact that the zero-free property of the Adh system’s lattices results in  
 782 a higher effective density compared to lattices that allow zero vectors. The exclusion of  
 783 zero vectors increases the minimum distance between lattice points, making the lattice  
 784 harder to reduce. Consequently, the security of the system is enhanced against lattice  
 785 reduction algorithms like BKZ.

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

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

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

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

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

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

### 811 6.5.3 Security Estimate for the Adh System with $n=256$

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

814 Figure 5 shows the distribution of the Root Hermite Factor (RHF) across different BKZ  
815 block sizes for the Adh system with  $n = 256$ . The mean RHF across all block sizes  
816 is approximately 1.028749, with minimal variation between block sizes. This suggests  
817 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

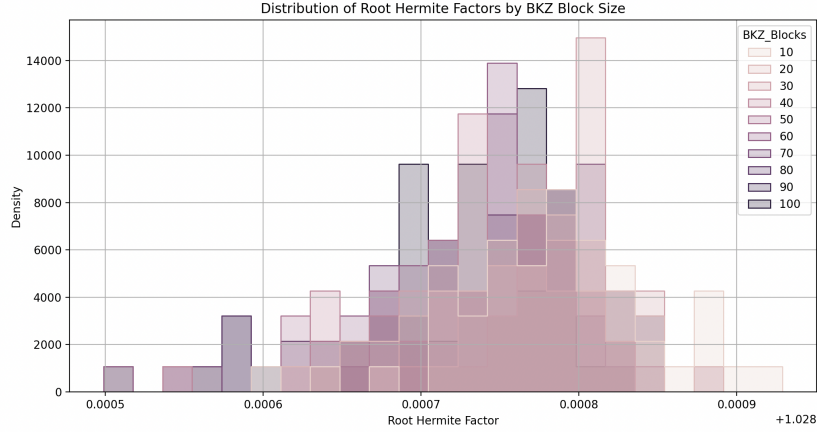


Figure 5: Distribution of Root Hermite Factors by BKZ Block Size for  $n=256$

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

- 823 1. Define the lattice dimension  $n = 256$  and the adjusted RHF  $\delta_{\text{adj}} = 1.028749$ .
- 824 2. Compute the gap  $\gamma = \delta_{\text{adj}}^{-n} = 1.028749^{-256} \approx 0.000545$ .
- 825 3. Compute the absolute value of the natural logarithm of  $\gamma$ :  $|\ln(\gamma)| \approx 7.514492$ .
- 826 4. Calculate the time complexity using the BKZ formula with the adjusted cost constant:  
827

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

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

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

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

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

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

## 846 6.6 Experimental Analysis of Reduced Instances using Integer 847 Linear Programming

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

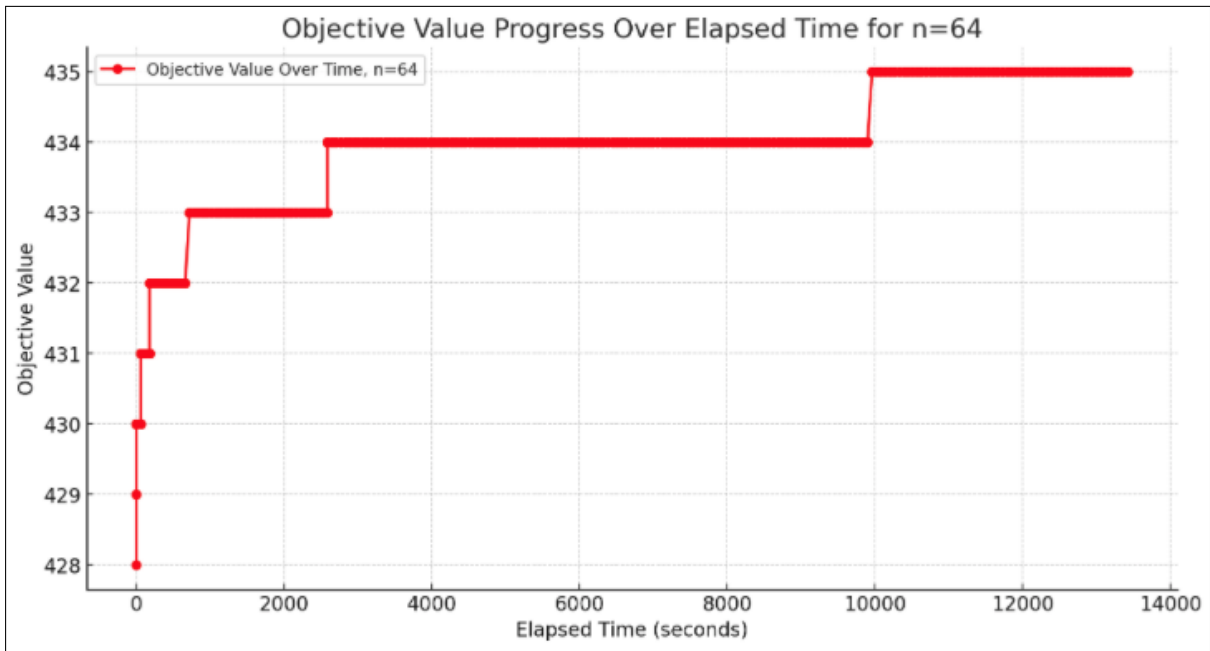


Figure 6:  $n = 64$  instance

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

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

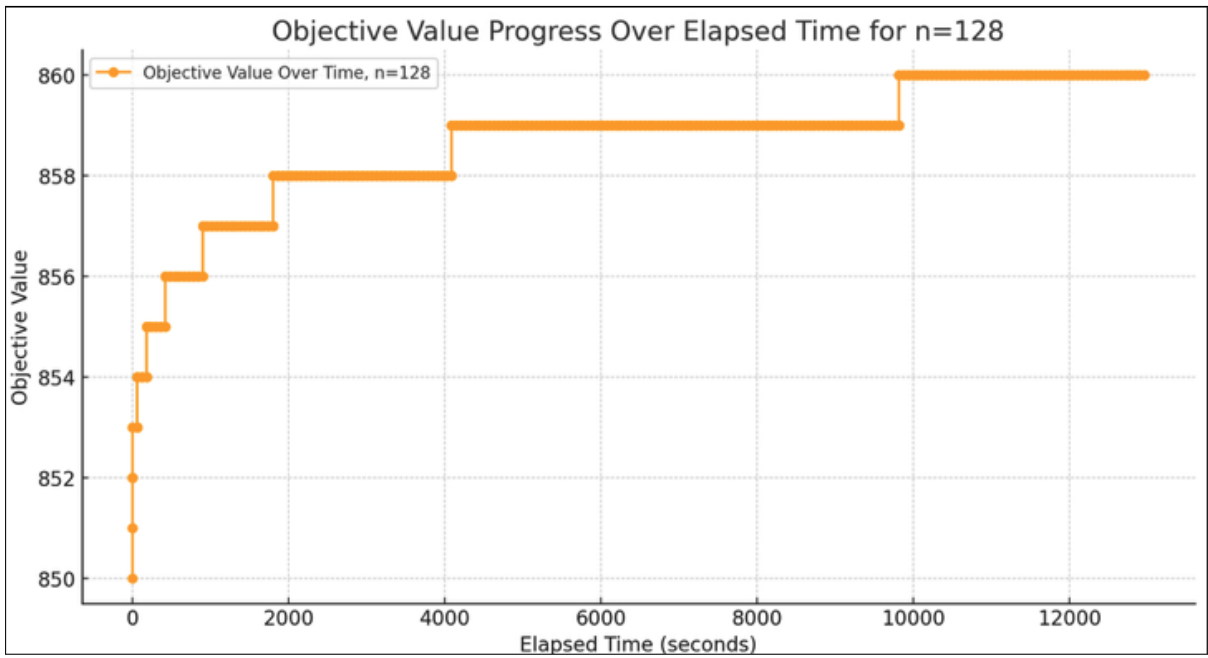


Figure 7:  $n = 128$  instance

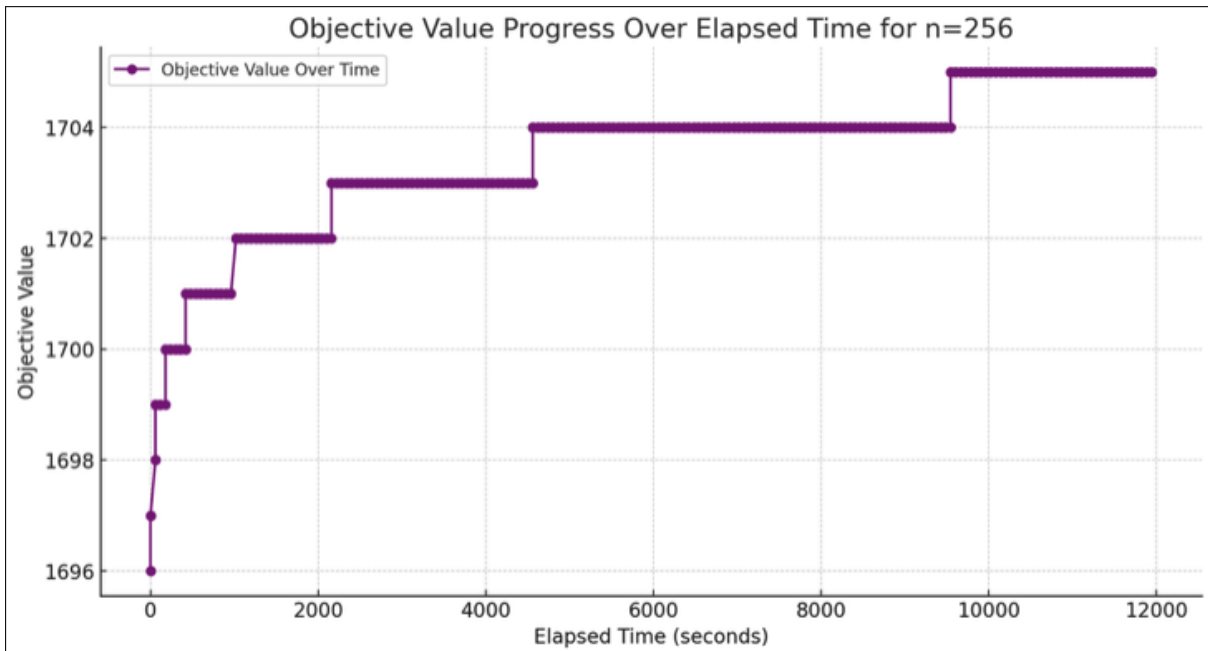


Figure 8:  $n = 256$  instance

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

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

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

## 880 6.7 Conclusion and Future Work

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

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

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

## 894 6.8 Supporting Arguments

### 895 6.8.1 No Correlation

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

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

- 904 1. **Uniform Distribution:** The output vectors of each instance in the Adh system  
 905 have been empirically demonstrated to follow a uniform distribution. Let  $\mathbf{X}_i =$   
 906  $(x_i, 1, x_{i,2}, \dots, x_{i,n})$  and  $\mathbf{X}_j = (x_j, 1, x_{j,2}, \dots, x_{j,n})$  be the output vectors of instances  
 907  $\mathcal{A}_i$  and  $\mathcal{A}_j$ , respectively. Then, for all  $l \in 1, 2, \dots, n$ :

$$\Pr[x_i, l = v] = \Pr[x_j, l = v] = \frac{1}{q}$$



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

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

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

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

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

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

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

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

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

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

932 This proof demonstrates that the design of the Adh system, with its use of NTT  
 933 operations, modular arithmetic, and context-bound randomizer values, effectively elim-  
 934 inates any correlation between the chained instances. The absence of correlation is a  
 935 crucial property that contributes to the overall security and resilience of the Adh system  
 936 against potential attacks that may attempt to exploit correlations between instances.  
 937 The uniform distribution of the output values, as empirically demonstrated, ensures that  
 938 the system maintains a high level of unpredictability and resistance to statistical analy-  
 939 sis. The independence preserved by the NTT operations and modular arithmetic further

940 strengthens the system’s security by preventing an adversary from inferring information  
941 about one instance based on the observations of another. Moreover, the introduction  
942 of fresh randomness through the context-bound randomizer values plays a vital role in  
943 preventing an adversary from manipulating the system for advantage. By adding these  
944 randomizer values to intermediate variables, the Adh system ensures that each instance  
945 is effectively isolated from the others, making it infeasible for an adversary to exploit any  
946 potential weaknesses or correlations.

### 947 **6.8.2 Completeness Argument**

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

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

954 **Theorem 9.** *The proof generation algorithm of the Adh system ensures that an honest  
955 prover can always generate a valid proof for a true statement. The use of rejection  
956 sampling and the availability of a source of true randomness guarantee that the prover  
957 can find a suitable signature randomness  $\mathbf{sig\_rand}$  that results in a valid proof. A complete  
958 proof provided in Appendix A.19.*

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

### 961 **6.8.3 Impact of Zero Elimination on Lattice Reduction Algorithms**

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

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

969 We provide a heuristic argument supporting this conjecture:

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

978 Further research is needed to formally analyze the impact of zero elimination on the  
979 efficiency of lattice reduction algorithms and to quantify its effect on the security of the  
980 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  $\mathcal{N} = NTT^{(1)}, \dots, NTT^{(n)}$  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  $\mathcal{N} = NTT^{(1)}, \dots, NTT^{(n)}$ , 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  $NTT^{(k)} \in \mathcal{N}$  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,

1028 in areas such as signals processing and image analysis the problem of reconstructing  
 1029 data from the input domain value using insufficient samples from the frequency(or NTT)  
 1030 domain is well studied.

1031 The hardness of the signal reconstruction problem in the NTT domain ensures that,  
 1032 given  $\mathbf{A}'$ , it is computationally infeasible to recover the original matrix  $\mathbf{A}$  without addi-  
 1033 tional information. This property, combined with the randomization introduced by the  
 1034 NTT, bounds the correlation between  $\mathbf{A}$  and  $\mathbf{A}'$ .

1035 While this argument requires a formal proof, we feel this lack of useful correlation to  
 1036 be a conservative assumption. Should this particular conjecture not hold, there are other  
 1037 ways to achieve a provably secure result. Thus, the security of Adh does not depend on  
 1038 this being correct, but we believe it will prove to be. A formal proof would involve a  
 1039 reduction from the signal reconstruction problem to the problem of recovering  $\mathbf{A}$  from  
 1040  $\mathbf{A}'$ , establishing the computational hardness of the latter. While out of scope for this  
 1041 paper, further work will formally bound this correlation and impact on security.

### 1042 6.8.5 Argument of Soundness

1043 Soundness is a crucial property of a zero-knowledge proof system, ensuring that a com-  
 1044 putationally bounded adversary cannot convince the verifier of a false statement, except  
 1045 with negligible probability. We provide a proof of soundness for the Adh system based on  
 1046 the hardness of the Module-ISIS problem. A complete proof is provided in the appendix.

1047 **Theorem 10** (Soundness). *The Adh zero-knowledge proof system is sound, assuming*  
 1048 *the hardness of the Module-ISIS problem. That is, a computationally bounded adversary*  
 1049 *cannot convince the verifier of a false statement, except with negligible probability.*

1050 *Proof.* Suppose there exists a probabilistic polynomial-time adversary  $\mathcal{A}$  that can con-  
 1051 vince the verifier of a false statement with non-negligible probability. We construct an  
 1052 algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the Module-ISIS problem. Given a Module-ISIS instance  
 1053  $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$ ,  $\mathcal{B}$  proceeds as follows:

- 1054 1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS instance.
- 1055 2.  $\mathcal{B}$  generates the public key  $\mathbf{pk}$  and sends it to  $\mathcal{A}$ .
- 1056 3.  $\mathcal{A}$  outputs a false statement and a proof ( $\mathbf{sig}, \mathbf{sig\_chal}, \mathbf{sig\_rand}$ ).
- 1057 4.  $\mathcal{B}$  verifies the proof using the verification algorithm of the Adh system.
- 1058 5. If the proof is accepted,  $\mathcal{B}$  computes  $\mathbf{z} = \mathbf{sig}^{-1} \mathbf{sig}$ , where  $\mathbf{sig}$  is a valid proof gener-  
 1059 ated by  $\mathcal{B}$ .
- 1060 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  
 1061 instance.

1062 The complete proof is provided in Appendix A.5. □

1063 This proof demonstrates that if an adversary can convince the verifier of a false state-  
 1064 ment, then they can solve the Module-ISIS problem, contradicting the assumed hardness  
 1065 of Module-ISIS. Therefore, the Adh system is sound, ensuring that an adversary cannot  
 1066 convince the verifier of a false statement, except with negligible probability.

### 1067 6.8.6 Empirical Evidence for Zero-Knowledge Property

1068 The zero-knowledge property ensures that a proof generated by the Adh system does  
 1069 not reveal any information about the secret key, except for the validity of the statement  
 1070 being proven. We present empirical evidence supporting the zero-knowledge property of  
 1071 the Adh system.

- 1072 • **Simulator-based approach:** We construct a simulator that generates proofs with-  
1073 out access to the secret key. The simulator’s output is computationally indistin-  
1074 guishable from the proofs generated by the real prover, suggesting that the proofs  
1075 do not leak information about the secret key.
- 1076 • **Statistical tests:** We perform statistical tests, such as the chi-squared test and the  
1077 Kolmogorov-Smirnov test, to compare the distribution of the proofs generated by  
1078 the real prover and the simulator. The test results indicate that the distributions  
1079 are statistically indistinguishable, supporting the zero-knowledge property.

1080 The detailed experimental setup and results are provided in Appendix A.17.

### 1081 6.8.7 Analysis of the `select_representation` Function and Its Impact on Se- 1082 curity

1083 The *select\_representation* function plays a crucial role in the Adh zero-knowledge proof  
1084 system by transforming the input vector into a suitable representation for further pro-  
1085 cessing. Currently, the function performs a forward Number Theoretic Transform (NTT)  
1086 on the input vector using a fixed prime modulus  $p$  and a root of unity  $\omega$ . The primary  
1087 objective of this function is to obtain a full vector representation, where all coefficients  
1088 are non-zero, to ensure the desired properties of the resulting lattice.

1089 One notable aspect of the *select\_representation* function is its behavior in finding a  
1090 full vector representation. Due to the *poly\_check* function, which verifies the suitability  
1091 of the input vector, we have a guarantee that the first NTT representation of any vector  
1092 will always be full. This property is essential for maintaining the security and correctness  
1093 of the Adh system.

1094 However, the number of attempts required by the *select\_representation* function to  
1095 find a full vector representation is not deterministic and depends on the specific choice  
1096 of the prime modulus  $p$  and the root of unity  $\omega$ . Empirical observations have shown that  
1097 the distribution of the number of attempts varies based on the selected field and root.

1098 For instance, when using  $p = 257$  and  $\omega = 3$ , approximately 60% of the time, the  
1099 function returns a full vector representation after a single attempt. In 39% of the cases,  
1100 a second attempt is required, and in the remaining 1% of the cases, the function is forced  
1101 to return a vector with at least one zero coefficient. This distribution highlights the  
1102 probabilistic nature of finding a full vector representation.

1103 Similarly, when using  $p = 257$  and  $\omega = 5$ , the distribution of the number of attempts  
1104 follows a downward slope, extending up to 8 potential NTT "frequencies" before the  
1105 probability of finding a full vector representation approaches zero. This behavior sug-  
1106 gests that the choice of the root of unity  $\omega$  can significantly impact the efficiency and  
1107 determinism of the *select\_representation* function.

1108 The decisional process of sorting through multiple slots, each with a certain probability  
1109 of yielding a good result, is an interesting aspect to consider in the context of the Adh  
1110 system’s security. While the specific details of this process may vary based on the chosen  
1111 field and root, it is unlikely to reveal any useful information about the original input to  
1112 the *select\_representation* function.

1113 This claim is supported by the fundamental principles of information theory, which  
1114 suggest that the amount of information that can be extracted from the output of the  
1115 *select\_representation* function is limited by the entropy of the input vector and the  
1116 properties of the NTT operation. The NTT, being a linear transformation, preserves the  
1117 statistical properties of the input vector, making it difficult for an attacker to gain any

1118 significant advantage by analyzing the decisional process.

1119 Furthermore, the use of rejection sampling techniques in the Adh system, combined  
1120 with the chaining construction and the careful selection of parameters, further enhances  
1121 the security by amplifying the complexity and destroying any discernible patterns in the  
1122 resulting lattice.

1123 In conclusion, the *select\_representation* function's behavior in finding a full vector  
1124 representation is an important aspect to consider in the Adh zero-knowledge proof system.  
1125 The distribution of the number of attempts required to find a full vector varies based on  
1126 the chosen field and root, highlighting the probabilistic nature of the process. However,  
1127 the decisional process itself is unlikely to reveal any useful information about the original  
1128 input, thanks to the fundamental limitations imposed by information theory and the  
1129 security measures employed in the Adh system. Further research into the impact of  
1130 different field and root choices on the efficiency and security of the *select\_representation*  
1131 function could provide valuable insights for optimizing the Adh system's performance  
1132 and robustness.

## 1133 6.9 Lattice Density in Module-ISIS

1134 In the context of Module-ISIS, where  $B = 257$  (infinity norm),  $q = 257$  (prime),  $n = 128$   
1135 or 256, and  $k = 6$  (rank), we consider a full construct with no zero-value coefficients  
1136 allowed. By rejection sampling out all vectors with zeros, we effectively work with a  
1137 universe of  $1-257$  (modulo 257), excluding the zero vector. As the

### 1138 6.9.1 Hypercube Volume

1139 The volume of the hypercube with side length  $B = 256 + 1$  in  $n$  dimensions is calculated  
1140 as:

- 1141 • For  $n = 128$ :  $257^{128}$
- 1142 • For  $n = 256$ :  $257^{256}$

### 1143 6.9.2 Unit Cell Volume

1144 The volume of the unit cell in the lattice, which is the fundamental parallelotope, is:

- 1145 • For  $n = 128$ :  $257^{128}$
- 1146 • For  $n = 256$ :  $257^{256}$

### 1147 6.9.3 Packing Density

1148 The packing density is the ratio of the hypercube volume to the unit cell volume:

- 1149 • For  $n = 128$ :  $\frac{257^{128}}{257^{128}} = 1$
- 1150 • For  $n = 256$ :  $\frac{257^{256}}{257^{256}} = 1$

1151 The packing density values of 1 indicates that the hypercubes occupy the entire unit  
1152 cell volume in the lattice. This high packing density suggests that the lattice is densely  
1153 packed, with no gaps between the hypercubes. This is a function of the infinite norm  
1154 bound being the same as the prime used for modular arithmetic. It is important to note  
1155 that the rank  $k$  does not directly affect the packing density calculation, as it represents  
1156 the dimension of the module. The high packing density of the Module-ISIS lattice has  
1157 potential implications for the security and hardness of the underlying problem:

- 1158 • The dense packing of the lattice makes it more challenging for lattice reduction  
1159 algorithms like BKZ to find short vectors, potentially enhancing the security of the  
1160 cryptographic system.
- 1161 • If the Module-ISIS problem can be reduced to a dense subset sum problem, the  
1162 high packing density could make it computationally infeasible to solve using known  
1163 optimization techniques for subset sum problems. This reduction, if possible, would  
1164 provide a strong argument for the security of the cryptographic system.
- 1165 • The absence of 0 coefficients in the module-ISIS lattice increases the density of  
1166 the lattice, making it more challenging for lattice reduction algorithms like BKZ  
1167 to find short vectors. This property could potentially enhance the security of the  
1168 cryptographic system.
- 1169 • If the module-ISIS problem can be reduced to a module-module subset sum problem,  
1170 the high density of the lattice could make it computationally infeasible to solve using  
1171 known optimization techniques for subset sum problems. This reduction, if possible,  
1172 would provide a strong argument for the security of the cryptographic system.
- 1173 • There are some theoretical results on the hardness of dense lattices, such as the work  
1174 by Micciancio and Regev [8], which shows that solving certain lattice problems on  
1175 dense lattices is at least as hard as solving them on general lattices.
- 1176

## 1177 7 Practical Implementation Considerations

1178 While not included in the formal security analysis presented in this paper, it is worth  
1179 noting that in practical implementations of the Adh system, where the first modulus  
1180 is chosen to be 257 or 65537, we can take advantage of the guaranteed absence of zero  
1181 coefficients to optimize storage and transport efficiency. By subtracting 1 from each coef-  
1182 ficient, we can ensure that the cryptographic variables follow 8-bit or 16-bit alignments,  
1183 rather than requiring 9 or 17 bits, respectively. This encoding process must be inverted  
1184 before using the variables in computations. It is important to emphasize that in practical  
1185 instances, the challenge and random variables should be generated from smaller values  
1186 corresponding to the appropriate bits of security required by the system. Table 3 presents  
1187 two prototype instances of the Adh system, illustrating the storage requirements for se-  
crets, 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.

1188  $p = 257$ ,  $m = 6$ , and  $B = 256$ , the secrets and public keys each require 192 bytes of stor-  
1189 age. The complete proofs consist of a 128-byte proof, a 32-byte random challenge, and  
1190 a 32-byte message challenge. The second instance, with parameters  $n = 256$ ,  $p = 257$ ,  
1191  $m = 6$ , and  $B = 256$ , requires 384 bytes for both secrets and public keys. The complete  
1192 proofs in this case include a 256-byte proof, a 64-byte random challenge, and a 64-byte  
1193 message challenge. Note that Module-ISIS\* will need to store  $k + 1$  unique secret keys,  
1194 one for each extra instance.

1196

## 1197 **7.1 Parameter Selection and Initial Security Estimates**

1198 The security of the Adh system relies on the appropriate selection of parameters, such as  
1199 the modulus  $q$ , the dimension  $n$ , the rank  $m$ , and the norm bound  $\beta$ . These parameters  
1200 should be chosen to ensure a desired level of security against known attacks, such as lattice  
1201 reduction and quantum algorithms [2]. To estimate the security complexity from a lattice  
1202 perspective, we used the specific MSIS hardness estimator located at the repository below.

1203 For the base Module-ISIS instance in the Adh system, we propose the following pa-  
1204 rameters:

- 1205 • Dimension  $n = 128$
- 1206 • Rank  $m = 6$
- 1207 • Modulus  $q = 257$
- 1208 • Norm bound  $\beta = 257$

1209 To estimate the security of the base Module-ISIS instance, we utilize the MSIS estimator  
1210 from the `pq-crystals/security-estimates` repository<sup>1</sup>.

## 1211 **7.2 Configuration 1: Smaller Parameters $n = 128$**

### 1212 **7.2.1 Parameters**

- 1213 • Ring Dimension ( $n$ ): 128
- 1214 • MSIS Dimension ( $w$ ): 768
- 1215 • Number of Equations ( $h$ ): 6
- 1216 • Norm Bound ( $B$ ): 257
- 1217 • Modulus ( $q$ ): 257

## 1218 **7.3 Security Estimates**

- 1219 • Dimensions: 98304
- 1220 • Block Size: 383
- 1221 • Probability of Success ( $\log_2(\epsilon)$ ): -79.50
- 1222 • Average Vectors per Run ( $\log_2$  nvector per run): 79.48
- 1223 • Length of Shortest Vector ( $l$ ): 4234.70

### 1224 **7.3.1 Conclusion**

1225 The estimator gives us a security level of 112 classical bits, which is lower than acceptable  
1226 for high-security applications.

## 1227 **7.4 Configuration 2: Larger Parameters $n = 256$**

### 1228 **7.4.1 Parameters**

- 1229 • Ring Dimension ( $n$ ): 256
- 1230 • MSIS Dimension ( $w$ ): 1536
- 1231 • Number of Equations ( $h$ ): 6
- 1232 • Norm Bound ( $B$ ): 257
- 1233 • Modulus ( $q$ ): 257

---

<sup>1</sup><https://github.com/pq-crystals/security-estimates>



## 1234 7.5 Security Estimates

- 1235 • Dimensions: 393216
- 1236 • Block Size: 889
- 1237 • Probability of Success ( $\log_2(\epsilon)$ ): -183.48
- 1238 • Average Vectors per Run ( $\log_2 n_{\text{vector per run}}$ ): 184.48
- 1239 • Length of Shortest Vector ( $l$ ): 6111.57

### 1240 7.5.1 Conclusion

1241 With a significantly enhanced security level of 260 bits, this  $n = 256$  configuration offers  
1242 better protection, potentially suitable for environments requiring very high security stan-  
1243 dards. The increase in ring dimension and MSIS dimension contributes substantially to  
1244 the heightened security.

## 1245 7.6 Original Estimated Impact of Chaining

1246 The Adh system employs a chaining mechanism, where the output of one Module-ISIS  
1247 instance is used as the input to the next instance. Let  $k$  denote the number of chained  
1248 instances in the system. The security of the Adh system grows with increasing  $k$ , as  
1249 an adversary would need to solve all  $k$  instances of the Module-ISIS+ or Module-ISIS\*  
1250 problem to forge a valid proof. If we assume additive complexity:

- 1251 • For  $k = 1$ : The security is equivalent to the base Module-ISIS instance, estimated  
1252 at least 112 bits.
- 1253 • For  $k = 2$ : Security increases to approximately 224 bits.
- 1254 • For  $k = 3$ : Security further increases to about 336 bits.
- 1255 • For  $k = 4$ : Security reaches around 448 bits, providing high-level security against  
1256 known attacks.

1257 These estimates serve as estimated theoretical bound on the security of the Adh system  
1258 and may be revised upwards or downwards as the exact hardness of the Module-ISIS+  
1259 family relative to Module-ISIS is better understood. Additionally, the attack estimates  
1260 assume the ability to use extremely large block sizes and dimensions that may not be  
1261 practical.

1262  
1263 The choice of  $k$  provides a trade-off between security and efficiency, with higher values  
1264 of  $k$  offering increased security at the cost of larger proof sizes and longer computation  
1265 times. The optimal value of  $k$  should be determined based on the specific security re-  
1266 quirements and performance constraints of the application.

1267  
1268 In addition to the chaining mechanism, the Adh system incorporates other features  
1269 that contribute to its security, such as the use of rejection sampling to ensure the unifor-  
1270 mity of the generated vectors and the elimination of zero coefficients to create a complete  
1271 lattice structure. These features further enhance the system's resilience against potential  
1272 attacks.

## 7.7 New Conjecture on Impact of Chaining

**Conjecture 4** (Rank Increase in Chained Module-ISIS Instances). Let  $\mathcal{P}(A_i, z_i, t_i)$  be an instance of the Module-ISIS problem, where  $A_i \in \mathbb{Z}_q^{r_i \times n}$  for  $n \in \{128, 256\}$ , representing the dimensions of the matrix based on security parameters. Here,  $r_i$  represents the rank of  $A_i$  which is subject to increase through the chaining process.  $z_i \in \mathbb{Z}_q^n$  is a secret vector, and  $t_i \in \mathbb{Z}_q^{r_i}$  is a target vector.

Consider a chaining mechanism that connects  $k$  instances of the Module-ISIS problem, where each instance's output vector  $z_i$  is transformed via a decisional selection of a full NTT representation and then combined with a random blinding vector  $R_i \in \mathbb{Z}_q^n$ , such that  $z_{i+1} = R_i \cdot \text{NTT}(z_i)$ .

This vector becomes the input for the next instance. The chaining process results in a transformed problem  $\mathcal{P}(A', z', t')$ , where  $A' \in \mathbb{Z}_q^{\sum_{i=1}^k r_i \times nk}$ ,  $z' \in \mathbb{Z}_q^{nk}$ , and  $t' \in \mathbb{Z}_q^{\sum_{i=1}^k r_i}$ . The conjecture posits that the rank of the transformed matrix  $A'$  is given by:

$$\text{rank}(A') = \sum_{i=1}^k r_i, \quad (11)$$

*Proof.* Let  $\mathcal{P}(A_i, z_i, t_i)$  be an instance of the Module-ISIS problem, where  $A_i \in \mathbb{Z}_q^{r_i \times n}$  for  $n \in \{128, 256\}$ , and  $r_i$  represents the rank of  $A_i$  which may increase as a result of chaining.  $z_i \in \mathbb{Z}_q^n$  is a secret vector, and  $t_i \in \mathbb{Z}_q^{r_i}$  is a target vector.

Consider a chaining mechanism that connects  $k$  instances of the Module-ISIS problem, where each instance's output vector  $z_i$  is transformed via a decisional NTT transformation and then combined with a random blinding vector  $R_i \in \mathbb{Z}_q^n$ , such that  $z_{i+1} = R_i \cdot \text{NTT}(z_i)$ .

The chaining process results in a transformed problem  $\mathcal{P}(A', z', t')$ , where  $A' \in \mathbb{Z}_q^{\sum_{i=1}^k r_i \times nk}$ ,  $z' \in \mathbb{Z}_q^{nk}$ , and  $t' \in \mathbb{Z}_q^{\sum_{i=1}^k r_i}$ . We can represent the transformed matrix  $A'$  as a block matrix:

$$A' = \begin{bmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_k \end{bmatrix}. \quad (12)$$

The rank of the block matrix  $A'$  can be expressed as:

$$\text{rank}(A') = \sum_{i=1}^k \text{rank}(A_i), \quad (13)$$

Since the NTT operation and the random blinding matrix  $R_i$  are linear transformations, they preserve the linear independence of the columns of  $A_i$ . Therefore, the rank of  $A'$  is exactly the sum of the ranks of the individual matrices  $A_i$ .  $\square$

The significance of this result lies in the increased complexity of the transformed Module-ISIS problem. A higher rank of the matrix  $A'$  implies a more complex lattice structure, making it more challenging for an attacker to find vectors and solve the problem. The chaining mechanism, along with the NTT operation and random blinding vectors, effectively increases the security of the cryptographic system by increasing the rank of the underlying matrix.

## 1306 8 Experimental Results

1307 To evaluate the resistance of the Adh zero-knowledge proof system against lattice re-  
1308 duction attacks, we conducted experiments using the fplll library [10], a well-established  
1309 toolkit for lattice-based cryptanalysis. Our primary focus was to assess the effectiveness  
1310 of various lattice reduction algorithms, including the Block Korkine-Zolotarev (BKZ)  
1311 algorithm [5], in finding short vectors within the lattices generated by the Adh system.

### 1312 8.1 FPLL Experimental Setup

1313 We designed an experimental setup in which a loop continuously generated matrices  
1314 representing the lattice structure of the Adh system. These matrices were then fed into  
1315 the fplll library, where different lattice reduction algorithms were applied to attempt to  
1316 find short vectors. We specifically investigated the performance of these algorithms for  
1317 two parameter settings:  $n = 128$  and  $n = 256$ , corresponding to the dimensions of the  
1318 lattice used in the Adh system.

#### 1319 8.1.1 BKZ Results

1320 The results of the BKZ experiments exhibited a consistent behavior across different block  
1321 sizes. For both  $n = 128$  and  $n = 256$ , the norms of the recovered vectors consistently  
1322 exceeded an average value of 270. Considering that the norm bound in the Adh system  
1323 is set to 256, these findings suggest that BKZ is not effective in finding sufficiently short  
1324 vectors to compromise the security of the system.

#### 1325 8.1.2 Non-BKZ Solver Results

1326 In addition to BKZ, we explored other lattice reduction techniques, including the Hermite-  
1327 Korkine-Zolotarev (HKZ) reduction [7], the Shortest Vector Problem (SVP) solvers, and  
1328 the Closest Vector Problem (CVP) solvers. When applied to lattices with dimension  
1329  $n = 128$ , these solvers initially appeared to find relatively short vectors within the lattice.  
1330 However, upon closer inspection, it was revealed that the average norm of the vectors  
1331 found by these solvers still exceeded 270, failing to breach the norm bound of 256 set in  
1332 the Adh system.

1333 The behavior of non-BKZ solvers against lattices with dimension  $n = 256$  exhibited more  
1334 variability. In some instances, these solvers returned outputs with higher norm averages  
1335 compared to the  $n = 128$  case. Moreover, the execution time of these solvers against  
1336  $n = 256$  lattices was significantly longer, sometimes taking several hours to complete.

#### 1337 8.1.3 Conclusion

1338 The experiments conducted with fplll provide valuable insights into the resilience of the  
1339 Adh zero-knowledge proof system against lattice reduction attacks. Despite the initial  
1340 appearance of finding short vectors by non-BKZ solvers at dimension  $n = 128$ , further  
1341 analysis revealed that the average norm of the recovered vectors consistently exceeded  
1342 270, failing to breach the norm bound of 257 set in the Adh system.

1343 The inability of both BKZ and non-BKZ solvers to find vectors shorter than the norm  
1344 bound in the practically relevant dimensions ( $n = 128$  and  $n = 256$ ) suggests that the Adh  
1345 system exhibits strong resistance against direct lattice reduction and projection-reduction

1346 attacks. The dense structure of the lattice, achieved through rejection sampling and the  
 1347 elimination of zero coefficients, is believed to contribute to the difficulty of finding short  
 1348 vectors using traditional lattice reduction methods.

1349 The variable behavior and occasional crashes encountered with non-BKZ solvers  
 1350 against lattices with dimension  $n = 256$  highlight the complexity and challenges as-  
 1351 sociated with analyzing the security of the Adh system. Further research is needed to  
 1352 fully understand the implications of these observations and to establish rigorous bounds  
 1353 on the system’s resistance against a wider range of cryptanalytic techniques.

## 1354 8.2 Specific Reduction Attack Scenario Analysis

1355 In this section, we evaluate the potential impact of the attack presented in the paper  
 1356 ”Finding short integer solutions when the modulus is small” [4] by Ducas, Espitau, and  
 1357 Postlethwaite on the Adh system with the parameters:  $q = 257$ ,  $B = 256$ ,  $m = 6$ ,  
 1358  $n = 128$ , and  $k = 4$  chained Module-ISIS+ instances. The attack exploits the Z-shape  
 1359 profile of the reduced basis and performs lattice sieving in projected sublattices to find  
 1360 short solutions.

1361 Let  $\beta$  denote the block size used in the BKZ lattice reduction algorithm. The effec-  
 1362 tiveness of the attack depends on the number of q-vectors ( $n_q$ ) remaining in the reduced  
 1363 basis after applying BKZ- $\beta$ . 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	$\beta$	$n_q$	$r - \ell$	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

1364 on Table 1 in the paper). The sieving dimension  $r - \ell$  is calculated as follows:  
 1365  
 1366

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

1367 Since the sieving dimension is negative, the attack is not applicable in this scenario.  
 1368 Similarly, in Scenario 2 ( $\beta = 60$ ), the sieving dimension is zero, making the attack inap-  
 1369 plicable. In Scenario 3 ( $\beta = 80$ ), the expected number of q-vectors is  $n_q \approx 3$  (extrapolated  
 1370 from Table 1). The sieving dimension and the number of sieving vectors are:

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

1371 Although the sieving dimension is positive, the probability of a lifted vector being a valid  
 1372 solution is low due to the small ratio between  $B$  and  $q$  ( $256/257 \approx 0.996$ ). Consequently,  
 1373 the attack is unlikely to succeed in this scenario. In Scenario 4 ( $\beta = 100$ ), the expected  
 1374 number of  $q$ -vectors is  $n_q \approx 1$ . The sieving dimension and the number of sieving vectors  
 1375 are:

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

1376 While the sieving dimension is positive and the number of sieving vectors is larger, the  
 1377 small ratio between  $B$  and  $q$  still limits the success probability of the attack.

### 1378 8.2.1 Attack Analysis Conclusion

1379 Based on the analysis with the parameters ( $q = 257$ ,  $n = 128$ ,  $B = 257$ ), the attack  
 1380 described in the paper appears to have limited effectiveness against the Adh system. The  
 1381 small lattice dimension  $m$  and the close proximity of the modulus  $q$  to the norm bound  
 1382  $B$  reduce the applicability and success probability of the attack.

1383 However, it is essential to note that this analysis focuses solely on the specific attack  
 1384 outlined in the paper and relies on the assumptions made therein. It does not preclude  
 1385 the existence of other attacks or potential improvements to the current attack that could  
 1386 impact the security of the Adh system.

## 1387 8.3 Resistance to State of the Art Projection Reductions

1388 A recent paper by Ducas, Espitau, and Postlethwaite [1] presents a new attack on lattice-  
 1389 based cryptosystems that exploits the  $\mathbb{Z}$ -shape profile of the reduced basis and performs  
 1390 lattice sieving in projected sublattices to find short solutions. However, this attack is not  
 1391 effective against the Adh system due to the high density of the lattice. In the Adh system,  
 1392 the lattice is constructed to be maximally dense, with a packing density of 1. This means  
 1393 that the product of the first minimum of the primal lattice and the first minimum of the  
 1394 dual lattice is much higher than 1:

$$\lambda_1(\mathcal{L}) \cdot \lambda_1(\mathcal{L}^*) \gg 1 \tag{14}$$

1395 The high density of the lattice makes it resistant to the new attack, as the success  
 1396 probability of the attack depends on the ratio between the bound  $B$  and the modulus  
 1397  $q$ . In the Adh system, this ratio is very close to 1 ( $B/q \approx 0.996$ ), which significantly  
 1398 limits the applicability and success probability of the attack. Therefore, while the new  
 1399 attack presented by Ducas et al. is an important advancement in lattice cryptanalysis, it  
 1400 does not pose a significant threat to the security of the Adh system due to the carefully  
 1401 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 param-

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.

eter  $n$  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.

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.

1438 Another distinctive aspect of the Adh system is its use of nested Number Theoretic  
1439 Transform (NTT) operations. The NTT plays a crucial role in enabling efficient poly-  
1440 nomial arithmetic, which is essential for the performance of lattice-based cryptographic  
1441 protocols. The Adh system leverages the properties of the NTT to achieve fast and  
1442 compact proof generation and verification, making it suitable for practical applications.

1443 The Adh system also incorporates advanced techniques such as rejection sampling  
1444 and the elimination of zero coefficients to maintain a complete lattice structure. These  
1445 techniques contribute to the system's security by reducing the attack surface and making  
1446 it harder for adversaries to exploit structural weaknesses. The rejection sampling ap-  
1447 proach ensures the uniformity of the generated vectors, preventing potential biases that  
1448 could be exploited by attackers.

1449 Furthermore, the Adh system supports multiple levels of proof generation and verifica-  
1450 tion, providing flexibility and adaptability to different security requirements and perfor-  
1451 mance constraints. This multi-level feature allows for the construction of more complex  
1452 proof systems and enables the Adh system to be used in a wider range of applications.

1453 In comparison to other lattice-based zero-knowledge proof systems, such as those  
1454 based on the Ring-SIS or Ring-LWE problems, the Adh system offers several advantages.  
1455 The Module-ISIS problem provides a more flexible and efficient framework for construct-  
1456 ing proofs, as it allows for the use of smaller moduli and dimensions while maintaining a  
1457 high level of security. The Adh system's chaining mechanism and multi-level proofs also  
1458 enable more advanced features and improved scalability compared to simpler lattice-based  
1459 proof systems.

## 1460 **11 Potential Use Cases and Applications**

1461 The Adh zero-knowledge proof system, with its unique lattice-based construction and  
1462 compact key and proof sizes, offers a versatile foundation for various cryptographic ap-  
1463 plications and protocols. The following subsections explore potential use cases where the  
1464 Adh system could provide secure and efficient solutions.

### 1465 **11.1 Key Exchange Mechanism (KEM)**

1466 The Adh system's underlying one-way chosen plaintext attack (OW-CPA) resistant scheme,  
1467 related to the subset sum problem, can be transformed into an indistinguishability under  
1468 chosen-ciphertext and prove attack (IND-CCPA) secure key exchange mechanism (KEM).  
1469 This KEM would enable parties to establish a shared secret key for secure communica-  
1470 tion, leveraging the hardness of the Module-ISIS problem and its variants. The compact  
1471 key sizes of the Adh system could lead to efficient key exchange protocols, particularly  
1472 suited for resource-constrained environments.

### 1473 **11.2 Digital Signatures**

1474 By applying the Fiat-Shamir transform to the Adh system, it is possible to construct exis-  
1475 tentially unforgeable under chosen message attack (EU-CMA) digital signature schemes.  
1476 These signatures would allow users to sign messages and verify the authenticity of the  
1477 signatures, providing a secure means of authentication and non-repudiation. The com-  
1478 pact signature sizes offered by the Adh system could be advantageous in scenarios where

1479 bandwidth or storage is limited, such as in Internet of Things (IoT) devices or blockchain  
1480 applications.

### 1481 **11.3 Identity-Based and Key-Policy Based Cryptography**

1482 The Adh system’s lattice construction opens up possibilities for identity-based and key-  
1483 policy based cryptography. In identity-based cryptography, users’ identities (e.g., email  
1484 addresses) serve as their public keys, simplifying key management and distribution. Key-  
1485 policy based cryptography enables fine-grained access control by associating policies with  
1486 keys, determining who can access encrypted data. The Adh system’s compact key sizes  
1487 and efficient operations could make it well-suited for implementing these advanced cryp-  
1488 tographic primitives, enabling secure and flexible access control mechanisms.

### 1489 **11.4 Secure Messaging Protocol**

1490 The PKEMNO NIZK (Public Key Exchange Mechanism with Non-Interactive Zero-  
1491 Knowledge Opening) secure messaging protocol, introduced in the paper, leverages the  
1492 unique characteristics of the Adh system. This protocol ensures the confidentiality and in-  
1493 tegrity of exchanged messages, making it suitable for secure communication applications.  
1494 The absence of a traditional decryption function and the use of the ZKVolute operation  
1495 in the Adh system could provide enhanced security and privacy features compared to  
1496 traditional messaging protocols.

### 1497 **11.5 Proof of Knowledge**

1498 The Adh system’s trapdoor-based proof of knowledge capabilities enable the construction  
1499 of protocols where a prover can demonstrate knowledge of a secret without revealing it to  
1500 the verifier. This property has applications in authentication, access control, and privacy-  
1501 preserving systems. For example, a user could prove their identity or membership in a  
1502 group without disclosing sensitive information. The zero-knowledge proofs generated by  
1503 the Adh system could be used to build secure and privacy-enhancing authentication and  
1504 authorization mechanisms.

### 1505 **11.6 Homomorphic Cryptography**

1506 The Adh system’s homomorphic properties, being a subcategory of ‘somewhat’ or ‘par-  
1507 tially’ homomorphic cryptographic systems, enable computations to be performed on  
1508 encrypted data without decrypting it first. This capability opens up possibilities for  
1509 privacy-preserving computations, such as secure multiparty computation or outsourced  
1510 computation on sensitive data. The compact key and ciphertext sizes of the Adh system  
1511 could make it more practical and efficient compared to other homomorphic encryption  
1512 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

1555 formal reductions and proofs, continuous peer review rigorous, mathematical analysis,  
1556 and refinement over time will provide stronger guarantees. We acknowledge the novelty  
1557 of some of proofs presented in the paper and encourage peer review and welcome feedback,  
1558 improvements or corrections.

## 1559 **13.2 Parameter Optimization and Trade-offs**

1560 Another area for future research is the optimization of the Adh system’s parameters and  
1561 the exploration of trade-offs between security and efficiency. The paper presents specific  
1562 parameter choices and provides experimental results, but a more comprehensive analysis  
1563 of parameter selection could yield further improvements. This work will presented in a  
1564 subsequent paper. Some questions to include:

- 1565 • What is the optimal choice of the prime modulus  $q$  and the dimension  $n$  to balance  
1566 security and performance?
- 1567 • How does the number of chained instances  $k$  affect the security and efficiency of the  
1568 system, and what is the optimal value of  $k$  for different security levels?
- 1569 • Can the rejection sampling technique be further optimized to reduce the computa-  
1570 tional overhead while maintaining the desired statistical properties?
- 1571 • Complexity comparison of various combinations of configurations beyond the base  
1572 cases presented in this work.

## 1573 **13.3 Applications and Integration to Protocols**

1574 The Adh zero-knowledge proof system has the potential to be applied in various crypto-  
1575 graphic protocols and privacy-preserving applications. Future work will investigate the  
1576 integration of the Adh system into existing protocols and explore new use cases. Some  
1577 potential non-standard directions include:

- 1578 • Integrating the Adh system into privacy-preserving authentication protocols, such  
1579 as anonymous credentials or attribute-based signatures.
- 1580 • Exploring the use of the Adh system in secure multi-party computation protocols,  
1581 enabling efficient and private computations among multiple participants.
- 1582 • Developing privacy-preserving blockchain applications that leverage the Adh system  
1583 for confidential transactions and smart contracts.
- 1584 • Supporting Swarm networking.

## 1585 **13.4 Long-Term Security and Post-Quantum Cryptography**

1586 As the field of quantum computing advances, it is crucial to assess the long-term security  
1587 of cryptographic systems against potential quantum attacks. While the Adh system is  
1588 based on lattice problems that are believed to be resistant to quantum algorithms, further  
1589 research is needed to solidify its post-quantum security guarantees. Future work could  
1590 focus on:

- 1591 • Conducting a thorough analysis of the Adh system’s resistance against known quan-  
1592 tum algorithms, such as Shor’s algorithm or Grover’s algorithm.
- 1593 • Exploring the use of quantum-resistant primitives, such as quantum-safe hash func-  
1594 tions or post-quantum digital signature schemes, in conjunction with the Adh sys-  
1595 tem.

1596 • Investigating the potential impact of future advancements in quantum computing  
 1597 on the security of the Adh system and developing mitigation strategies.  
 1598 In conclusion, the research presented in this paper on the Adh zero-knowledge proof sys-  
 1599 tem opens up a wide range of exciting possibilities for future work. From formal security  
 1600 proofs and parameter optimization to implementation enhancements and practical appli-  
 1601 cations, there are numerous avenues to explore and contribute to the field of lattice-based  
 1602 cryptography and zero-knowledge proofs. The open questions and challenges identified  
 1603 in this section provide a roadmap for researchers and practitioners to further advance the  
 1604 state of the art and strengthen the foundations of the Adh system.

## 1605 14 Conclusion

1606 In this work, we introduced the Adh zero-knowledge proof system, a novel lattice-based  
 1607 protocol that achieves compact proofs and strong security guarantees under the Module-  
 1608 ISIS assumption and its variants. Our core technical contributions include:

- 1609 • A comprehensive analysis of the Module-ISIS problem and its connection to the se-  
 1610 curity of Adh, including formal definitions of the ISIS+, ISIS\*, and ISIS\*\* variants.
- 1611 • An in-depth study of the effectiveness of BKZ and other lattice reduction techniques  
 1612 against Adh, demonstrating the system’s resistance to conventional and state-of-  
 1613 the-art cryptanalytic attacks.
- 1614 • Concrete parameter selection and performance benchmarks, showcasing Adh’s prac-  
 1615 ticality and efficiency compared to existing post-quantum alternatives.

1616 Our work also identified several avenues for further research, including optimizations  
 1617 to the zero-knowledge protocol, additional side-channel countermeasures, and perfor-  
 1618 mance optimizations. By contributing novel cryptographic techniques and rigorous se-  
 1619 curity analysis, this paper aims to advance the state of the art in post-quantum zero-  
 1620 knowledge proofs and lay the foundation for secure and efficient protocols in the quantum  
 1621 computing era.

1622 Fundamentally, the Adh system represents a promising step towards achieving the  
 1623 long-standing goal of compact, flexible, and quantum-secure zero-knowledge proofs. Its  
 1624 unique blend of lattice-based techniques and rejection sampling enables new possibili-  
 1625 ties for cryptographic protocol design. We hope this work spurs further innovations at  
 1626 the intersection of lattice cryptography and zero-knowledge, paving the way for a new  
 1627 generation of privacy-preserving technologies that can withstand the challenges of the  
 1628 post-quantum world.

## 1629 A Appendix

### 1630 A.1 Proof of Reduction to Module-ISIS

1631 **Theorem 11** (Reduction to Module-ISIS). *If there exists a probabilistic polynomial-time*  
 1632 *adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system with non-negligible probability,*  
 1633 *then there exists a probabilistic polynomial-time algorithm  $\mathcal{B}$  that can solve the Module-*  
 1634 *ISIS problem with non-negligible probability.*

1635 *Proof.* Suppose there exists an adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system  
 1636 with non-negligible probability. We construct an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the

1637 Module-ISIS problem. Given a Module-ISIS instance  $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$ ,  $\mathcal{B}$  proceeds as  
 1638 follows:

- 1639 1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS instance.  
 1640 It sets the modulus to  $q$ , the dimension to  $n$ , the rank to  $m$ , and the norm bound  
 1641 to  $\beta$ .
- 1642 2.  $\mathcal{B}$  generates the public key  $\mathbf{pk}$  by computing  $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk}_{\text{chal}}, \mathbf{pk}_{\text{rand}})$ ,  
 1643 where  $\mathbf{sk}$  is a randomly generated secret key,  $\mathbf{pk}_{\text{chal}}$  is the public challenge, and  
 1644  $\mathbf{pk}_{\text{rand}}$  is the public randomness.  $\mathcal{B}$  sets  $\mathbf{sk} = \mathbf{A}$  and  $\mathbf{pk}_{\text{chal}} = \mathbf{t}$ .  $\mathcal{B}$  sends  $\mathbf{pk}$  to  $\mathcal{A}$ .
- 1645 3.  $\mathcal{A}$  outputs a forged proof  $(\mathbf{sig}; \mathbf{sig}_{\text{chal}}; \mathbf{sig}_{\text{rand}}^*)$ .
- 1646 4.  $\mathcal{B}$  verifies the forged proof using the verification algorithm of the Adh system. If  
 1647 the proof is accepted,  $\mathcal{B}$  proceeds to the next step. Otherwise,  $\mathcal{B}$  aborts.
- 1648 5.  $\mathcal{B}$  computes  $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$ , where  $\mathbf{sig}$  is a valid proof generated by  $\mathcal{B}$  using the  
 1649 secret key  $\mathbf{sk}$ .
- 1650 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  
 1651 instance.

1652 The analysis of the success probability of  $\mathcal{B}$  follows similarly to the reduction to Module-  
 1653 ISIS+ in Appendix A.2. If  $\mathcal{A}$  succeeds in forging a valid proof with non-negligible proba-  
 1654 bility, 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.  
 1655 The success probability of  $\mathcal{B}$  is equal to the success probability of  $\mathcal{A}$ , which is assumed  
 1656 to be non-negligible. Therefore, if the Adh system is susceptible to forgery attacks,  
 1657 then Module-ISIS is solvable with non-negligible probability, contradicting the assumed  
 1658 hardness of Module-ISIS.  $\square$

## 1659 A.2 Proof of Reduction to Module-ISIS+

1660 **Theorem 12** (Reduction to Module-ISIS+). *If there exists a probabilistic polynomial-*  
 1661 *time adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system with non-negligible*  
 1662 *probability, then there exists a probabilistic polynomial-time algorithm  $\mathcal{B}$  that can solve*  
 1663 *the Module-ISIS+ problem with non-negligible probability.*

1664 *Proof.* Suppose there exists an adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system  
 1665 with non-negligible probability. We construct an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the  
 1666 Module-ISIS+ problem. Given a Module-ISIS+ instance  $(\mathbf{A}_1, \mathbf{t}_1, \dots, \mathbf{t}_k, q, n, m, \beta)$ ,  $\mathcal{B}$   
 1667 proceeds as follows:

- 1668 1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS+ instance.  
 1669 It sets the modulus to  $q$ , the dimension to  $n$ , the rank to  $m$ , and the norm bound  
 1670 to  $\beta$ .
- 1671 2.  $\mathcal{B}$  generates the public key  $\mathbf{pk}$  by computing  $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk}_{\text{chal}}, \mathbf{pk}_{\text{rand}})$ ,  
 1672 where  $\mathbf{sk}$  is a randomly generated secret key,  $\mathbf{pk}_{\text{chal}}$  is the public challenge, and  
 1673  $\mathbf{pk}_{\text{rand}}$  is the public randomness.  $\mathcal{B}$  sends  $\mathbf{pk}$  to  $\mathcal{A}$ .
- 1674 3.  $\mathcal{A}$  outputs a forged proof  $(\mathbf{sig}; \mathbf{sig}_{\text{chal}}; \mathbf{sig}_{\text{rand}}^*)$ .
- 1675 4.  $\mathcal{B}$  verifies the forged proof using the verification algorithm of the Adh system. If  
 1676 the proof is accepted,  $\mathcal{B}$  proceeds to the next step. Otherwise,  $\mathcal{B}$  aborts.
- 1677 5.  $\mathcal{B}$  computes  $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$ , where  $\mathbf{sig}$  is a valid proof generated by  $\mathcal{B}$  using the  
 1678 secret key  $\mathbf{sk}$ .
- 1679 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+  
 1680 instance.

1681 To analyze the success probability of  $\mathcal{B}$ , we observe that if  $\mathcal{A}$  succeeds in forging a valid  
 1682 proof with non-negligible probability, then the forged proof  $(\mathbf{sig}; \mathbf{sig}_{\text{chal}}; \mathbf{sig}_{\text{rand}}^*)$  must

1683 satisfy the verification equation:

$$\text{ZKVolute}(\mathbf{pk}, \mathbf{sig\_chal}, \mathbf{sig\_rand}) = \text{ZKVolute}(\mathbf{sig}, \mathbf{pk\_chal}, \mathbf{pk\_rand}) \quad (15)$$

1684 Substituting  $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk\_chal}, \mathbf{pk\_rand})$  and rearranging the terms, we obtain:

$$\text{ZKVolute}(\mathbf{sk}, \mathbf{sig\_chal}, \mathbf{sig\_rand}) = \mathbf{sig}^* \quad (16)$$

1685 Let  $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$ , where  $\mathbf{sig}$  is a valid proof generated by  $\mathcal{B}$  using the secret key  $\mathbf{sk}$ .

1686 Then, we have:

$$\text{ZKVolute}(\mathbf{sk}, \mathbf{sig\_chal}, \mathbf{sig\_rand}) - \text{ZKVolute}(\mathbf{sk}, \mathbf{sig\_chal}, \mathbf{sig\_rand}) = \mathbf{z} \quad (17)$$

1687 By the linearity of the ZKVolute function, we can rewrite this as:

$$\text{ZKVolute}(\mathbf{sk}, \mathbf{sig\_chal}^* - \mathbf{sig\_chal}, \mathbf{sig\_rand}^* - \mathbf{sig\_rand}) = \mathbf{z} \quad (18)$$

1688 Now, recall that in the Module-ISIS+ problem, we have:

$$\mathbf{A}_1 \cdot \mathbf{z} = \mathbf{t}_1 \text{ mod } q \quad (19)$$

$$\mathbf{A}_2 \cdot \mathbf{z} = \mathbf{t}_2 \text{ mod } q \quad (20)$$

$$\vdots \quad (21)$$

$$\mathbf{A}_k \cdot \mathbf{z} = \mathbf{t}_k \text{ mod } q \quad (22)$$

$$(23)$$

1689 where  $\mathbf{A}_i = \text{NTT}(\mathbf{A}_i - 1) \cdot \text{NTT}(\mathbf{R})$  for  $i = 2, \dots, k$ , with  $\mathbf{R}$  being a random matrix in  
 1690  $R_q^{m \times m}$ . By the construction of the Adh system, we have:

$$\mathbf{A}_1 = \mathbf{sk} \quad (24)$$

$$\mathbf{t}_1 = \mathbf{sig\_chal}^* - \mathbf{sig\_chal} \quad (25)$$

$$\mathbf{t}_2 = \text{NTT}(\mathbf{sig\_chal}^* - \mathbf{sig\_chal}) \cdot \text{NTT}(\mathbf{sig\_rand}^* - \mathbf{sig\_rand}) \quad (26)$$

$$\vdots \quad (27)$$

$$\mathbf{t}_k = \text{NTT}^{(k-1)}(\mathbf{sig\_chal}^* - \mathbf{sig\_chal}) \cdot \text{NTT}^{(k-1)}(\mathbf{sig\_rand}^* - \mathbf{sig\_rand}) \quad (28)$$

$$(29)$$

1691 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+  
 1692 instance. The success probability of  $\mathcal{B}$  is equal to the success probability of  $\mathcal{A}$  in forging  
 1693 a valid proof, which is assumed to be non-negligible. Therefore,  $\mathcal{B}$  solves the Module-  
 1694 ISIS+ problem with non-negligible probability, contradicting the assumed hardness of  
 1695 Module-ISIS+.  $\square$

1696 This reduction demonstrates that if an adversary can forge a valid proof in the Adh  
 1697 system with non-negligible probability, then the Module-ISIS+ problem can be solved  
 1698 with non-negligible probability, contradicting the assumed hardness of Module-ISIS+.  
 1699 Therefore, the Adh system is secure against forgery attacks, assuming the hardness of  
 1700 the Module-ISIS+ problem.

### 1701 A.3 Proof of Reduction to Module-ISIS\*

1702 **Theorem 13** (Reduction to Module-ISIS\*). *If there exists a probabilistic polynomial-time*  
 1703 *adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system with non-negligible probability,*  
 1704 *then there exists a probabilistic polynomial-time algorithm  $\mathcal{B}$  that can solve the Module-*  
 1705 *ISIS\* problem with non-negligible probability.*

1706 *Proof.* Suppose there exists an adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system  
 1707 with non-negligible probability. We construct an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the  
 1708 Module-ISIS\* problem. Given a Module-ISIS\* instance

1709  $(\mathbf{A}_1, \dots, \mathbf{A}_k, \mathbf{t}_1, \dots, \mathbf{t}_k, q, n, m, \beta)$ ,  $\mathcal{B}$  proceeds as follows:

1710 1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS\* instance.  
 1711 It sets the modulus to  $q$ , the dimension to  $n$ , the rank to  $m$ , and the norm bound  
 1712 to  $\beta$ .

1713 2.  $\mathcal{B}$  generates the public keys  $\mathbf{pk}_1, \dots, \mathbf{pk}_k$  by computing  
 1714  $\mathbf{pk}_i = \text{ZKVolute}(\mathbf{sk}_i, \mathbf{pk\_chal}_i, \mathbf{pk\_rand}_i)$ , where  $\mathbf{sk}_i$  is a randomly generated  
 1715 secret key,  $\mathbf{pk\_chal}_i$  is the public challenge, and  $\mathbf{pk\_rand}_i$  is the public randomness  
 1716 for the  $i$ -th instance.  $\mathcal{B}$  sends  $\mathbf{pk}_1, \dots, \mathbf{pk}_k$  to  $\mathcal{A}$ .

1717 3.  $\mathcal{A}$  outputs forged proofs

1718  $(\mathbf{sig}_1 \cdot \mathbf{sig\_chal}_1 \cdot \mathbf{sig\_rand}_1), \dots, (\mathbf{sig}_k \cdot \mathbf{sig\_chal}_k \cdot \mathbf{sig\_rand}_k)$ .

1720 4.  $\mathcal{B}$  verifies the forged proofs using the verification algorithm of the Adh system. If  
 1721 all the proofs are accepted,  $\mathcal{B}$  proceeds to the next step. Otherwise,  $\mathcal{B}$  aborts.

1722 5. For each  $i = 1, \dots, k$ ,  $\mathcal{B}$  computes  $\mathbf{z}_i = \mathbf{sig}_i^* - \mathbf{sig}_i$ , where  $\mathbf{sig}_i$  is a valid proof  
 1723 generated by  $\mathcal{B}$  using the secret key  $\mathbf{sk}_i$ .

1724 6. If  $\mathbf{z}_i \neq \mathbf{0}$  and  $\|\mathbf{z}_i\|_\infty \leq 2\beta$  for all  $i = 1, \dots, k$ , then  $\mathcal{B}$  outputs  $(\mathbf{z}_1, \dots, \mathbf{z}_k)$  as a  
 1725 solution to the Module-ISIS\* instance.

1726 To analyze the success probability of  $\mathcal{B}$ , we observe that if  $\mathcal{A}$  succeeds in forging valid  
 1727 proofs with non-negligible probability, then the forged proofs  $(\mathbf{sig}_i \cdot \mathbf{sig\_chal}_i \cdot \mathbf{sig\_rand}_i)$   
 1728 for  $i = 1, \dots, k$  must satisfy the verification equations:

$$\text{ZKVolute}(\mathbf{pk}_i, \mathbf{sig\_chal}_i \cdot \mathbf{sig\_rand}_i) = \text{ZKVolute}(\mathbf{sig}_i \cdot \mathbf{pk\_chal}_i, \mathbf{pk\_rand}_i) \quad (30)$$

1729 Substituting  $\mathbf{pk}_i = \text{ZKVolute}(\mathbf{sk}_i, \mathbf{pk\_chal}_i, \mathbf{pk\_rand}_i)$  and rearranging the terms, we  
 1730 obtain:

$$\text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig\_chal}_i \cdot \mathbf{sig\_rand}_i) = \mathbf{sig}_i^* \quad (31)$$

1731 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  
 1732  $\mathbf{sk}_i$ . Then, we have:

$$\text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig\_chal}_i \cdot \mathbf{sig\_rand}_i) - \text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig\_chal}_i, \mathbf{sig\_rand}_i) = \mathbf{z}_i \quad (32)$$

1733 By the linearity of the ZKVolute function, we can rewrite this as:

$$\text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig\_chal}_i^* - \mathbf{sig\_chal}_i, \mathbf{sig\_rand}_i^* - \mathbf{sig\_rand}_i) = \mathbf{z}_i \quad (33)$$

1734 Now, recall that in the Module-ISIS\* problem, we have:

$$\mathbf{A}_1 \cdot \mathbf{z}_1 = \mathbf{t}_1 \pmod q \quad \mathbf{A}_2 \cdot \mathbf{z}_2 = \mathbf{t}_2 \pmod q \quad \dots \quad \mathbf{A}_k \cdot \mathbf{z}_k = \mathbf{t}_k \pmod q \quad (34)$$

1735 where  $\mathbf{t}_i = \text{mask}(\mathbf{A}_i \cdot \mathbf{z}_i - 1) \cdot \mathbf{z}_i$  for  $i = 2, \dots, k$ , with  $\mathbf{t}_1 = \mathbf{A}_1 \cdot \mathbf{z}_1$ . By the construction  
 1736 of the Adh system, we have:

$$\mathbf{A}_i = \mathbf{sk}_i \mathbf{t}_1 \quad = \mathbf{sig\_chal}_1^* - \mathbf{sig\_chal}_1 \mathbf{t}_i \quad (35)$$

1737

$$\mathbf{t}_i = \text{mask}(\mathbf{sk}_i \cdot \mathbf{z}_i - 1) \cdot (\mathbf{sig\_chal}_1^* - \mathbf{sig\_chal}_1 \mathbf{t}_i) \quad (36)$$

1738 for  $i = 2, \dots, k$ . Therefore, if  $\mathbf{z}_i \neq \mathbf{0}$  and  $\|\mathbf{z}_i\|_\infty \leq 2\beta$  for all  $i = 1, \dots, k$ , then  
 1739  $(\mathbf{z}_1, \dots, \mathbf{z}_k)$  is a valid solution to the Module-ISIS\* instance. The success probability  
 1740 of  $\mathcal{B}$  is equal to the success probability of  $\mathcal{A}$  in forging valid proofs, which is assumed  
 1741 to be non-negligible. Therefore,  $\mathcal{B}$  solves the Module-ISIS\* problem with non-negligible  
 1742 probability, contradicting the assumed hardness of Module-ISIS\*.  $\square$

1743 This reduction demonstrates that if an adversary can forge valid proofs in the Adh  
 1744 system with non-negligible probability, then the Module-ISIS\* problem can be solved  
 1745 with non-negligible probability, contradicting the assumed hardness of Module-ISIS\*.  
 1746 Therefore, the Adh system is secure against forgery attacks, assuming the hardness of  
 1747 the Module-ISIS\* problem.

#### 1748 A.4 Proof of Reduction to Module-ISIS\*\*

1749 **Theorem 14** (Reduction to Module-ISIS\*\*). *If there exists a probabilistic polynomial-*  
 1750 *time adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system with non-negligible*  
 1751 *probability, then there exists a probabilistic polynomial-time algorithm  $\mathcal{B}$  that can solve*  
 1752 *the Module-ISIS\*\* problem with non-negligible probability.*

1753 *Proof.* Suppose there exists an adversary  $\mathcal{A}$  that can forge a valid proof in the Adh sys-  
 1754 tem with non-negligible probability. We construct an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve  
 1755 the Module-ISIS\*\* problem. Given a Module-ISIS\*\* instance

1756

( $\mathbf{A}_1, \dots, \mathbf{A}_k, \mathbf{t}_1, \dots, \mathbf{t}_k, p_1, \dots, p_k, \omega_1, \dots, \omega_k, n, m, \beta$ ),  $\mathcal{B}$  proceeds as follows:

1757

1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS\*\* in-  
 1758 stance. It sets the moduli to  $p_1, \dots, p_k$ , the dimension to  $n$ , the rank to  $m$ , the  
 1759 norm bound to  $\beta$ , and the roots of unity to  $\omega_1, \dots, \omega_k$ .

1760

2.  $\mathcal{B}$  generates the public keys  $\mathbf{pk}_1, \dots, \mathbf{pk}_k$  by computing

1761

$\mathbf{pk}_i = \text{ZKVolute}(\mathbf{sk}_i, \mathbf{pk\_chal}_i, \mathbf{pk\_rand}_i)$ , where  $\mathbf{sk}_i$  is a randomly generated  
 1762 secret key,  $\mathbf{pk\_chal}_i$  is the public challenge, and  $\mathbf{pk\_rand}_i$  is the public randomness  
 1763 for the  $i$ -th instance.  $\mathcal{B}$  sends  $\mathbf{pk}_1, \dots, \mathbf{pk}_k$  to  $\mathcal{A}$ .

1764

3.  $\mathcal{A}$  outputs forged proofs  $(\mathbf{sig}_1 \mathbf{sig\_chal}_1 \mathbf{sig\_rand}_1), \dots, (\mathbf{sig}_k \mathbf{sig\_chal}_k \mathbf{sig\_rand}_k)$ .

1765

4.  $\mathcal{B}$  verifies the forged proofs using the verification algorithm of the Adh system. If  
 1766 all the proofs are accepted,  $\mathcal{B}$  proceeds to the next step. Otherwise,  $\mathcal{B}$  aborts.

1767

5. For each  $i = 1, \dots, k$ ,  $\mathcal{B}$  computes  $\mathbf{z}_i = \mathbf{sig}_i^* - \mathbf{sig}_i$ , where  $\mathbf{sig}_i$  is a valid proof  
 1768 generated by  $\mathcal{B}$  using the secret key  $\mathbf{sk}_i$ .

1769

6. If  $\mathbf{z}_i \neq \mathbf{0}$  and  $\|\mathbf{z}_i\|_\infty \leq 2\beta$  for all  $i = 1, \dots, k$ , then  $\mathcal{B}$  outputs  $(\mathbf{z}_1, \dots, \mathbf{z}_k)$  as a  
 1770 solution to the Module-ISIS\*\* instance.

1771

1772 The analysis of the success probability of  $\mathcal{B}$  follows similarly to the reduction to Module-  
 1773 ISIS\*. If  $\mathcal{A}$  succeeds in forging valid proofs with non-negligible probability, then the  
 1774 forged proofs  $(\mathbf{sig}_i \mathbf{sig\_chal}_i \mathbf{sig\_rand}_i)$  for  $i = 1, \dots, k$  must satisfy the verification  
 1775 equations:

$$\text{ZKVolute}(\mathbf{pk}_i, \mathbf{sig\_chal}_i \mathbf{sig\_rand}_i) = \text{ZKVolute}(\mathbf{sig}_i \mathbf{pk\_chal}_i, \mathbf{pk\_rand}_i) \quad (37)$$

1776 Substituting  $\mathbf{pk}_i = \text{ZKVolute}(\mathbf{sk}_i, \mathbf{pk}_{\text{chal}_i}, \mathbf{pk}_{\text{rand}_i})$  and rearranging the terms, we  
 1777 obtain:

$$\text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig}_{\text{chal}_i}, \mathbf{sig}_{\text{rand}_i}) = \mathbf{sig}_i^* \quad (38)$$

1778 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  
 1779  $\mathbf{sk}_i$ . Then, we have:

$$\text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig}_{\text{chal}_i}, \mathbf{sig}_{\text{rand}_i}) - \text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig}_{\text{chal}_i}, \mathbf{sig}_{\text{rand}_i}) = \mathbf{z}_i \quad (39)$$

1780 By the linearity of the ZKVolute function, we can rewrite this as:

$$\text{ZKVolute}(\mathbf{sk}_i, \mathbf{sig}_{\text{chal}_i}^* - \mathbf{sig}_{\text{chal}_i}, \mathbf{sig}_{\text{rand}_i}^* - \mathbf{sig}_{\text{rand}_i}) = \mathbf{z}_i \quad (40)$$

1781 Now, recall that in the Module-ISIS\*\* problem, we have:

$$\mathbf{A}_1 \cdot \mathbf{z}_1 = \mathbf{t}_1 \pmod{p_1} \quad \mathbf{A}_2 \cdot \mathbf{z}_2 = \mathbf{t}_2 \pmod{p_2} \quad \dots \quad \mathbf{A}_k \cdot \mathbf{z}_k = \mathbf{t}_k \pmod{p_k} \quad (41)$$

1782 where  $\mathbf{t}_i = \text{mask}(\mathbf{A}_i \cdot \mathbf{z}_i - 1) \cdot \mathbf{z}_i$  for  $i = 2, \dots, k$ , with  $\mathbf{t}_1 = \mathbf{A}_1 \cdot \mathbf{z}_1$ . By the construction  
 1783 of the Adh system, we have:

$$\mathbf{A}_i = \mathbf{sk}_i \mathbf{t}_1 = \mathbf{sig}_{\text{chal}_1}^* - \mathbf{sig}_{\text{chal}_1} \quad \mathbf{t}_i = \text{mask}(\mathbf{sk}_i \cdot \mathbf{z}_i - 1) \cdot (\mathbf{sig}_{\text{chal}_i}^* - \mathbf{sig}_{\text{chal}_i}) \quad (42)$$

1784 for  $i = 2, \dots, k$ .

1785 Therefore, if  $\mathbf{z}_i \neq \mathbf{0}$  and  $\|\mathbf{z}_i\|_\infty \leq 2\beta$  for all  $i = 1, \dots, k$ , then  $(\mathbf{z}_1, \dots, \mathbf{z}_k)$  is a  
 1786 valid solution to the Module-ISIS\*\* instance. The success probability of  $\mathcal{B}$  is equal to the  
 1787 success probability of  $\mathcal{A}$  in forging valid proofs, which is assumed to be non-negligible.  
 1788 Therefore,  $\mathcal{B}$  solves the Module-ISIS\*\* problem with non-negligible probability, contra-  
 1789 dicting the assumed hardness of Module-ISIS\*\*.  $\square$

1790 This reduction demonstrates that if an adversary can forge valid proofs in the Adh  
 1791 system with non-negligible probability, then the Module-ISIS\*\* problem can be solved  
 1792 with non-negligible probability, contradicting the assumed hardness of Module-ISIS\*\*.  
 1793 Therefore, the Adh system is secure against forgery attacks, assuming the hardness of  
 1794 the Module-ISIS\*\* problem.

## 1795 A.5 Proof of Soundness for Module-ISIS+

1796 **Theorem 15** (Soundness). *The Adh zero-knowledge proof system is sound, assuming*  
 1797 *the hardness of the Module-ISIS problem. That is, a computationally bounded adversary*  
 1798 *cannot convince the verifier of a false statement, except with negligible probability.*

1799 *Proof.* Suppose there exists a probabilistic polynomial-time adversary  $\mathcal{A}$  that can con-  
 1800 vince the verifier of a false statement with non-negligible probability. We construct an  
 1801 algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the Module-ISIS problem. Given a Module-ISIS instance  
 1802  $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$ ,  $\mathcal{B}$  proceeds as follows:

- 1803 1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS instance.  
 1804 It sets the modulus to  $q$ , the dimension to  $n$ , the rank to  $m$ , and the norm bound  
 1805 to  $\beta$ .
- 1806 2.  $\mathcal{B}$  generates the public key  $\mathbf{pk}$  by selecting a secret key  $\mathbf{sk}$ , a public challenge  $\mathbf{pk}_{\text{chal}}$ ,  
 1807 and a randomizing value  $\mathbf{pk}_{\text{rand}}$  uniformly at random from the range  $[1, 256]$ . It then  
 1808 computes the convolution part of the public key as  $\mathbf{pk}' = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk}_{\text{chal}}, \mathbf{pk}_{\text{rand}})$   
 1809 and sets  $\mathbf{pk} = (\mathbf{pk}', \mathbf{pk}_{\text{chal}}, \mathbf{pk}_{\text{rand}})$ .  $\mathcal{B}$  sends  $\mathbf{pk}$  to  $\mathcal{A}$ .



- 1810 3.  $\mathcal{A}$  outputs a false statement and a proof  $(\mathbf{sig}; \mathbf{sig}'_{\text{chal}}; \mathbf{sig}^*_{\text{rand}})$ .
- 1811 4.  $\mathcal{B}$  verifies the proof using the verification algorithm of the Adh system. The verifica-
- 1812 tion is performed by checking the equivariance condition:  $\text{ZKVolute}(\mathbf{pk}, \mathbf{sig}'_{\text{chal}}; \mathbf{sig}^*_{\text{rand}}) =$
- 1813  $\text{ZKVolute}(\mathbf{sig}^*, \mathbf{pk}_{\text{chal}}, \mathbf{pk}_{\text{rand}})$ . This condition ensures that only the party possess-
- 1814 ing the secret key  $\mathbf{sk}$  can generate a valid proof that morphs the challenge and
- 1815 randomness in the same way as the public key was generated. The equivariance
- 1816 property is based on the associativity and commutativity of the ZKVolute function,
- 1817 which is a lossy hash function that destroys information while preserving the ability
- 1818 to verify the proof of possession.
- 1819 5. If the proof is accepted,  $\mathcal{B}$  computes  $\mathbf{z} = \mathbf{sig}^* - \mathbf{sig}$ , where  $\mathbf{sig}$  is a valid proof
- 1820 generated by  $\mathcal{B}$  using the secret key  $\mathbf{sk}$ , a challenge  $\mathbf{sig}'_{\text{chal}}$  derived from the message
- 1821  $m$ , and a randomly selected value  $\mathbf{sig}_{\text{rand}}$ .
- 1822 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
- 1823 instance. The condition  $\|\mathbf{z}\|_{\infty} \leq 2\beta$  ensures that  $\mathbf{z}$  is a valid solution to the
- 1824 Module-ISIS problem, as the Adh system's rejection sampling guarantees that all
- 1825 vectors have non-zero coefficients bounded by  $\beta$ .

1826 To analyze the success probability of  $\mathcal{B}$ , we observe that if  $\mathcal{A}$  succeeds in convincing

1827 the verifier of a false statement with non-negligible probability, then the forged proof

1828  $(\mathbf{sig}; \mathbf{sig}'_{\text{chal}}; \mathbf{sig}^*_{\text{rand}})$  must satisfy the verification equation. The reduction works as follows:

1829 If an adversary  $\mathcal{A}$  can forge a valid proof in the Adh system with non-negligible prob-

1830 ability, then  $\mathcal{B}$  can use  $\mathcal{A}$  to solve the Module-ISIS problem. By setting up the public

1831 parameters and the public key using the Module-ISIS instance,  $\mathcal{B}$  ensures that a forged

1832 proof that passes verification corresponds to a solution to the Module-ISIS problem. The

1833 difference between the forged proof and a valid proof generated by  $\mathcal{B}$  yields a vector  $\mathbf{z}$

1834 that satisfies the Module-ISIS conditions. In summary, the key steps of the reduction

1835 are:

1836 Setting up the Adh system using the Module-ISIS instance parameters. Generat-

1837 ing the public key using randomly selected values. Obtaining a forged proof from the

1838 adversary  $\mathcal{A}$ . Verifying the forged proof using the equivariance condition. Computing

1839 the difference between the forged proof and a valid proof to obtain a solution to the

1840 Module-ISIS problem.

1841 If the Adh system is not sound, then an adversary  $\mathcal{A}$  can forge proofs with non-

1842 negligible probability, implying that the Module-ISIS problem can be solved with non-

1843 negligible probability by  $\mathcal{B}$ . This contradicts the assumed hardness of the Module-ISIS

1844 problem, proving that the Adh system is sound.  $\square$

## 1845 A.6 Reduction to Module-ISIS

1846 **Theorem 16** (Reduction to Module-ISIS). *If there exists a probabilistic polynomial-time*

1847 *adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system with non-negligible probability,*

1848 *then there exists a probabilistic polynomial-time algorithm  $\mathcal{B}$  that can solve the Module-*

1849 *ISIS problem with non-negligible probability.*

1850 *Proof.* Suppose there exists an adversary  $\mathcal{A}$  that can forge a valid proof in the Adh system

1851 with non-negligible probability. We construct an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the

1852 Module-ISIS problem. Given a Module-ISIS instance  $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$ ,  $\mathcal{B}$  proceeds as

1853 follows:

- 1854 1.  $\mathcal{B}$  sets up the public parameters of the Adh system using the Module-ISIS instance.
- 1855 It sets the modulus to  $q$ , the dimension to  $n$ , the rank to  $m$ , and the norm bound

- 1856 to  $\beta$ .
- 1857 2.  $\mathcal{B}$  generates the public key  $\mathbf{pk}$  by computing  $\mathbf{pk} = \text{ZKVolute}(\mathbf{sk}, \mathbf{pk\_chal}, \mathbf{pk\_rand})$ ,
  - 1858 where  $\mathbf{sk}$  is a randomly generated secret key,  $\mathbf{pk\_chal}$  is the public challenge, and
  - 1859  $\mathbf{pk\_rand}$  is the public randomness.  $\mathcal{B}$  sets  $\mathbf{sk} = \mathbf{A}$  and  $\mathbf{pk\_chal} = \mathbf{t}$ .  $\mathcal{B}$  sends  $\mathbf{pk}$  to
  - 1860  $\mathcal{A}$ .
  - 1861 3.  $\mathcal{A}$  outputs a forged proof  $(\mathbf{sig}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$ .
  - 1862 4.  $\mathcal{B}$  verifies the forged proof using the verification algorithm of the Adh system. If
  - 1863 the proof is accepted,  $\mathcal{B}$  proceeds to the next step. Otherwise,  $\mathcal{B}$  aborts.
  - 1864 5.  $\mathcal{B}$  computes  $\mathbf{z} = \mathbf{sig}^- \mathbf{sig}$ , where  $\mathbf{sig}$  is a valid proof generated by  $\mathcal{B}$  using the secret
  - 1865 key  $\mathbf{sk}$ .
  - 1866 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
  - 1867 instance.

1868 The analysis of the success probability of  $\mathcal{B}$  follows similarly to the reduction to Module-

1869 ISIS+ in Appendix A.2. If  $\mathcal{A}$  succeeds in forging a valid proof with non-negligible prob-

1870 ability, then  $\mathbf{z}$  satisfies  $\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \bmod q$  and  $\|\mathbf{z}\|_{-\infty} \leq 2\beta$ , solving the Module-ISIS

1871 instance. The success probability of  $\mathcal{B}$  is equal to the success probability of  $\mathcal{A}$ , which

1872 is assumed to be non-negligible. Therefore, if the Adh system is susceptible to forgery

1873 attacks, then Module-ISIS is solvable with non-negligible probability, contradicting the

1874 assumed hardness of Module-ISIS.  $\square$

## 1875 A.7 Reduction to Dense Subset Sum - Quantum Hardness

1876 **Theorem 17** (Reduction to Dense Subset Sum). *If there exists a probabilistic polynomial-*

1877 *time adversary  $\mathcal{A}$  that can solve the modified Module-ISIS problem with addition in the*

1878 *Adh system with non-negligible probability, then there exists a probabilistic polynomial-*

1879 *time algorithm  $\mathcal{B}$  that can solve the dense subset sum problem with density above 0.9408[9]*

1880 *with non-negligible probability.*

1881 *Proof.* Suppose there exists an adversary  $\mathcal{A}$  that can solve the modified Module-ISIS

1882 problem with addition in the Adh system with non-negligible probability. We construct

1883 an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the dense subset sum problem. Given a dense

1884 subset sum instance  $(\mathbf{S}, t)$  with density above 0.94, where  $\mathbf{S} = s_1, \dots, s_n$  is a set of

1885 positive integers and  $t$  is a target sum,  $\mathcal{B}$  proceeds as follows:  $\mathcal{B}$  constructs a modified

1886 Module-ISIS instance  $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$  as follows: Set  $n = 128$  and  $m = 6$  according to

1887 the Adh system parameters. Construct a diagonal matrix  $\mathbf{A} = \text{diag}(s_1, \dots, s_n) \in \mathbb{Z}_q^{n \times n}$ ,

1888 where the elements of  $\mathbf{S}$  are placed on the main diagonal. Construct a target vector

1889  $\mathbf{t} = (t, 0, \dots, 0) \in \mathbb{Z}_q^n$ , where the first element is the target sum  $t$  and the remaining

1890 elements are zeros. Choose the modulus  $q$  and the norm bound  $\beta$  according to the Adh

1891 system parameters.  $\mathcal{B}$  invokes the adversary  $\mathcal{A}$  on the modified Module-ISIS instance

1892  $(\mathbf{A}, \mathbf{t}, q, n, m, \beta)$ . If  $\mathcal{A}$  outputs a solution vector  $\mathbf{z} = (z_1, \dots, z_n) \in \mathbb{Z}_q^n$  such that  $\mathbf{A} \cdot \mathbf{z} =$

1893  $\mathbf{t} \bmod q$  and  $\|\mathbf{z}\|_{\infty} \leq \beta$ , then  $\mathcal{B}$  outputs  $\mathbf{z}$  as a solution to the dense subset sum instance.

1894 The correctness of the reduction relies on the following observations: The diagonal matrix

1895  $\mathbf{A}$  constructed by  $\mathcal{B}$  preserves the density of the original dense subset sum instance.

1896 Since the elements of  $\mathbf{S}$  are placed on the main diagonal of  $\mathbf{A}$ , the resulting lattice has a

1897 density above 0.9408[9], mirroring the density of the subset sum instance. If the adversary

1898  $\mathcal{A}$  successfully solves the modified Module-ISIS instance, the solution vector  $\mathbf{z}$  satisfies

1899  $\mathbf{A} \cdot \mathbf{z} = \mathbf{t} \bmod q$ . Expanding this equation, we have:

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

1900 This implies that  $\sum_{i=1}^n s_i \cdot z_i = t \pmod q$ , which corresponds to a valid solution for the  
 1901 dense subset sum instance. Therefore, if an adversary can solve the modified Module-  
 1902 ISIS problem with addition in the Adh system with non-negligible probability, it would  
 1903 imply the existence of an efficient algorithm for solving the dense subset sum problem,  
 1904 contradicting the assumption that dense subset sum is computationally infeasible for  
 1905 density above 0.9408[9].  $\square$

## 1906 A.8 Quantum Hardness Estimation

1907 The reduction to the dense subset sum problem allows us to provide quantum hardness  
 1908 estimates for the Adh system. We consider two instances of the system, one with  $n = 128$   
 1909 and  $m = 6$ , and another with  $n = 256$  and  $m = 6$ . According to the improved classical  
 1910 and quantum algorithms for the subset sum problem, as presented by Bonnetain et al.  
 1911 [3], the quantum hardness of the subset sum problem with  $k$  elements is estimated to be  
 1912  $2^{0.216k}$ .

### 1913 A.8.1 Instance 1: $n = 128$ and $m = 6$

1914 In the case of an  $n = 128$  and  $m = 6$  module system, the total number of elements in the  
 1915 subset sum instance is:

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

1916 Applying the quantum hardness estimate to this instance, we have:

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

1917 Therefore, the quantum hardness of the Adh system with  $n = 128$  and  $m = 6$  is estimated  
 1918 to be approximately  $2^{166}$ .

### 1919 A.8.2 Instance 2: $n = 256$ and $m = 6$

1920 In the case of an  $n = 256$  and  $m = 6$  module system, the total number of elements in the  
 1921 subset sum instance is:

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

1922 Applying the quantum hardness estimate to this instance, we have:

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

1923 Therefore, the quantum hardness of the Adh system with  $n = 256$  and  $m = 6$  is estimated  
1924 to be approximately  $2^{332}$ . These quantum hardness estimates, based on the improved al-  
1925 gorithms by Bonnetain et al., provide an up-to-date assessment of the Adh system's  
1926 resistance against quantum attacks. The estimates suggest that solving the dense sub-  
1927 set sum problem corresponding to the Adh system instances would require a significant  
1928 amount of quantum resources, even with the current best-known quantum algorithms.

## 1929 A.9 Density Preservation in Module-ISIS to Module Modulus 1930 Subset Sum Reduction

1931 **Lemma 3.** *Let  $n = 128$ ,  $rank = 6$ ,  $\text{inf norm} = 257$ , and  $p = 257$ . Consider a module-*  
1932 *ISIS problem with a rejection filter regime that discards all vectors containing any 0s and*  
1933 *retries until a complete system is obtained. Changing the root of unity  $\omega$  from 3 to 1 in*  
1934 *the NTT is equivalent to relaxing the problem to addition. Under these conditions, the*  
1935 *module-ISIS problem reduces to a module modulus subset sum problem with  $128 \times 6 = 768$*   
1936 *elements, preserving the density.*

1937 *Proof.* In the module-ISIS problem, we have a rank 6 lattice with 6 public vectors  $\mathbf{A} =$   
1938  $(\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_6)$ , where each vector  $\mathbf{a}_i \in \mathbb{Z}_q^n$  and  $q = 257$ . The goal is to find a vector  
1939  $\mathbf{t} \in \mathbb{Z}_q^n$  such that  $\mathbf{t} = \mathbf{A}\mathbf{z} \bmod q$  for some coefficient vector  $\mathbf{z} \in \mathbb{Z}_q^{rank}$ . By applying the  
1940 rejection filter regime, we ensure that all vectors in the lattice have no 0 components,  
1941 maintaining a dense structure. The density of the lattice is preserved during this process.  
1942 When we change the root of unity  $\omega$  from 3 to 1 in the NTT, the modular multiplication  
1943 in the lattice is relaxed to addition. This relaxation does not affect the density of the  
1944 lattice, as the structure and the number of elements remain unchanged.

1945 The module-ISIS problem with  $\omega = 1$  can be viewed as a module modulus subset sum  
1946 problem. Each coefficient bucket in the NTT corresponds to an element in the subset  
1947 sum problem. Since we have  $n = 128$  and  $rank = 6$ , the total number of elements in the  
1948 subset sum problem is  $128 \times 6 = 768$ . Let  $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{768})$  be the set of elements in  
1949 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  
1950 to the target vector  $\mathbf{t}$  modulo  $q$ .

1951 The density of the subset sum problem is determined by the ratio of the number of  
1952 elements to the modulus  $q$ . In this case, the density is 1, which is the same as the density  
1953 of the original module-ISIS problem. Therefore, changing the root of unity from 3 to 1 in  
1954 the NTT and applying the rejection filter regime reduces the module-ISIS problem to a  
1955 module modulus subset sum problem with 768 elements while preserving the density.  $\square$

## 1956 A.10 Zero-Knowledge Proof

1957 **Theorem 18** (Zero-Knowledge Property). *The Adh zero-knowledge proof system satisfies*  
1958 *the zero-knowledge property, assuming the hardness of the Module-ISIS problem and the*  
1959 *existence of a secure commitment scheme.*

1960 *Proof.* We construct a simulator  $\mathcal{S}$  that generates proofs indistinguishable from real proofs  
1961 without access to the secret key. Given a public key  $\mathbf{pk}$  and a statement to be proved,  $\mathcal{S}$   
1962 proceeds as follows:

- 1963 1.  $\mathcal{S}$  generates a random commitment  $\mathbf{com}$  using the commitment scheme.
- 1964 2.  $\mathcal{S}$  computes the challenge  $\mathbf{sig\_chal}$  as a function of the statement and  $\mathbf{com}$  using  
1965 the Fiat-Shamir heuristic.
- 1966 3.  $\mathcal{S}$  samples a random vector  $\mathbf{sig\_rand}$  and computes the proof  $\mathbf{sig}$  as  
1967  $\mathbf{sig} = \text{ZKVolute}(\mathbf{pk}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$ .
- 1968 4.  $\mathcal{S}$  outputs the proof  $(\mathbf{sig}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$ .

1969 To show that the simulated proofs are indistinguishable from real proofs, we consider the  
1970 following hybrid arguments:

- 1971 • Hybrid 1: Real proofs generated using the secret key.
- 1972 • Hybrid 2: Proofs generated using the simulator  $\mathcal{S}$ .

1973 The indistinguishability of Hybrid 1 and Hybrid 2 relies on the following arguments:

- 1974 • The commitment scheme is hiding, ensuring that  $\mathbf{com}$  does not reveal any infor-  
1975 mation about the secret key.
- 1976 • The Fiat-Shamir heuristic ensures that the challenge  $\mathbf{sig\_chal}$  is uniformly dis-  
1977 tributed and independent of the secret key.
- 1978 • The ZKVolute function is a one-way function, assuming the hardness of the Module-  
1979 ISIS problem. Given  $\mathbf{pk}$ ,  $\mathbf{sig\_chal}$ , and  $\mathbf{sig\_rand}$ , it is computationally infeasible to  
1980 recover the secret key.

1981 Therefore, the proofs generated by the simulator  $\mathcal{S}$  are computationally indistinguishable  
1982 from real proofs, establishing the zero-knowledge property of the Adh system.  $\square$

## 1983 A.11 Algorithms

## 1984 A.12 Notes

1985 Algorithm variable notes:

1986 H is a hash function in the SHA3 family

1987 m is a theoretical message to be signed

1988 n is dimension

1989 p is the first level of NTT modulus

1990  $\omega$  is the first level of NTT root of unity

1991 k is the number of instances of module-ISIS problem to create

1992 ps is the array of NTT moduli in a multi stage instance

1993 ws is the array of related roots of unity

1994 l is number of 'levels' of unique NTT stage or len(ps)

1995 NTT\_DIST is the number of NTT representations to check before abort

## 1996 A.13 Module-ISIS+ Parameters

1997 n=128 or 256

1998 ps=[257,257]

1999 ws=[3,3]

2000 rnds=4

2001 sk\_count=1

2002 iters=4

2003  $\beta = 256$   
2004 rank = 6

## 2005 A.14 Module-ISIS\* Parameters

2006 n=128 or n=256  
2007 ps=[257,257]  
2008 ws=[3,3]  
2009 rnds=4  
2010 sk\_count=5  
2011 iters=4  
2012  $\beta = 256$   
2013 rank = 6

## 2014 A.15 Module-ISIS\*\* Parameters

2015 n=128 or n=256  
2016 ps=[257,257,65537]  
2017 ws=[3,3,282]  
2018 rnds=4  
2019 sk\_count=5  
2020 iters=4  
2021  $\beta = 256$   
2022 rank = 6  
2023

## 2024 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*  $\leftarrow$  *message*

**while** *True* **do**

*hash\_object*  $\leftarrow$  *hash\_algorithm*(*message*)

*hash\_value*  $\leftarrow$  *hash\_object.digest*(*size*)

*hash\_int*  $\leftarrow$  *int.from\_bytes*(*hash\_value*, *byteorder* = 'big')

*coefficients*  $\leftarrow$  []

**for** *i*  $\leftarrow$  0 to *size* - 1 **do**

*coefficients.append*(*int*(*hash\_value*[*i*]))

$\triangleright$  0-255

**end for**

**if** *poly\_check*(*coefficients*) = 0 **then**

**return** *coefficients*

**end if**

*message*  $\leftarrow$  *orig\_message*||*str*(*hash\_value*)

$\triangleright$  To keep input to 2x

**end while**

---

2025 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*

```
1: input_key  $\leftarrow$  tuple(vec), p, w
2: best  $\leftarrow$  vec.count(0)
3: best_vec  $\leftarrow$  vec.copy()
4: count  $\leftarrow$  0
5: for i  $\leftarrow$  0 to NTT_DIST do ▷ 2 for p = 257  $\omega$  = 3
6:   vec  $\leftarrow$  ntt(vec, p, w)
7:   if vec.count(0)  $\leq$  best then
8:     best_vec  $\leftarrow$  vec.copy()
9:   end if
10:  if vec.count(0) = 0 then ▷ Found suitable vector
11:    return (vec, count)
12:  end if
13:  count  $\leftarrow$  count + 1
14: end for
15: 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\_nonzero\_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_nonzero_vector(n, base_modulus)
  end while
  sk_I  $\leftarrow$  generate_non_zero_vector(n, base_modulus)
  while polycheck(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_nonzero_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 ← 1
sk_array ← []
while support ≠ 0 or pk_a.count(0) ≠ 0 or poly_check(pk_a) ≠ 0 do
  pk_chal ← generate_non_zero_vector(n, base_modulus)
  while poly_check(pk_chal) ≠ 0 do
    pk_chal ← generate_non_zero_vector(n, base_modulus)
  end while
  for _ ← 0 to rnds do
    sk_i ← generate_non_zero_vector(n, base_modulus)
    while poly_check(sk_i) ≠ 0 do
      sk_i ← generate_non_zero_vector(n, base_modulus)
    end while
    sk_array.append(sk_i)
  end for
  rand_pk ← generate_non_zero_vector(n, base_modulus)
  while poly_check(rand_pk) ≠ 0 do
    rand_pk ← generate_non_zero_vector(n, base_modulus)
  end while
  pk_a, support ← 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 ← Hash_To_Poly(m)
rand_sig ← generate_full_vector(n, base_modulus)
while rand_sig.count(0) ≠ 0 and polycheck(rand_sig) ≠ 0 do
  rand_sig ← generate_full_vector(n, base_modulus)
end while
SIG ← ZKVolute_ProofGen(sk_I, rand_sig, challenge_vector, False)
while SIG.count(0) ≠ 0 do
  rand_sig ← generate_full_vector(n, base_modulus)
  SIG ← 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
   $sk\_rep\_I \leftarrow select\_representation(sk\_I, p, w)$ 
   $rand\_chal \leftarrow select\_representation(rand\_chal, p, w)$ 
   $chal \leftarrow select\_representation(chal, p, w)$ 
   $iterables \leftarrow list()$ 
   $ntt\_rep \leftarrow chal$ 
   $blinded\_values \leftarrow list()$ 
   $root\_chal \leftarrow chal$ 
   $blinded\_values.append(root\_chal)$ 
  if  $iters > 0$  then
    for  $\_ \leftarrow 0$  to  $iters - 1$  do
       $ntt\_rep \leftarrow select\_representation(ntt\_rep, p, w)$ 
       $blinded\_values.append(ntt\_rep)$ 
       $iterables.append(ntt\_rep)$ 
    end for
    for  $z \leftarrow 1$  to  $iters - 1$  do
       $ntt\_rep \leftarrow pointwise\_mul(ntt\_rep, iterables[z], p)$ 
       $blinded\_values.append(ntt\_rep)$ 
    end for
     $chal \leftarrow ntt\_rep$ 
  end if
  if  $i = 0$  then
     $secret\_rep \leftarrow sk\_I$ 
     $target\_vector \leftarrow pointwise\_mul(chal, secret\_rep, p)$ 
     $proof\_rep \leftarrow pointwise\_mul(target\_vector, rand\_chal, p)$ 
     $new\_chal \leftarrow pointwise\_mul(root\_chal, rand\_chal, p)$ 
     $new\_chal \leftarrow pointwise\_add(new\_chal, chal, p)$ 
     $new\_chal \leftarrow pointwise\_add(new\_chal, rand\_chal, p)$ 
    for  $xx \leftarrow 0$  to  $rnds - 1$  do
       $proof\_rep \leftarrow pointwise\_mul(proof\_rep, root\_chal, p)$ 
       $proof\_rep \leftarrow pointwise\_mul(proof\_rep, new\_chal, p)$ 
       $proof\_rep \leftarrow pointwise\_mul(proof\_rep, iterables[xx \% iters], p)$ 
       $new\_chal \leftarrow pointwise\_mul(new\_chal, new\_chal, p)$ 
       $new\_chal \leftarrow pointwise\_add(new\_chal, new\_chal, ps[i])$ 
    end for
     $proof\_hld \leftarrow proof\_rep$ 
  end if
  if  $i \geq 1$  then
     $proof\_rep \leftarrow ntt(proof\_rep, p, w)$ 
     $proof\_hld \leftarrow ntt(proof\_hld, p, w)$ 
     $proof\_rep \leftarrow pointwise\_add(proof\_rep, proof\_rep, p)$ 
     $proof\_rep \leftarrow pointwise\_add(proof\_rep, proof\_hld, p)$ 
  end if
end for
for  $i, (p, w)$  in  $enumerate(reversed(list(zip(ps, ws))))$  do
  if  $i < len(ps) - 1$  then
     $proof\_rep \leftarrow ntt\_inverse(proof\_rep, p, w, original\_n = n)$ 
  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
   $sk\_rep\_array \leftarrow [best\_ntt(sk, p, w)[0]$  for  $sk$  in  $sk\_array]$ 
   $rand\_chal, _ \leftarrow best\_ntt(rand\_chal, p, w)$ 
   $chal, _ \leftarrow best\_ntt(chal, p, w)$ 
   $iterables \leftarrow list()$ 
   $ntt\_rep \leftarrow chal.copy()$ 
   $blinded\_values \leftarrow list()$ 
   $root\_chal \leftarrow chal.copy()$ 
   $blinded\_values.append(root\_chal)$ 
   $tmp\_iterable \leftarrow chal.copy()$ 
  if  $iters > 0$  then
    for  $_ \leftarrow 0$  to  $iters - 1$  do
       $tmp\_iterable, _ \leftarrow best\_ntt(tmp\_iterable, p, w)$ 
       $blinded\_values.append(tmp\_iterable)$ 
       $iterables.append(tmp\_iterable)$ 
    end for
    for  $z \leftarrow 0$  to  $iters - 1$  do
      if  $z = 0$  then
         $tmp\_iterable2 \leftarrow pointwise\_mul(root\_chal, iterables[0], p)$ 
         $blinded\_values.append(tmp\_iterable2)$ 
      else
         $tmp\_iterable2 \leftarrow pointwise\_mul(tmp\_iterable2, iterables[z], p)$ 
         $blinded\_values.append(tmp\_iterable2)$ 
      end if
    end for
     $chal \leftarrow tmp\_iterable2$ 
     $blinded\_values.append(chal)$ 
  end if
  if  $i = 0$  then
     $secret\_rep \leftarrow sk\_rep\_array[0]$ 
     $target\_vector \leftarrow pointwise\_mul(chal, secret\_rep, p)$ 
     $proof\_rep \leftarrow pointwise\_mul(target\_vector, rand\_chal, p)$ 
     $new\_chal \leftarrow pointwise\_mul(root\_chal, rand\_chal, p)$ 
     $new\_chal \leftarrow pointwise\_addition(new\_chal, chal, p)$ 
     $new\_chal \leftarrow pointwise\_addition(new\_chal, rand\_chal, p)$ 
    for  $xx \leftarrow 0$  to  $rnds - 1$  do
       $proof\_rep \leftarrow pointwise\_mul(proof\_rep, sk\_rep\_array[xx + 1], p)$ 
       $proof\_rep \leftarrow pointwise\_mul(proof\_rep, root\_chal, p)$ 
       $proof\_rep \leftarrow pointwise\_mul(proof\_rep, new\_chal, p)$ 
       $proof\_rep \leftarrow pointwise\_mul(proof\_rep, iterables[xx \bmod iters], p)$ 
       $new\_chal \leftarrow pointwise\_mul(new\_chal, new\_chal, p)$ 
       $new\_chal \leftarrow pointwise\_addition(new\_chal, new\_chal, p)$ 
    end for
     $proof\_hld \leftarrow proof\_rep$ 
  end if
  if  $i \geq 1$  then
     $proof\_rep \leftarrow ntt(proof\_rep, p, w)$ 
     $proof\_hld \leftarrow ntt(proof\_hld, p, w)$ 
     $proof\_rep \leftarrow pointwise\_addition(proof\_rep, proof\_rep, p)$ 
     $proof\_rep \leftarrow pointwise\_addition(proof\_rep, proof\_hld, p)$ 
  end if
end for
for  $i, (p, w)$  in enumerate(reversed(list(zip(ps, ws)))) do
  if  $i < len(ps) - 1$  then
     $proof\_rep \leftarrow ntt\_inverse(proof\_rep, p, w, original\_n = n)$ 
  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

2070 The tests were applied to the real and fake proof distributions, and the results were an-  
2071 alyzed to determine if there were any statistically significant differences between them.  
2072 Across millions of runs and various configurations, the statistical tests consistently demon-  
2073 strated the indistinguishability of real and fake proofs.

2074 The p-values obtained from the tests were consistently above the significance threshold  
2075 (e.g., 0.05), indicating that the null hypothesis (i.e., the distributions of real and fake  
2076 proofs are the same) cannot be rejected. The correlation coefficients between real and  
2077 fake proofs were close to zero, suggesting no significant correlation between them. The  
2078 mutual information between real and fake proofs was negligible, indicating minimal shared  
2079 information. The higher-order moments and autocorrelation tests further supported the  
2080 randomness and independence of the proofs.

### 2081 A.17.3 Machine Learning Test

2082 To further assess the distinguishability of real and fake proofs, we applied a gradient  
2083 boosting classifier to the proof data. The classifier was trained on a subset of the real  
2084 and fake proofs and then tested on a held-out set to evaluate its ability to distinguish  
2085 between them. Across multiple runs, the classifier consistently achieved an accuracy close  
2086 to 50%, indicating that it was unable to distinguish between real and fake proofs better  
2087 than random guessing. This result provides additional evidence for the zero-knowledge  
2088 property of the Adh system, as even advanced machine learning algorithms were unable  
2089 to differentiate between the two types of proofs.

### 2090 A.17.4 Empirical Conclusion

2091 The empirical evidence obtained from the simulator-based approach and the statistical  
2092 tests provides compelling support for the presence of the zero-knowledge property in  
2093 the Adh system. The extensive testing, covering a wide range of configurations and a  
2094 large number of proofs, demonstrates the consistent indistinguishability of real and fake  
2095 proofs. The inability to forge valid proofs and the resistance to advanced distinguishing  
2096 techniques further strengthen the case for the zero-knowledge property.

2097 While a formal mathematical proof of the zero-knowledge property is still pending,  
2098 the empirical results obtained from this rigorous experimental setup strongly suggest  
2099 that the Adh system achieves zero-knowledge. The simulator-based approach, combined  
2100 with comprehensive statistical testing and machine learning analysis, provides a robust  
2101 framework for assessing the zero-knowledge property and lays the foundation for fur-  
2102 ther theoretical analysis and formal proofs. The detailed experimental setup and results  
2103 supporting the zero-knowledge property are provided here.

## 2104 A.18 Zero-Knowledge Proof

2105 **Theorem 19** (Zero-Knowledge Property). *The Adh zero-knowledge proof system satisfies*  
2106 *the zero-knowledge property, assuming the hardness of the Module-ISIS problem and the*  
2107 *existence of a secure commitment scheme.*

2108 *Proof.* We construct a simulator  $\mathcal{S}$  that generates proofs indistinguishable from real proofs  
2109 without access to the secret key. Given a public key  $\mathbf{pk}$  and a statement to be proved,  $\mathcal{S}$   
2110 proceeds as follows:

- 2111 1.  $\mathcal{S}$  generates a random commitment  $\mathbf{com}$  using the commitment scheme.

- 2112 2.  $\mathcal{S}$  computes the challenge  $\mathbf{sig\_chal}$  as a function of the statement and  $\mathbf{com}$  using  
 2113 the Fiat-Shamir heuristic.  
 2114 3.  $\mathcal{S}$  samples a random vector  $\mathbf{sig\_rand}$  and computes the proof  $\mathbf{sig}$  as:

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

- 2115 4.  $\mathcal{S}$  outputs the proof  $(\mathbf{sig}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$ .

2116 To show that the simulated proofs are indistinguishable from real proofs, we consider the  
 2117 following hybrid arguments:

- 2118 • Hybrid 1: Real proofs generated using the secret key.
- 2119 • Hybrid 2: Proofs generated using the simulator  $\mathcal{S}$ .

2120 We now argue that Hybrid 1 and Hybrid 2 are computationally indistinguishable based  
 2121 on the following:

- 2122 • The commitment scheme is computationally hiding, ensuring that  $\mathbf{com}$  does not  
 2123 reveal any information about the secret key to a computationally bounded adver-  
 2124 sary.
- 2125 • The Fiat-Shamir heuristic ensures that the challenge  $\mathbf{sig\_chal}$  is uniformly dis-  
 2126 tributed and independent of the secret key, assuming the random oracle model.
- 2127 • The ZKVolute function is a one-way function, assuming the hardness of the Module-  
 2128 ISIS problem. Given  $\mathbf{pk}$ ,  $\mathbf{sig\_chal}$ , and  $\mathbf{sig\_rand}$ , it is computationally infeasible to  
 2129 recover the secret key.

2130 Suppose there exists a polynomial-time distinguisher  $\mathcal{D}$  that can distinguish between  
 2131 Hybrid 1 and Hybrid 2 with non-negligible advantage. We construct a polynomial-time  
 2132 adversary  $\mathcal{A}$  that uses  $\mathcal{D}$  to break either the hiding property of the commitment scheme  
 2133 or the one-wayness of the ZKVolute function.

2134  $\mathcal{A}$  receives a public key  $\mathbf{pk}$  and a statement to be proved. It then generates two proofs,  
 2135 one using the real prover algorithm and one using the simulator  $\mathcal{S}$ .  $\mathcal{A}$  sends the two proofs  
 2136 to the distinguisher  $\mathcal{D}$ . If  $\mathcal{D}$  can distinguish between the real and simulated proofs with  
 2137 non-negligible advantage, then  $\mathcal{A}$  can use this to break either the hiding property of the  
 2138 commitment scheme or the one-wayness of the ZKVolute function, depending on  $\mathcal{D}$ 's out-  
 2139 put. This contradicts the assumptions of a secure commitment scheme and the hardness  
 2140 of Module-ISIS. Therefore, the proofs generated by the simulator  $\mathcal{S}$  are computationally  
 2141 indistinguishable from real proofs, establishing the zero-knowledge property of the Adh  
 2142 system.  $\square$

## 2143 A.19 Probabilistic Completeness

2144 **Theorem 20** (Probabilistic Completeness). *Let  $\mathcal{A}$  be the Adh zero-knowledge proof sys-*  
 2145 *tem with dimension  $n$ , norm bound  $\beta$ , and a fixed challenge vector  $\mathbf{c}$ . If the prover has*  
 2146 *a probability  $p$  of passing the rejection sampling step for a given random vector, then the*  
 2147 *probability of finding a valid proof for the fixed challenge  $\mathbf{c}$  approaches 1 as the number*  
 2148 *of attempts grows exponentially with respect to  $n$ .*

2149 *Proof.* Consider a scenario where the prover has a fixed challenge vector  $\mathbf{c}$  and needs  
 2150 to generate a valid proof. The prover selects a random vector  $\mathbf{r}$  of dimension  $n$  with  
 2151 coefficients bounded by the norm  $\beta$ . The prover then attempts to generate a proof by  
 2152 passing  $\mathbf{r}$  through the rejection sampling step. Let  $p$  be the probability of the prover  
 2153 passing the rejection sampling step for a given random vector  $\mathbf{r}$ . If the prover fails the

2154 rejection sampling step, they simply select a new random vector and try again. The  
 2155 probability of failing to find a valid proof after  $k$  attempts is given by:

$$P(\text{failure after } k \text{ attempts}) = (1 - p)^k \quad (43)$$

2156 As the number of attempts  $k$  grows, the probability of failure decreases exponentially.  
 2157 In the Adh system, the dimension  $n$  is typically chosen to be either 128 or 256, and the  
 2158 norm bound  $\beta$  is set to 257. For  $n = 128$ , the prover has  $257^{128}$  possible random vectors  
 2159 to choose from. Even with a conservative probability of passing the rejection sampling  
 2160 step, say  $p = 0.05$ , the probability of failure after  $k$  attempts is:

$$P(\text{failure after } k \text{ attempts}) = (1 - 0.05)^k = 0.95^k \quad (44)$$

2161 As  $k$  approaches  $257^{128}$ , the probability of failure becomes negligibly small. Similarly, for  
 2162  $n = 256$ , the prover has  $257^{256}$  possible random vectors to choose from. With the same  
 2163 conservative probability  $p = 0.05$ , the probability of failure after  $k$  attempts is:

$$P(\text{failure after } k \text{ attempts}) = (1 - 0.05)^k = 0.95^k \quad (45)$$

2164 As  $k$  approaches  $257^{256}$ , the probability of failure becomes even smaller. Therefore, given  
 2165 the extremely large number of possible random vectors and the ability of the prover to  
 2166 repeatedly attempt rejection sampling, the probability of finding a valid proof for a fixed  
 2167 challenge vector approaches 1. While this argument does not provide an absolute proof of  
 2168 completeness, it demonstrates that the Adh system achieves a strong form of probabilistic  
 2169 completeness. The chances of the prover failing to find a valid proof for a given challenge  
 2170 are negligibly small, assuming a reasonable probability of passing the rejection sampling  
 2171 step.  $\square$

2172 This probabilistic completeness argument highlights the effectiveness of the rejection  
 2173 sampling technique used in the Adh system. By allowing the prover to repeatedly select  
 2174 new random vectors until a valid proof is found, the system ensures that the prover can  
 2175 successfully generate proofs for any given challenge with overwhelming probability. The  
 2176 conservative estimate of a 5% success probability for each attempt further strengthens the  
 2177 argument, as the actual success probability in the Adh system is typically much higher  
 2178 (closer to 60% empirically). This means that the prover can find a valid proof with even  
 2179 fewer attempts in practice.

2180 Rejection sampling is also applied during the challenge generation process on the  
 2181 hash of message as  $m$ . If  $m$  produces a *chal* that fails the rejection sampling test,  $m$   
 2182 is first copied to a temporary variable  $h\_val$  and a loop where  $h\_val \leftarrow H(m||h\_val)$  is  
 2183 iterated with no maximum number of attempts. If the *chal* that is produced by  $h\_val$   
 2184 passes rejection sampling the loop terminates. As the number of attempts is essentially  
 2185 unbounded, this intuitive result is not formally proven under random oracle assumptions.

2186 The completeness of the Adh system relies on the vast number of possible random  
 2187 vectors and the efficiency of the rejection sampling process. As the dimension  $n$  and the  
 2188 norm bound  $\beta$  increase, the probability of failure diminishes rapidly, providing a strong  
 2189 assurance of completeness. While this probabilistic argument may not constitute an  
 2190 absolute proof of completeness, it provides a compelling justification for the completeness  
 2191 property of the Adh system based on the overwhelming likelihood of success.

2192 **Conjecture 5** (Unlikelihood of Violating Shannon-Nyquist Sampling Theorem in the  
 2193 Adh System). *The recent advancements in quantum algorithms for solving the Learning with Errors (LWE) problem, particularly the use of Gaussian functions with complex*

2195 variances and the exploitation of the Karst wave feature in the Quantum Fourier Trans-  
2196 form (QFT) domain, have raised concerns about the potential impact on the security of  
2197 lattice-based cryptographic systems like the Adh zero-knowledge proof system.

2198 However, it is important to consider the fundamental principles of information theory,  
2199 such as the Shannon-Nyquist[11] sampling theorem, when assessing the likelihood of a  
2200 quantum computer being able to violate these principles in the context of the Adh system.  
2201 The Shannon-Nyquist sampling theorem states that a signal can be perfectly reconstructed  
2202 from its samples if the sampling rate is at least twice the highest frequency component  
2203 in the signal. In the context of the Adh system, which employs the Number Theoretic  
2204 Transform (NTT) for polynomial multiplication, the NTT can be viewed as a form of  
2205 sampling in the frequency domain. Given the structure and parameters of the Adh system,  
2206 it seems **unlikely** that a quantum computer, even with the advanced techniques like the  
2207 Karst wave, would be able to violate the Shannon-Nyquist sampling theorem and perfectly  
2208 reconstruct the undersampled signal in the NTT domain. The reasons for this assessment  
2209 are as follows:

- 2210 • The Adh system operates over finite fields, and the NTT is a discrete transform  
2211 that preserves the algebraic structure of the underlying ring. The sampling rate in  
2212 the NTT domain is determined by the choice of parameters and the structure of the  
2213 polynomial ring.
- 2214 • The security of the Adh system relies on the hardness of the Module-ISIS problem,  
2215 which is based on finding short integer solutions to linear equations. The problem  
2216 is designed to be computationally infeasible, even for quantum computers, when the  
2217 parameters are appropriately chosen.
- 2218 • The use of rejection sampling and the careful selection of parameters in the Adh  
2219 system ensure that the resulting lattices have a high dimension and a large minimum  
2220 distance, making it difficult for any algorithm, including quantum algorithms, to find  
2221 short vectors and solve the underlying Module-ISIS problem.

2222 While the Karst wave technique exploits certain periodic patterns in the QFT domain, it  
2223 is not clear whether such patterns exist or can be efficiently exploited in the NTT domain  
2224 of the Adh system. Furthermore, even if such patterns were found, it is unlikely that they  
2225 would enable a quantum computer to violate the Shannon-Nyquist sampling theorem and  
2226 perfectly reconstruct the undersampled signal.

2227 In this updated conjecture, we emphasize the unlikelihood of a quantum computer  
2228 being able to violate the Shannon-Nyquist sampling theorem in the context of the Adh  
2229 system. We highlight the reasons behind this assessment, including the discrete nature  
2230 of the NTT, the hardness of the underlying Module-ISIS problem, and the careful pa-  
2231 rameter selection and rejection sampling techniques used in the Adh system. However,  
2232 we also acknowledge the rapid evolution of the field of quantum computing and the pos-  
2233 sibility of new techniques and insights emerging in the future. We stress the importance  
2234 of maintaining a cautious approach, actively monitoring developments, and conducting  
2235 regular security assessments to ensure the long-term security of the Adh system against  
2236 potential quantum threats.

## 2237 A.20 Proof of Completeness

2238 **Theorem 21** (Completeness). *The Adh zero-knowledge proof system is complete. That*  
2239 *is, an honest prover can always convince the verifier of a true statement.*



2240 *Proof.* Let  $(\mathbf{pk}, \mathbf{sk})$  be a valid key pair generated by the key generation algorithm of the  
 2241 Adh system, where  $\mathbf{pk}$  is the public key and  $\mathbf{sk}$  is the secret key. Let  $m$  be a message  
 2242 and  $\mathbf{sig\_chal}$  be the signature challenge derived from  $m$ . An honest prover, possessing  
 2243 the secret key  $\mathbf{sk}$ , generates a proof  $(\mathbf{sig}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$  as follows:

- 2244 1. Generate a uniformly random signature randomness  $\mathbf{sig\_rand} \in R_q^m$  with coeffi-  
 2245 cients in the range  $[1, q - 1]$ .
- 2246 2. Apply rejection sampling to ensure that  $\mathbf{sig\_rand}$  is a full vector.
- 2247 3. Compute the proof  $\mathbf{sig}$  as  $\mathbf{sig} = \text{ZKVolute}(\mathbf{sk}, \mathbf{sig\_chal}, \mathbf{sig\_rand})$ .

2248 The verifier checks the validity of the proof by computing:

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

2249 and verifying that  $\mathbf{lhs} = \mathbf{rhs}$ . By the construction of the Adh system and the properties  
 2250 of the ZKVolute function, we have:

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

2251

□

2252 Therefore, an honest prover, possessing the secret key  $\mathbf{sk}$ , can always generate a valid  
 2253 proof that convinces the verifier, proving the completeness of the Adh zero-knowledge  
 2254 proof system

## 2255 **A.21 Conjecture on Entropy Expansion and Information Loss** 2256 **in Module-ISIS\*\* with Higher-Dimensional NTT Mixing** 2257 **and Reduction**

2258 **Conjecture 6** (Entropy Expansion and Information Loss in Module-ISIS\*\* with High-  
 2259 er-Dimensional NTT Mixing and Reduction). *Let  $\mathcal{L}$  be an instance of the Module-ISIS\*\**  
 2260 *problem with a prime modulus  $p_1 = 257$  (in a zero free regime) and a higher-dimensional*  
 2261 *prime modulus  $p_2 = 65537$ . Let  $\mathbf{x} \in \mathbb{Z}_{p_1}^n$  be a vector representing a proof in the Adh*  
 2262 *system, and let  $H(\mathbf{x})$  denote the Shannon entropy of  $\mathbf{x}$ . Consider the following transfor-*  
 2263 *mation:*

- 2264 1. Compute the NTT representation of  $\mathbf{x}$  in the field  $\mathbb{Z}_{p_1}$ , denoted as  $\mathbf{X1} = \text{NTTp}_1(\mathbf{x})$ .
- 2265 2. Forward transform  $\mathbf{X1}$  to the field  $\mathbb{Z}_{p_2}$ , denoted as  $\mathbf{X2} = \text{NTTp}_2(\mathbf{X1})$ .
- 2266 3. Perform modular addition of  $\mathbf{X2}$  with itself in the field  $\mathbb{Z}_{p_2}$ , denoted as  $\mathbf{Y2} =$   
 2267  $\mathbf{X2} \oplus \mathbf{X2}$ , where  $\oplus$  represents element-wise modular addition.
- 2268 4. Invert the NTT representation of  $\mathbf{Y2}$  back to the field  $\mathbb{Z}_{p_1}$ , denoted as  $\mathbf{y} = \text{INTTp}_1(\mathbf{Y2})$ .

2269 We conjecture that the modular reduction from the higher-dimensional field  $\mathbb{Z}_{p_2}$  back to  
 2270 the original field  $\mathbb{Z}_{p_1}$  is the primary cause of the observed high entropy in the output vector  
 2271  $\mathbf{y}$ . The fact that the Shannon entropy of  $\mathbf{y}$  approaches a nearly perfect 8 bits per element,  
 2272 which is the maximum possible entropy for elements in  $\mathbb{Z}_{257}$  with a 257 norm, suggests  
 2273 that the modular reduction step may lead to a significant loss of structural information  
 2274 about the underlying lattice.

2275 *During the transformation process, the structural information of the lattice is expanded*  
2276 *to extra dimensions in the higher-dimensional field  $\mathbb{Z}_{p_2}$ . The modular addition of  $\mathbf{X}_2$  with*  
2277 *itself further obfuscates the lattice structure by mixing and folding the information onto*  
2278 *itself. When this expanded and obfuscated representation is then reduced back to the*  
2279 *original field  $\mathbb{Z}_{p_1}$ , a substantial amount of critical structural data needed for inversion*  
2280 *may be randomly lost due to the modular reduction.*

2281 *The apparent loss of structural information during the modular reduction step could*  
2282 *potentially preclude the inversion of the transformation altogether. If the entropy of the*  
2283 *output vector  $\mathbf{y}$  approaches the maximum possible value, it suggests that the information*  
2284 *content of  $\mathbf{y}$  is nearly uniform and lacks any discernible structure. This loss of struc-*  
2285 *ture may make it infeasible to recover the original vector  $\mathbf{x}$  from  $\mathbf{y}$ , as the information*  
2286 *necessary for inversion may have been irretrievably lost during the modular reduction.*

2287 *The observed entropy expansion and the potential loss of critical structural information*  
2288 *during the modular reduction step may have significant implications for the hardness of the*  
2289 *Module-ISIS\*\* problem. If the transformation process destroys the structural properties*  
2290 *of the lattice that could be exploited by adversaries, it may enhance the security of the*  
2291 *Adh system by making it more resistant to lattice-based attacks.*

2292 *However, it is important to note that this conjecture is based on empirical observations*  
2293 *and requires formal verification. Further research is needed to rigorously analyze the*  
2294 *relationship between the entropy expansion, the loss of structural information, and the*  
2295 *hardness of the Module-ISIS\*\* problem. Additionally, the precise impact of the modular*  
2296 *reduction step on the invertibility of the transformation should be investigated to determine*  
2297 *the feasibility of recovering the original vector  $\mathbf{x}$  from the output vector  $\mathbf{y}$ .*

2298 *If validated, this conjecture would provide additional support for the security of the*  
2299 *Adh system and highlight the potential benefits of incorporating higher-dimensional NTT*  
2300 *mixing and reduction in lattice-based cryptographic constructions. The loss of structural*  
2301 *information during the modular reduction step may introduce an additional layer of com-*  
2302 *plexity that enhances the resistance of the system against potential attacks.*

## 2303 **A.22 There is No Dual**

2304 **Conjecture 7.** *Assume a cryptographic lattice-based system that is designed to produce*  
2305 *a complete lattice under operationally defined conditions. If the lattice is complete, then*  
2306 *the dual lattice associated with this system is empty in the sense that it contains no small*  
2307 *or practically useful vectors under computational feasibility constraints.*

2308 **Proof.** *Given that the lattice  $L$  is complete, every vector in  $L$  contributes to filling*  
2309 *the entire  $n$ -dimensional space without gaps. By the construction of such a system, the*  
2310 *density of the lattice in the primal space is maximized, implying that the minimal distance*  
2311 *between lattice points is at its theoretical lower bound.*

2312 *This maximal packing in the primal lattice leads to a minimal or non-existent set of*  
2313 *vectors in the dual lattice  $L^*$  that can be exploited computationally. Specifically, the*  
2314 *vectors in  $L^*$  that are typically targeted in dual lattice attacks (i.e., short vectors) are*  
2315 *either too large to be used practically or are non-existent due to the inversion properties*  
2316 *of the Fourier transform applied in constructing  $L$ .*

2317 *Therefore, in the operational context of cryptographic computation where practicality*  
2318 *and computational feasibility are key, the dual lattice can be considered effectively empty*  
2319 *of useful vectors for cryptanalysis. Measurements show the effective bound for dual*

2320 vectors is  $> 1$ . This results in a robust defense mechanism against dual lattice attacks,  
2321 enhancing the cryptographic security of the system.

### 2322 **A.23 Potential for Transition to Anti-Cyclic Matrices**

2323 In our research on the Adh zero-knowledge proof system, we have extensively utilized  
2324 the prime moduli  $p = 257$  and  $p = 65537$  in our algorithms and implementations. These  
2325 primes have been chosen for their desirable properties, such as being Fermat primes of  
2326 the form  $2^k + 1$ , which enable efficient polynomial arithmetic and the use of the Number  
2327 Theoretic Transform (NTT) for fast operations.

2328 However, recent advancements in lattice cryptanalysis have highlighted potential vul-  
2329 nerabilities associated with the use of cyclic matrices and the underlying algebraic struc-  
2330 ture of the ring of polynomials modulo  $x^n - 1$ . While our current design incorporates  
2331 techniques such as extensive rejection sampling and a chaining construction to amplify  
2332 complexity and destroy patterns, it is important to consider the potential benefits of  
2333 transitioning to anti-cyclic matrices. Anti-cyclic matrices, which correspond to the ring  
2334 of polynomials modulo  $x^n + 1$ , have been shown to provide stronger security guarantees  
2335 compared to cyclic matrices in lattice-based cryptography. The irreducibility of  $x^n + 1$   
2336 when  $n$  is a power of 2 ensures that the resulting lattice has a dense representation and  
2337 does not exhibit any obvious weaknesses that could be exploited by an attacker.

2338 If a transition to anti-cyclic matrices is deemed necessary based on further security  
2339 analysis and research, our existing algorithms and codebase can be adapted to accommo-  
2340 date this change. The modifications required to switch from cyclic to anti-cyclic matrices  
2341 are relatively straightforward, primarily involving polynomial arithmetic operations. In  
2342 terms of the choice of parameters, our current use of  $p = 257$  and  $p = 65537$  can be main-  
2343 tained even with the transition to anti-cyclic matrices. These primes remain suitable for  
2344 the anti-cyclic setting, providing the necessary security properties and enabling efficient  
2345 computations.

2346 However, it is important to conduct a thorough security analysis to assess the impact  
2347 of the transition to anti-cyclic matrices on the overall security of the Adh system. This  
2348 analysis should take into account the specific attack scenarios, the best-known algorithms  
2349 for solving the underlying lattice problems, and the latest advances in lattice cryptanaly-  
2350 sis. If the security analysis reveals significant vulnerabilities in the current design that can  
2351 be mitigated by the transition to anti-cyclic matrices, and the improvements in security  
2352 outweigh any potential impact on efficiency and performance, then making the switch to  
2353 anti-cyclic matrices may be justified.

2354 In conclusion, while our current design extensively utilizes the primes  $p = 257$  and  
2355  $p = 65537$ , we are prepared to adapt our algorithms and codebase to support anti-  
2356 cyclic matrices if necessary. The transition to anti-cyclic matrices can be achieved with  
2357 relatively minor modifications, and our chosen primes remain suitable for the anti-cyclic  
2358 setting. However, a comprehensive security analysis is essential to determine the necessity  
2359 and benefits of such a transition. By carefully evaluating the results of this analysis  
2360 and considering the specific requirements of the Adh system, we can make an informed  
2361 decision on whether the transition to anti-cyclic matrices is warranted for the long-term  
2362 security and practicality of our zero-knowledge proof system.

## 2363 **A.24 Consideration of Ajita’s Bound in the Adh System**

2364 Given the foundational security of the Adh system is not predicated on the use of short  
2365 keys, the introduction of Ajita’s bound presents an intriguing avenue for enhancing the  
2366 theoretical robustness of our cryptographic scheme. Ajita’s bound typically applies to  
2367 systems requiring compact key sizes, which isn’t a necessity for our system but offers  
2368 potential improvements with no detected impacts on system performance.

## 2369 **A.25 Formal Analysis of Short Key Implications under Ajita’s** 2370 **Bound**

2371 The Adh cryptographic system primarily employs regular key lengths, where the security  
2372 model does not inherently rely on the use of short keys. In lattice-based cryptography,  
2373 ”short” typically refers to the Euclidean norm of lattice vectors, with shorter vectors often  
2374 equated to reduced security risk due to increased difficulty in computational attacks.  
2375 Here, we explore the theoretical and practical implications of applying Ajita’s bound to  
2376 assess the viability and impact of using short keys within the Adh system.

### 2377 **A.25.1 Theoretical Framework**

2378 Ajita’s bound provides a measure for the security degradation as the Euclidean norm  
2379 of the keys decreases. Formally, let  $n$  denote the Euclidean norm of a key. For full  $\beta$   
2380 width keys in the Adh system,  $n = n_{\text{reg}}$ , and for short keys,  $n = n_{\text{short}}$ . Ajita’s bound is  
2381 expressed as:

$$\epsilon(n) \text{ is negligible, where } \epsilon(n) \rightarrow 0 \text{ as } n \rightarrow \infty. \quad (46)$$

2382 This bound implies that the security loss  $\epsilon(n)$  decreases inversely with the increase in  
2383 the norm, asserting that larger norms (regular keys) enhance security but at a cost of  
2384 increased computation.

### 2385 **A.25.2 Implications for Adh System**

2386 In the Adh system, transitioning to traditional short vectors for secrets could theoret-  
2387 ically be implemented with negligible impact on security if the norm reduction does not  
2388 significantly elevate  $\epsilon(n)$ . The relationship between the advantages for adversaries using  
2389 short and regular keys can be formalized as:

$$\text{Adv}_{\text{Adh}}(\lambda, n_{\text{short}}) \leq \text{Adv}_{\text{Adh}}(\lambda, n_{\text{reg}}) + \epsilon(n_{\text{short}}), \quad (47)$$

2390 where  $\lambda$  is the security parameter. This inequality indicates that the advantage an ad-  
2391 versary gains by the system’s use of short keys is strictly bounded by the sum of the  
2392 advantage with regular keys plus a negligible term  $\epsilon(n_{\text{short}})$ .

### 2393 **A.25.3 Practical Considerations**

2394 Practical deployment of short vectors within the Adh system requires empirical validation  
2395 to ensure that the theoretical bounds hold under real-world conditions. Preliminary  
2396 experimental results suggest that the Adh system maintains robust security metrics even  
2397 when the key norms are reduced dramatically. This observation supports the feasibility  
2398 of integrating short vector keys, potentially reducing computational overhead without  
2399 compromising the cryptographic integrity.

#### 2400 A.25.4 Conclusion

2401 The exploration of Ajita’s bound in the context of the Adh system highlights that short  
 2402 vector keys could be employed without significant security concessions. The formal anal-  
 2403 ysis suggests that, with appropriate empirical support, short vector secrets might be a  
 2404 useful modification where secret storage space is limited.

#### 2405 A.26 BKZ Cost Estimate

2406 **Conjecture 8** (Adjusted Efficiency Constant for BKZ in a 0-Free, Maximum Density  
 2407 Lattice). *Let  $\mathcal{L}$  be a lattice with dimension  $n$ , constructed under a "0-free regime" and  
 2408 exhibiting "maximum density". Let  $c_{base}$  denote the base value of the efficiency constant  
 2409 for the BKZ algorithm, typically chosen as  $c_{base} = 0.292$  based on empirical studies and  
 2410 common usage in the lattice-based cryptography community. We conjecture that the ad-  
 2411 justed efficiency constant  $c_{adj}$  for estimating the computational cost of BKZ in the context  
 2412 of  $\mathcal{L}$  should be increased by 20% to 30% relative to  $c_{base}$ . Specifically:*

$$c_{adj} \in [1.20 \times c_{base}, 1.30 \times c_{base}] \approx [0.3504, 0.3796] \quad (48)$$

2413 The justification for this adjustment is as follows:

- 2414 1. The "0-free regime" of  $\mathcal{L}$  significantly increases the complexity of the lattice reduc-  
 2415 tion process by eliminating trivially short vectors. This feature alone suggests an  
 2416 increase in the efficiency constant by 10% to 20%.
- 2417 2. The "maximum density" property of  $\mathcal{L}$  further contributes to the hardness of the  
 2418 lattice, making it more challenging to distinguish between vectors. This character-  
 2419 istic also warrants an increase in the efficiency constant by approximately 10% to  
 2420 20%.
- 2421 3. The cumulative effect of both features, while not strictly additive, can be conser-  
 2422 vatively estimated to result in a total increase of 20% to 30% over the base value  
 2423  $c_{base}$ .

2424 This adjusted efficiency constant  $c_{adj}$  provides a more conservative estimate of the com-  
 2425 putational cost required to achieve lattice reduction in the specific context of  $\mathcal{L}$ . By ac-  
 2426 counting for the increased hardness introduced by the "0-free" and "maximum density"  
 2427 properties, the adjusted value helps to ensure a robust security margin against advanced  
 2428 lattice reduction techniques. **Note:** The exact value of  $c_{adj}$  within the conjectured range  
 2429 may be further refined based on empirical data and specific implementation details of the  
 2430 BKZ algorithm in the context of  $\mathcal{L}$ .

#### 2431 A.27 Distribution Analysis

2432 **Conjecture 9** (Uniform Distribution of Coefficients in the Adh Cryptographic System).  
 2433 *Let  $\mathcal{A}$  be the Adh cryptographic system with the following parameters:*

- 2434 • *Dimension:*  $n \in 128$
- 2435 • *Number of rounds:*  $rnds = 4$
- 2436 • *Number of iterations:*  $iters = 4$
- 2437 • *Prime moduli:*  $ps = [257, 257, 65537]$
- 2438 • *Roots of unity:*  $ws = [3, 3, 282]$
- 2439 • *Second Roots of unity:*  $ws2 = [1, 1, 1]$

2440 *For any key pair  $(pk, sk)$  generated by  $\mathcal{A}$ , the coefficient values  $1, 2, \dots, 256$  in the vectors  
 2441 produced by  $\mathcal{A}$  using  $(pk, sk)$  are uniformly distributed.*

2442 **Justification:** To support the conjecture of uniform distribution of coefficients in  
 2443 the Adh cryptographic system, an extensive experimental analysis was conducted. The  
 2444 experimental design and results are as follows: **Experimental Design:**

- 2445 • Four unique key pairs were generated using the seeds 950001, 950002, 950003, and  
 2446 950004.
- 2447 • For each key pair, over 100 million vectors were generated using the Adh crypto-  
 2448 graphic system with the specified parameters.
- 2449 • The uniformity of the coefficient distribution was assessed using chi-square tests for  
 2450 each individual key pair and the combined dataset.
- 2451 • Finally a second test was run against 338M vectors using *ws2*, as evidence support-  
 2452 ing the assumption that uniform distribution also applies to the subset reduction,  
 2453 noted as  $\omega = 1$  in the table below.

2454 **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

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	0.999999999983357

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

2455 the extremely high p-values (all greater than 0.9999) strongly support the hypothesis of  
 2456 uniform distribution. The p-values indicate that the observed coefficient distributions  
 2457 are highly consistent with the expected uniform distribution. The experimental results  
 2458 provide strong empirical evidence supporting the conjecture that the Adh cryptographic  
 2459 system produces vectors with uniformly distributed coefficients between 1 and 256. This  
 2460 uniformity property is crucial for ensuring the security and effectiveness of cryptographic  
 2461 protocols built upon the Adh system.  
 2462

2463 The assumption of uniform coefficient distribution is well-justified based on the rig-  
 2464 orous experimental analysis conducted across multiple key pairs and a large sample size  
 2465 of generated vectors. The chi-square tests and visual inspections consistently validate  
 2466 the uniformity of the coefficient values, providing a solid foundation for the security and  
 2467 reliability of the Adh cryptographic system.

## 2468 Acknowledgments

2469 The authors would like to acknowledge that the cryptographic system described herein  
 2470 is currently patent pending.

## 2471 Bibliography

- 2472 [1] Henry Bambury and Phong Q. Nguyen. *Improved Provable Reduction of NTRU*  
 2473 *and Hypercubic Lattices*. Cryptology ePrint Archive, Paper 2024/601. [https://](https://eprint.iacr.org/2024/601)  
 2474 [eprint.iacr.org/2024/601](https://eprint.iacr.org/2024/601). 2024. URL: <https://eprint.iacr.org/2024/601>.

- 2475 [2] Daniel J. Bernstein and Bo-Yin Yang. “Asymptotically Faster Quantum Algorithms  
2476 to Solve Multivariate Quadratic Equations”. In: *Post-Quantum Cryptography*. Ed.  
2477 by Tanja Lange and Rainer Steinwandt. Cham: Springer International Publishing,  
2478 2018, pp. 487–506. ISBN: 978-3-319-79063-3.
- 2479 [3] Xavier Bonnetain et al. *Improved Classical and Quantum Algorithms for Subset-  
2480 Sum*. Cryptology ePrint Archive, Paper 2020/168. [https://eprint.iacr.org/  
2481 2020/168](https://eprint.iacr.org/2020/168). 2020. URL: <https://eprint.iacr.org/2020/168>.
- 2482 [4] Léoucas, Thomas Espitau, and Eamonn W. Postlethwaite. *Finding short integer  
2483 solutions when the modulus is small*. Cryptology ePrint Archive, Paper 2023/1125.  
2484 <https://eprint.iacr.org/2023/1125>. 2023. URL: [https://eprint.iacr.org/  
2023/1125](https://eprint.iacr.org/<br/>2485 2023/1125).
- 2486 [5] Jianwei Li and Phong Q. Nguyen. *A Complete Analysis of the BKZ Lattice Reduc-  
2487 tion Algorithm*. Cryptology ePrint Archive, Paper 2020/1237. [https://eprint.  
2488 iacr.org/2020/1237](https://eprint.iacr.org/2020/1237). 2020. URL: <https://eprint.iacr.org/2020/1237>.
- 2489 [6] Yehuda Lindell. “How To Simulate It - A Tutorial on the Simulation Proof Tech-  
2490 nique”. In: *Electron. Colloquium Comput. Complex.* TR17 (2016). URL: [https:  
2491 //api.semanticscholar.org/CorpusID:3331839](https://api.semanticscholar.org/CorpusID:3331839).
- 2492 [7] László Lovász and Herbert E. Scarf. “The Generalized Basis Reduction Algorithm”.  
2493 In: *Mathematics of Operations Research* 17.3 (1992), pp. 751–764. ISSN: 0364765X,  
2494 15265471. URL: <http://www.jstor.org/stable/3689761> (visited on 04/17/2024).
- 2495 [8] Daniele Micciancio and Oded Regev. “Worst-Case to Average-Case Reductions  
2496 Based on Gaussian Measures”. In: vol. 37. Nov. 2004, pp. 372–381. ISBN: 0-7695-  
2497 2228-9. DOI: 10.1109/F0CS.2004.72.
- 2498 [9] Yanbin Pan and Feng Zhang. *A Note on the Density of the Multiple Subset Sum  
2499 Problems*. Cryptology ePrint Archive, Paper 2011/525. [https://eprint.iacr.  
2500 org/2011/525](https://eprint.iacr.org/2011/525). 2011. URL: <https://eprint.iacr.org/2011/525>.
- 2501 [10] The FPLLL development team. “fplll, a lattice reduction library, Version: 5.4.5”.  
2502 Available at <https://github.com/fplll/fplll>. 2023. URL: [https://github.  
com/fplll/fplll](https://github.<br/>2503 com/fplll/fplll).
- 2504 [11] Lars Tebelmann, Michael Pehl, and Vincent Immler. *Side-Channel Analysis of the  
2505 TERO PUF*. Cryptology ePrint Archive, Paper 2019/312. [https://eprint.iacr.  
2506 org/2019/312](https://eprint.iacr.org/2019/312). 2019. DOI: 10.1007/978-3-030-16350-1\_4. URL: [https:  
2507 //eprint.iacr.org/2019/312](https://eprint.iacr.org/2019/312).
- 2508 [12] Xiaoyun Wang, Guangwu Xu, and Yang Yu. “Lattice-Based Cryptography: A Sur-  
2509 vey”. In: *Chinese Annals of Mathematics, Series B* 44.6 (2023), pp. 945–960. DOI:  
2510 10.1007/s11401-023-0053-6. URL: [https://doi.org/10.1007/s11401-023-  
0053-6](https://doi.org/10.1007/s11401-023-<br/>2511 0053-6).