Shri Vile Parle Kelavani Mandal's

DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)

Digital identity system using post-quantum cryptographic paradigms

Submitted in partial fulfillment of the requirement
of the degree of

Bachelor of Technology in

Computer Science and Engineering (IoT and Cyber Security with Block Chain Technology)

By

Ansh Shah **60019210018**

Aditya Repe **60019210043**

Manas Patil **60019210046**

Soham Rane **60019210062**

Under the guidance of

Dr. Narendra Shekokar and Mrs. Dipali Bhole

University of Mumbai

A.Y. 2024 – 2025

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all the principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources, which have thus not been properly cited or from whom proper permission has not been taken, when needed.

Ansh Shah (60019210018)

Aditya Repe (60019210043)

Manas Patil (60019210046)

Soham Rane (60019210062)

Place: DJSCE
Date:

SVKM's Dwarkadas J. Sanghvi College of Engineering
Department of Computer Science and Engineering
(IoT and Cyber Security with Block Chain Technology)

Certificate

This is to certify that the project entitled "Digital identity system using post-quantum cryptographic paradigms" is a genuine work of "Ansh Shah" (60019210018), "Aditya Repe" (60019210043), "Manas Patil" (60019210046) and "Soham Rane" (60019210062) submitted in the partial fulfillment of the requirement for the award of the Bachelor of Technology in Computer Science and Engineering (IoT and Cyber Security with Block Chain Technology).

Dr. Narendra Shekokar
Project Guide

Mrs. Dipali Bhole
Project Guide

Dr. Narendra Shekokar
Vice Principal
and Head of the Department

Dr. Hari Vasudevan
Principal

Place: DJSCE
Date :

APPROVAL SHEET

Project entitled, ***"Digital identity system using post-quantum cryptographic paradigms"***, submitted by **"Ansh Shah" (60019210018), "Aditya Repe" (60019210043), "Manas Patil" (60019210046)** and **"Soham Rane" (60019210062)** is approved for the award of the Bachelor of Technology in Computer Science and Engineering (Internet of Things and Cyber Security with Blockchain Technology).

Internal Examiner
Name and Signature

External Examiner
Name and Signature

Dr. Narendra Shekokar
Project Guide

Dr. Narendra Shekokar
Head of the Department

Dr. Hari Vasudevan
Principal

Place: DJSCE
Date:

Acknowledgement

We take this opportunity to express our deep sense of gratitude to our project guides, Dr. Narendra Shekokar and Mrs. Dipali Bhole, for their continuous guidance and encouragement throughout the duration of our project work. It is because of their experience and knowledge that we were able to fulfill the requirement for the completion of this project within the stipulated time. We would also like to thank Dr. Narendra Shekokar, Vice Principal and Head of the Department, Computer Science and Engineering (IoT and Cyber Security with Block Chain Technology) for his encouragement, whole-hearted cooperation and support.

We would also like to thank our Principal, Dr. Hari Vasudevan and the management of D.J. Sanghvi College of Engineering, Vile Parle (W), Mumbai for providing us with all the facilities and a work-friendly environment. We acknowledge with thanks, the assistance provided by departmental staff, library, lab assistant lab attendants.

Ansh Shah (60019210018)
Aditya Repe (60019210043)
Manas Patil (60019210046)
Soham Rane (60019210062)

Abstract

In context of a rapidly developing digital landscape, it has become imperative that users secure their digital interactions. The advent of quantum computing presses this need further. This research presents a comprehensive framework for a digital identity system using post-quantum cryptographic paradigms to ensure long-term security and privacy. By integrating quantum-resistant algorithms for primitives such as key encapsulation mechanisms, digital signatures, and advanced hashing techniques, the proposed system delivers secure identity management. The framework is designed to withstand potential threats posed by quantum computers while maintaining compatibility with existing infrastructure. We evaluate the system's performance through a rigorous security analysis of the proposed algorithm by time leakage vectors and statistical analysis thereof, concluding that there are no implementation attack vectors independent to the proposed algorithm and that it's security is reliant on its components only.

Keywords: post-quantum cryptography, digital identity, security

List of Tables

List of Figures

List of Notations and Operations

Symbol	Meaning / Description
μ	Mean vector of a class (e.g., valid vs invalid ciphertexts)
Σ	Covariance matrix of a multivariate distribution
σ^2	Variance of a random variable
$\mathbb{E}[X]$	Expected value (mean) of a random variable X
$H(X)$	Shannon entropy of a distribution X
$D_M(\mathbf{x})$	Mahalanobis distance of $\mathbf{x}$ from the class mean
ct	Ciphertext
$Dec(sk, ct)$	Key decapsulation function
$Enc(pk, ss)$	Key encapsulation function
pk, sk	Public key and secret key respectively
R	Residue class ring $\mathbb{Z}_q[X]/(X^n + 1)$ used in Kyber
$Sig(sk, m)$	Digital signature of message m using secret key
ss	Shared secret (output of KEM)
T	Total execution time of a function call
t_i	Timing measurement of the i-th invocation
$Ver(pk, m, \sigma)$	Signature verification function
$\|$	Concatenation operator (e.g., for message encoding)
$\|\|x\|\|$	Norm of vector x, often used in noise estimation
$\oplus$	Bitwise XOR operation
$\mathbf{x}$	Observation or feature vector (e.g., timing, power usage, branch counts)

Abbreviations

Abbreviation	Full Form
API	Application Programming Interface
EUF-CMA	Existential Unforgeability under Adaptive Chosen Message Attack
FFI	Foreign Function Interface
FIPS	Federal Information Processing Standards
IND-CCA3	Indistinguishability under Adaptive Chosen Ciphertext Attack, 3rd level
IND-CPA	Indistinguishability under Chosen Plaintext Attack
KEM	Key Encapsulation Mechanism
LWE	Learning With Errors
MLWE	Module Learning With Errors
MSIS	Module Short Integer Solution
NIST	National Institute of Standards and Technology
PRF	Pseudo Random Function
RAM	Random Access Memory
SCA	Side-Channel Attack
XOF	eXtendable Output Function
ZKP	Zero-Knowledge Proofs
zk-SNARK	Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge

Contents

CHAPTER 1

Introduction

This chapter lays the foundation for understanding the transformative impact of quantum computing on digital security. It introduces the existential threat quantum algorithms pose to classical cryptographic techniques and highlights the urgent global response in the form of post-quantum cryptography (PQC). With the U.S. National Institute of Standards and Technology (NIST) spearheading standardization efforts, this section contextualizes the shift from traditional algorithms to quantum-resistant counterparts like CRYSTALS-Kyber and Dilithium. It sets the stage for exploring how this quantum shift could fundamentally reshape secure digital identity systems and data protection in the years ahead.

1.1 Motivation

As quantum computing technologies progress steadily toward practical implementation, conventional public key cryptographic systems such as RSA and ECC face an existential threat due to the efficiency of quantum algorithms such as Shor and Grover. These developments have prompted the cryptographic community to prioritize the development and standardization of post-quantum cryptographic (PQC) algorithms that are secure against quantum adversaries. One domain that will be critically affected is identity management, an area fundamental to authentication, authorization, and secure communication between digital systems.

Digital identity systems underpin services ranging from financial transactions to e-Governance. However, the security guarantees of many such systems are predicated on assumptions that fail in the presence of large-scale quantum computation. This thesis is motivated by the need to design identity frameworks that are robust in the post-quantum era, leveraging lattice-based primitives that offer strong security under well-studied hardness assumptions such as Module-LWE and Module-SIS.

1.2 Background

The National Institute of Standards and Technology (NIST) has recently concluded the third round of its Post-Quantum Cryptography Standardization process, recommending algorithms like CRYSTALS-Kyber for Key Encapsulation Mechanisms (KEM) and CRYSTALS-Dilithium for Digital Signatures [1]. These algorithms are founded on structured lattice problems and offer a compelling balance between performance and security. Given their prospective adoption in industry and government applications, these primitives form the cryptographic basis of our identity system.

CRYSTALS-Kyber is a module lattice-based KEM that achieves IND-CCA2 security using techniques like Fujisaki-Okamoto transformations and structured error sampling [2]. CRYSTALS-Dilithium, similarly, is a digital signature scheme based on the Fiat-Shamir with Aborts paradigm, exhibiting strong security in the Quantum Random Oracle Model (QROM) [3]. Together, these enable the construction of identity systems with confidentiality, authenticity, and post-quantum resilience.

1.3 Project Scope and Contributions

This thesis presents the design, implementation, and evaluation of a secure digital identity system rooted in post-quantum cryptographic foundations. The system architecture revolves around a novel, functional identity generation algorithm implemented in the C programming language. This implementation is augmented with a lightweight and secure interface, enabling usage in modern web environments via Foreign Function Interfaces (FFI) and JavaScript-based tooling.
Key contributions of this thesis include:

- **Design and Construction**: A post-quantum digital identity generation protocol based on Kyber768 and Dilithium2, implemented in C with cross-platform support.
- **Interface and Deployment**: A functional interface facilitating use in web ecosystems through Node.js and FFI bridges.
- **Threat Modeling**: Systematic threat modeling across classical and quantum threat vectors, including MITM, impersonation, and hash-forgery attacks.
- **Formal Security Analysis**: We perform a QROM security analysis of the key components, drawing from recent work on the tight security reductions of lattice-based primitives.
- **Timing and Statistical Analysis**: We empirically analyze the system for potential side-channel vulnerabilities, including timing variations and memory-access patterns. All critical functions are benchmarked and profiled for leakage resilience.
- **Safety and Performance Evaluation**: Our results affirm the system's safety under adversarial models and demonstrate competitive performance metrics consistent with reference implementations.

This thesis thus aims to offer a practical and secure foundation for post-quantum digital identities, contributing to the growing body of work on PQC deployment and secure identity design.

CHAPTER 2

Literature Survey

This chapter explores the evolving landscape of post-quantum cryptography (PQC) and its integration into digital identity systems. As traditional cryptographic methods such as RSA and ECC become vulnerable to quantum attacks, researchers have proposed quantum-resistant algorithms like CRYSTALS-Dilithium and novel schemes based on non-commutative algebra, aiming to balance security with efficiency. In parallel, advances in digital identity systems, such as those in decentralized and user-controlled models, are gaining traction. These systems, when combined with zero-knowledge proofs and post-quantum certificates, offer a promising path towards secure, privacy-preserving identity verification in a quantum era. Finally, recent bibliometric studies outline ongoing research trends, such as cryptographic performance optimization, while also highlighting critical gaps in practical deployment, especially within real-world digital identity infrastructures. These insights collectively underscore the need to adapt identity systems for countering future quantum threats.

2.1 History of Post-Quantum Cryptography

Post-quantum cryptography (PQC) describes cryptographic protocols that would be secure in case the adversary owns a quantum computer. RSA and ECC are standard public-key cryptosystems that are known to be vulnerable to quantum attacks, most importantly Shor's algorithm which can quickly compute integer factorization and discrete logarithms [4]. Researchers, therefore, have tested various quantum-resistant techniques.

Lattice-based cryptography has emerged as a promising contender due to its secure foundation and efficiency of operation. The Learning with Errors (LWE) problem, originally suggested by Regev [5], and its structured forms, Ring-LWE and Module-LWE, are the basis of most lattice-based schemes. CRYSTALS-Dilithium, a lattice-based digital signature scheme, is an example of a scheme and has been selected by NIST for standardization [6]. Some of the prominent post-quantum cryptography (PQC) schemes are hash-based signatures like SPHINCS+ [7], code-based constructs such as Classic McEliece [8], multivariate polynomial signatures such as Rainbow [9], and isogeny-based schemes represented by SIKE [10].

2.2 NIST PQC Standardization Process

Sensing the imminent danger of quantum computing, the National Institute of Standards and Technology (NIST) began a multi-round standardization of PQC in 2016 [11]. Following rigorous assessment, NIST shortlisted four algorithms to be standardized in July 2022: CRYSTALS-Kyber for key encapsulation mechanisms (KEMs), and CRYSTALS-Dilithium, FALCON, and SPHINCS+ for digital signatures [6]. These were chosen on the basis of security strength, performance, and implementation considerations.

The standardization continues, with the fourth round at NIST taking into account more signature schemes and specific applications. Of particular interest, HQC, a code-based KEM, was chosen in this round, demonstrating the diversity of techniques under investigation [12].

2.3 Transition towards Post-Quantum Cryptography

It is difficult to migrate legacy systems to PQC in a number of areas. One of them is the incorporation of new cryptographic primitives within the established protocols, e.g., TLS, X.509 certificates, and secure message systems. Post-quantum algorithms generally imply larger key sizes and signature sizes, which are a burden in resource-constrained environments such as IoT devices [13].

For a seamless migration, "cryptographic agility" has been put forth as a concept that allows systems to change between cryptographic algorithms without major overhauls [14]. Hybrid cryptographic schemes that incorporate classical and post-quantum algorithms have been suggested to offer interoperability during the transition period [15]. Google and Cloudflare have tested hybrid deployments to defend against pre-emptive quantum attacks while offering compatibility [16, 17].

Standards organizations such as ETSI and IETF are actively making changes to the cryptographic standards and protocols to facilitate PQC, with a global coordinated effort on this transition [18].

2.4 Digital Identity Systems

Digital identity systems are defined largely for authentication and access control purposes in a wide range of applications, and centralized models will most likely be based on centralized authorities. These centralized models face privacy issues as well as being single points of failure [19]. In response, decentralized and user-centric models, such as Self-Sovereign Identity (SSI), have gained traction. SSI empowers users with greater control over their identity attributes and verifiable credentials [20].

Blockchain platforms like Sovrin and Hyperledger Indy have explored SSI infrastructures, leveraging distributed ledgers to anchor decentralized identifiers (DIDs) and support zero-knowledge proof (ZKP) based verifications [21]. However, integrating PQC into these systems is imperative to withstand future quantum threats. Projects like PQID and efforts from European eIDAS frameworks illustrate early adoption of post-quantum primitives in digital identity stacks [22].

2.5 Research Gaps

Despite advancements in PQC and digital identity systems, several research gaps persist. Many proposed PQC schemes lack extensive testing in real-world identity systems, particularly in mobile and resource-limited settings [23]. While hybrid protocols exist, rigorous security models for their composition are underdeveloped [24]. Mechanisms for efficient revocation, attribute-based access control, and long-term storage security under PQC remain areas requiring further research [25].

Usability and privacy concerns also arise with post-quantum digital identity systems. While ZKPs offer privacy-preserving proofs, their post-quantum counterparts often introduce significant computational overhead, limiting scalability [26]. Furthermore, bibliometric analyses indicate a concentration of research on lattice-based systems, with an under-representation of alternative paradigms such as isogenies and multivariate cryptography [27].

The research gap can be encapsulated as:

1. Lack of practical implementations: Most PQC algorithms have only recently been implemented in a practical way and deployed in production according to the NIST standards. Many proposed algorithms are resource-intensive and have not yet been optimized for large-scale deployment in real-world systems. In particular, digital identity frameworks such as Self-Sovereign Identity (SSI) struggle to incorporate PQC due to issues surrounding efficiency and integration with existing infrastructures [28], [29]. This is because most current research focuses on improving these cryptographic primitives and lacks robust testing in diverse environments.

2. Scalability issues: Any identity system conceived with post-quantum paradigms in mind has to consider the amount and efficiency of computational resources it will consume at scale. The PQC algorithms' relatively larger key sizes and signature lengths hinder performance when deployed in cloud-based or decentralized identity systems, which must handle millions of transactions per second [29].

3. Transition frameworks: Effective frameworks that can facilitate the seamless transition of endpoint security from classical algorithms to post-quantum algorithms are scarce and in early stages of development. Any such system must be able to

dynamically switch between algorithms depending on threats or hardware [28].

4. Usability: A significant gap also lies in the user experience and usability of PQC-based digital identity systems. Complex cryptographic mechanisms may make these systems less intuitive for end-users, especially in self-managed identity models like SSI. Post-quantum digital identity systems must be user-friendly while maintaining security. This remains a key challenge that has not been sufficiently addressed [29].

A concrete conclusion of the above literature survey can be that,
"A practical post-quantum digital identity solution with robust testing and security analysis does not reasonably exist, or is not performant, efficient, or usable enough to be deployed on a user endpoint."
This thesis aims to address these gaps by evaluating the feasibility of integrating CRYSTALS-Dilithium into SSI frameworks and proposing a modular architecture that supports hybrid and quantum-safe identity verification workflows.

CHAPTER 3

Proposed Model

This chapter proposes a modular and security-focused model for a post-quantum cryptography-enabled digital identity system. The design distinctly separates the application logic from cryptographic operations, ensuring flexibility, maintainability, and rigorous security assurance. At its core, the model introduces a dedicated cryptographic kernel, implemented in C for performance and type safety. This kernel encapsulates key cryptographic primitives, such as hashing, digital signatures, and key encapsulation mechanisms, into a standalone, testable module compliant with modern security standards. To securely and efficiently manage application data, a state module is proposed. It acts as an intermediary between the application and the cryptographic core. In addition, the network and integration modules expand the capabilities of the system. These include secure communication layers, SSO interfaces, credential recovery mechanisms, and even multimedia signature support.

We propose a holistic design for the solution as concerned. The components of the solution separate the concerns of an end-user application not only from the user's point of view but the authors too.

3.1 Cryptographic Kernel

A system focused on the security of end users must separate the concerns of data across modules effectively. We intend to implement a pure cryptographic kernel containing all requisite functions in one place, importing only the essential libraries and exporting static data to act as state in further modules. The intended testing against performance benchmarks, IND-CCA 3 standard, and other standards will be performed on this kernel.

Reason

An integrated application module implementing all necessary cryptographic functions would be well-suited from the point of view of application performance and locality of behavior. However, such a module would be extremely difficult to isolate from the rest of the application for the rigorous security analysis required. Moreover, performance and metrics logs for implementation attacks would be easier to obtain for such a kernel. This design also means that the application will act as a wrapper around this kernel, allowing it to be manipulated without affecting the cryptographic functions.

Language

C. Low-level languages will enable strong performance and hardware-level optimizations, as well as strong type safety.

Components

Implementations of the following interfaces from scratch:

1. **Hashing interface** – hashing function
2. **KEM/DPKE interface** – keypair generation, key generation, encrypt, decrypt, encapsulate, decapsulate
3. **Digital Signature interface** – sign, verify

3.2 State Module

To better integrate this kernel with any application, a dedicated state machine can be implemented that will handle the state of data at any point in time, providing safe interfaces to other modules. This module is not pure, as state will be modified here. However, to hedge against the problems of object-oriented architecture, we will define and enforce custom types of required structures in functions and handle side effects on the client side.

Reason

Upon any CRUD operation, the application state has two disjoint sides for a short amount of time – the one that is rendered and shown to the user (interface value) and the actual state of the value updated after the round trip to the database. This small time is not insignificant and presents a crucial roadblock to good user experience. Additional processing on that data might be needed before and/or after the data is sent to and received from the database. A dedicated state module will take care of this processing and side-effects as a performant module reserved for this purpose. Strong type safety and testing will isolate the security issues of application state from the database. This is important because we are not performing a simplistic hash on the password that can be done directly at the site of the database methods themselves. The cryptographic kernel will handle the necessary cryptographic processes.

Components

- **User type** – contains attributes such as email address
- **Identity type** – separates archival of public keys for verifying digital signatures after keypair expiration, stored in the Identity type

3.3 Other Modules Considering Application

1. **Network module** –The network module will be responsible for handling secure communication between the application, the cryptographic kernel, and external systems. This involves request filtering, firewall integration, and managing network protocols to ensure the confidentiality and integrity of data in transit. The module

will act as the first line of defense, filtering out malicious requests and performing firewall-level checks before any sensitive data reaches the cryptographic kernel.

2. **Web extension** – The network module will be responsible for handling secure communication between the application, the cryptographic kernel, and external systems. This involves request filtering, firewall integration, and managing network protocols to ensure the confidentiality and integrity of data in transit. The module will act as the first line of defense, filtering out malicious requests and performing firewall-level checks before any sensitive data reaches the cryptographic kernel.
   Components:

   (a) **SSO Interface** – integrates with third-party SSO providers, enabling seamless authentication across multiple services while maintaining the security of the user's identity.

   (b) **Password Reset and Credential Generation Interface** – securely manages password reset workflows, and generates new credentials using strong cryptographic primitives, ensuring that users can easily recover access while protecting their data.

   (c) **Multimedia Signature Interface** – signs multimedia content (images, videos, documents) to ensure the authenticity and integrity of digital assets. This component interacts with the cryptographic kernel for signing operations, ensuring that all signatures are verifiable.

CHAPTER 4

Methodology

This chapter outlines a rigorous methodology for designing, implementing, and analyzing a cryptographically secure system with resilience against both classical and quantum adversaries. The approach emphasizes modularity, verifiability, and side-channel resistance through a layered architecture consisting of a hardened cryptographic kernel, state management modules, and carefully abstracted network interfaces. Security considerations are woven into each phase, from algorithmic selection (CRYSTALS-Kyber and Dilithium) to low-level implementation choices such as static memory allocation and deterministic control flow. The methodology further incorporates advanced statistical techniques, including Welch's t-test and Mahalanobis distance analysis, to empirically validate timing invariance and detect multidimensional leakage vectors. By systematically isolating concerns and subjecting each component to formal or statistical scrutiny, this framework aims to achieve IND-CCA3 security while maintaining practical performance. The following sections detail the architectural decisions, cryptographic foundations, and evaluation mechanisms that collectively address both functional correctness and implementation security.

4.1 Approach

The systematic approach adopted in this research integrates design, development, and analysis phases in a modular fashion to ensure separation of concerns, cryptographic security, and adaptability. The system is implemented in a layered architecture composed of a low-level cryptographic kernel, a state management module, and network interfaces. Each layer is independently verifiable and designed to be subjected to formal or statistical security analysis. The cryptographic kernel is developed in C for performance profiling and in-depth instrumentation, which aids in both functional correctness and side-channel leakage assessment.

The application stack wraps around the kernel through Foreign Function Interfaces (FFI), ensuring minimal exposure of sensitive logic to potentially vulnerable layers. Testing, logging, and benchmarking are systematically integrated using custom harnesses to isolate cryptographic operations, particularly during key encapsulation, decapsulation, and digital signing/verification steps.

4.2 Design Considerations

The architectural decisions are rooted in modularity, performance, and isolation principles. The cryptographic kernel is developed in pure C without dynamic memory allocation to reduce complexity and exposure to memory-based vulnerabilities. All cryptographic operations are statically linked and confined to this kernel. Statelessness is maintained as far as possible to allow deterministic behavior and easier reproducibility.

To mitigate timing and side-channel vulnerabilities, the kernel includes deterministic branching and cache-aware data access patterns. Further, to facilitate analysis, each operation is wrapped with time measurement hooks and benchmark counters during the testing phase. An external state module written in a high-level language interfaces with this kernel and manages data transformations and concurrency effects introduced by the runtime system or user interface.

Moreover, data serialization and network communication routines are defined outside the cryptographic scope, with pre- and post-processing functions invoked in a defined lifecycle sequence. All inputs and outputs of the kernel are strictly typed and bounded to

prevent buffer overflows and malformed data attacks.

4.3 Cryptographic Considerations

The security model assumes IND-CCA3 resilience as the target property, particularly for KEM operations. Cryptographic primitives are selected based on post-quantum cryptographic standards, specifically CRYSTALS-Kyber for KEM and CRYSTALS-Dilithium for digital signatures. Hashing functions are derived from SHAKE256 (XOF) and implemented using precomputed round constants for efficiency and reduced memory load.

To validate these primitives in practice, the following processes are enforced:

- All key generation routines are validated against FIPS 203 vectors.
- Signature verification includes expiry and archival checks for identity rotation.
- Decapsulation failure is handled securely to avoid leakage from control flow patterns.

Security goals are twofold: (i) resistance against quantum-enabled adversaries using cryptographically hard problems, and (ii) implementation security against physical or side-channel adversaries.

4.3.1 Timing Attack Evaluation

A dedicated timing analysis harness is developed to empirically measure the execution time of sensitive operations across thousands of controlled invocations. Each primitive (e.g., `kem_decapsulate`) is executed with both valid and malformed ciphertexts in a randomized schedule to avoid temporal patterns.

The time is measured using the `clock_gettime(CLOCK_MONOTONIC_RAW, ...)` interface for nanosecond granularity and precision. Timing traces are collected under controlled conditions to limit noise from scheduler interference and hardware prefetchers.

The obtained timing dataset is then analyzed statistically using a Welch's t-test and Cohen's d value to identify any significant deviation in execution profiles. If distinguishable timing behavior is observed (p ¡ 0.05, d ¿ 0.2), mitigations such as branch flattening, constant-time lookup tables, or masking are applied.

4.3.2 Statistical Space Evaluation using Mahalanobis Distance

To detect high-dimensional statistical leakage beyond first-order timing discrepancies, we analyze the operational state space via Mahalanobis distance. This statistical technique allows comparison between execution profiles by taking into account the covariance of the observed features.

Let $\mathbf{x}$ be a timing or resource usage vector of an operation (e.g., cycles taken, branch count, cache hits), and μ the mean vector of the class (valid or invalid ciphertext), and Σ its covariance matrix. Then:

$$D_M(\mathbf{x}) = \sqrt{(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)}$$

This distance metric accounts for inter-feature dependencies, which are crucial in detecting subtle leakages that univariate analysis (e.g., t-tests) might miss.

A statistical profile is trained on known secure operations. Subsequent samples are then measured and compared. A threshold T is empirically determined such that:

$$D_M(\mathbf{x}) > T \Rightarrow \text{Potential leakage / distinguishability}$$

This methodology enables detection of micro-architectural or implementation-related leakage vectors that might otherwise evade conventional testing.

CHAPTER 5

Requirements

This chapter lists system software and hardware requirements. To ensure successful deployment and execution of the digital identity system software, certain system requirements must be met. While these are not legally binding, they serve as a recommended baseline configuration to achieve reliable, secure, and reproducible results. The software components listed below are widely available and scalable, reducing the friction in both development and deployment stages.

While not binding in any real sense, the following are some requirements for the digital identity system software to run on. These requirements are easily available and scalable, making the results reproducible. The only exception these advantages is the node.js binary's version requirement - it is restricted to 18.9.0 for at least installing the ffi-napi and ref-napi libraries that act as Foreign Function Interfaces (FFIs) for the C code to run on. After installing these libraries however, one can switch back to the most recent version to benefit from latest recent security patches and software updates. We do recommend using node version manager (nvm) for the same, although that is certainly not required.

5.1 Software Requirements

- **Operating System:** Linux (UNIX-based), preferably **Debian 12 Bookworm** for consistent package support.
- **Compiler Tools:** `gcc`, `make` – essential for compiling C code and handling build processes.
- **Node.js Binaries:** Version **18.9.0** (mandatory during ffi-napi installation).
- **Node Libraries:** `ffi-napi`, `ref-napi`.
- **Web Browser:** A modern browser based on the **Chromium V8 engine**, such as Google Chrome or Microsoft Edge.

5.2 Hardware Requirements

- **Machine Type:** Intel Haswell-based Virtual Machine (VM) hosted on **Google Cloud Platform (GCP)**.
- **Memory:** Minimum **4 GB RAM** to handle development and runtime workloads.
- **CPU:** Shared virtual CPU (suitable for development and lightweight production use).
- **Storage:** Minimum **10 GB boot disk** for operating system, dependencies, and project files.

CHAPTER 6

Architecture

This chapter describes the system architecture, which is designed as a composition of formally verifiable cryptographic modules and stateful interfaces, each enforcing strict security boundaries while maintaining interoperability. At its core lies a Cryptographic Kernel implementing lattice-based primitives (CRYSTALS-Kyber for KEM and Dilithium for signatures) with hardness rooted in Module-LWE and Module-SIS problems. This kernel operates in an isolated execution environment with constant-time guarantees, around which higher-level components are organized as stateless wrappers. The State Module mediates between this kernel and user-facing interfaces, including both web clients and browser extensions, while enforcing zero-knowledge proof attestations via Groth16 circuits. Each architectural layer adheres to the principle of least privilege: the kernel handles only raw cryptographic operations, the state module manages session persistence and ZKP verification, and UI components render isolated views without direct crypto access. This decomposition enables independent security analysis of the mathematical foundations (MLWE/MSIS problems), protocol implementations (Kyber/Dilithium), and application logic (web extensions with ZKP-backed assertions) while maintaining end-to-end cryptographic integrity.

The architectural framework of the proposed system is characterized as modular, highly scalable, and focused on protecting security from all threats posed by quantum attacks-but in such a manner that aims not to compromise user privacy. The ease with which scalability and its modifications can be made throughout all components also supports the careful implementation of the necessary security protocols at a granular level. The software architecture consists of

1. the cryptographic kernel

2. the state module

3. the application itself

More generally, the fundamental context of this architecture is considered the vast Chromium-based ecosystem of browsers and their applications. The architectural design employs zero-knowledge proofs (ZKP) through a customized layer built for browser contexts. This will provide the secure authentication of users and yet offer confidentiality of the sensitive information. The user-facing element comprises a browser-centric interface. It works efficiently with the Chromium V8 engine. It supports functionalities, such as SSO, identity verification, and credential management. Additionally, this interface is supported by content security policies and HTTPS protocols. At the backend, a Node.js-built server serves as the middle layer connecting the front end to the crypto kernel core by use of packages like ffi-napi and ref-napi that support interaction with the low-level primitives from the crypto. In addition to this, a minimum of a database is added to have metadata needed for its operation without storing the sensitively private user data that usually comes with high privacy requirements and hence imposes heavy restriction. Architectural framework also incorporates a full package of testing as well as logging protocols, mainly used to validate cryptography operations, measure performance metrics, and identify any anomalies. In summary, the architecture provides a secure and efficient yet accessible digital identity system with high immunity against some challenges issued by quantum computing.

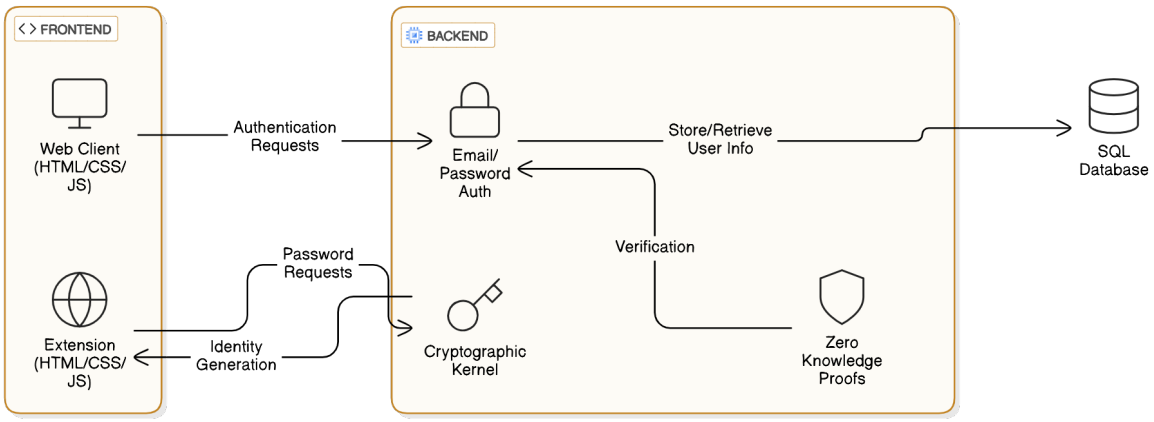

Figure 6.1: Software Architecture of the Digital Identity System

6.1 Cryptographic Kernel

An end-user-sensitive security system would, therefore, be likely to take up a modular design, allowing its components to remain in accordance with the principles of concern separation. In this regard, the architecture proposed takes the form of having a well-defined cryptographic kernel that will contain all the needed cryptographic functions within a single, well-isolated module. It has only a limited set of libraries and holds its static data as the state for other modules, thus carrying out sharp lines of demarcation between the cryptography process and application logic. This modularization simplifies intense security analyses, such as testing against NIST KATs; performance benchmarks, and standards, such as IND-CCA3. Independently fielding crypto-processes would allow for honest-to-goodness validity checks for correctness and robustness without interference from higher layers of application. In addition, measurements and logging about performance and implementation attacks might be made unobtrusively, thus on-the-fly optimization and fortification might be done. Though combining all functionality into a single application may result in benefits related either to performance or to behavioural locality, there are significant costs entailed in identifying suitable techniques for effectively isolating these components in order to thoroughly discuss any security implementation. A cryptographic kernel facilitates this process by functioning as an independent module where the application offers only a superficial layer of protection. This split clearly does permit the appli-

cation to adapt and become modified without jeopardizing cryptographic integrity. The kernel is written in C, chosen for the advantageous performance as well as hardware-level optimization opportunities, along with strong type safety. The core parts of the kernel include interfaces shaped to hashing functions, key encapsulation mechanisms (KEMs), deterministic public-key encryption (DPKE), and digital signature procedures. The interfaces enable key-pair generation, encryption, decryption, signing, and verification of critical functions conducted bottom-up in conformance with post-quantum cryptographic standards and to provide a high-level view of sensitive security-related operations.

6.2 Cryptosystems employed

6.2.1 CRYSTALS-Kyber

Description

CRYSTALS-Kyber is a lattice-based Key Encapsulation Mechanism (KEM) that has been chosen for standardization by NIST as part of the post-quantum cryptography suite. It is based on the hardness of the Module Learning with Errors (MLWE) problem, which is a generalization of the well-known Learning with Errors (LWE) problem. Kyber offers efficient key generation, encapsulation, and decapsulation algorithms and is designed for high performance and small communication overhead, making it suitable for practical use in secure communications.

Algorithms

Algorithm 1 Key Generation

1: **Input:** None
2: **Output:** Public key `pk`, Secret key `sk`
3: Generate random seed seed and derive $\mathbf{A}$ using a hash function
4: Sample secret vector $\mathbf{s}$ and error vector $\mathbf{e}$
5: Compute $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$
6: Set $\texttt{pk} = (\mathbf{A}, \mathbf{b})$
7: Set $\texttt{sk} = \mathbf{s}$
8: **return** `pk`, `sk`

Algorithm 2 Encapsulation

1: **Input:** Public key $(\mathbf{A}, \mathbf{b})$, Message m
2: **Output:** Ciphertext ct, Shared secret ss
3: Sample random vector $\mathbf{r}$ and error vectors $\mathbf{e}_1, \mathbf{e}_2$
4: Compute $\mathbf{u} = \mathbf{A} \cdot \mathbf{r} + \mathbf{e}_1$
5: Compute $\mathbf{v} = \mathbf{b} \cdot \mathbf{r} + \mathbf{e}_2$
6: Encrypt message m to produce ciphertext ct $= (\mathbf{u}, \mathbf{v})$
7: Derive shared secret ss from m using a hash function
8: **return** ct, ss

Algorithm 3 Decapsulation

1: **Input:** Ciphertext ct $= (\mathbf{u}, \mathbf{v})$, Secret key $\mathbf{s}$
2: **Output:** Shared secret ss
3: Compute $\mathbf{v}' = \mathbf{u} \cdot \mathbf{s}$
4: Recover message m from $\mathbf{v}'$ and $\mathbf{v}$
5: Derive shared secret ss from m using a hash function
6: **return** ss

6.2.2 CRYSTALS-Dilithium

Description

CRYSTALS-Dilithium is a post-quantum digital signature scheme based on the MLWE and Module Short Integer Solution (MSIS) problems. It provides secure and efficient digital signatures for applications that require authentication and data integrity.

Algorithms

Algorithm 4 Key Generation

1: **Input:** None
2: **Output:** Public key pk, Secret key sk
3: Generate matrix $\mathbf{A}$ using a random seed
4: Sample secret vectors $\mathbf{s}_1, \mathbf{s}_2$ from a discrete Gaussian distribution
5: Compute $\mathbf{t} = \mathbf{A} \cdot \mathbf{s}_1 + \mathbf{s}_2$
6: Set pk $= \mathbf{t}$ and sk $= (\mathbf{s}_1, \mathbf{s}_2)$
7: **return** pk, sk

Algorithm 5 Signing

1: **Input:** Message m, Secret key $(\mathbf{s}_1, \mathbf{s}_2)$, Matrix $\mathbf{A}$
2: **Output:** Signature $\sigma = (\mathbf{z}, c, \text{hint})$
3: Sample random vector $\mathbf{y}$ from a discrete Gaussian distribution
4: Compute $\mathbf{w} = \mathbf{A} \cdot \mathbf{y}$
5: Use hash function H to derive challenge c from m and $\mathbf{w}$
6: Compute response $\mathbf{z} = \mathbf{y} + c \cdot \mathbf{s}_1$
7: **if** $\mathbf{z}$ is not short **then**
8: Restart the process
9: **end if**
10: Compute side information hint
11: **return** $\sigma = (\mathbf{z}, c, \text{hint})$

Algorithm 6 Verification

1: **Input:** Message m, Signature $\sigma = (\mathbf{z}, c, \text{hint})$, Public key $\mathbf{t}$, Matrix $\mathbf{A}$
2: **Output:** Valid / Invalid
3: Compute $\mathbf{w}' = \mathbf{A} \cdot \mathbf{z} - c \cdot \mathbf{t}$
4: Use hint to adjust $\mathbf{w}'$ and compare with $\mathbf{w}$
5: **if** $\mathbf{z}$ is within bounds and checks pass **then**
6: **return** Valid
7: **else**
8: **return** Invalid
9: **end if**

6.3 Mathematical Basis for CRYSTALS Algorithms

6.3.1 Module Learning With Errors (MLWE) Problem

The Module Learning With Errors (MLWE) problem is a generalization of the Learning With Errors (LWE) problem, extended to module lattices. It is defined as follows:

Problem Definition

Let $R = \mathbb{Z}[x]/\langle f(x) \rangle$ be a ring where $f(x)$ is a monic polynomial of degree n. Let q be a positive integer, and let χ be an error distribution over $R_q = R/qR$. The MLWE problem is parameterized by:

- Dimension $k \in \mathbb{N}$
- Modulus q
- Error distribution χ

Instance Generation

Given a uniformly random matrix $\mathbf{A} \in R_q^{k \times k}$ and a secret vector $\mathbf{s} \in R_q^k$, the MLWE instance is generated by:

$$\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \mod q,$$

where $\mathbf{e} \in R_q^k$ is sampled from the error distribution χ.

Decision Version

The decision version of the MLWE problem asks to distinguish whether a given pair $(\mathbf{A}, \mathbf{b})$ is:

- A valid MLWE instance, i.e., $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ for some $\mathbf{s} \in R_q^k$ and $\mathbf{e} \in R_q^k$ sampled from χ, or
- A uniformly random pair $(\mathbf{A}, \mathbf{b})$.

6.3.2 Module Short Integer Solution (MSIS) Problem

The Module Short Integer Solution (MSIS) problem is a generalization of the Short Integer Solution (SIS) problem, extended to module lattices. It is defined as follows:

Problem Definition

Let $R = \mathbb{Z}[x]/\langle f(x) \rangle$ be a ring where $f(x)$ is a monic polynomial of degree n. Let q be a positive integer. The MSIS problem is parameterized by:

- Dimension $k \in \mathbb{N}$
- Modulus q
- Norm bound β

Instance Generation

Given a uniformly random matrix $\mathbf{A} \in R_q^{k \times k}$, the MSIS problem is defined as finding a non-zero vector $\mathbf{x} \in R^k$ such that:

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{0} \mod q \quad \text{and} \quad \|\mathbf{x}\| \leq \beta,$$

where $\| \cdot \|$ is a suitable norm (e.g., Euclidean norm or infinity norm).

Hardness Assumptions

The hardness of both MLWE and MSIS is based on worst-case to average-case reductions for problems on ideal and module lattices. These problems are believed to be hard even for quantum computers.

6.4 State Module

A state module is recommended especially to oversee the data state management at any given point in time, thus ensuring seamless integration between the cryptographic kernel and the larger application. This module functions as a conduit, offering secure and uniform interfaces that enable other application components to engage with the system's state. In contrast to a purely cryptographic kernel, the state module alters and preserves data, thereby rendering it inherently mutable. Nonetheless, to address challenges typically associated with object-oriented architectures, custom types and stringent structures will be delineated and enforced for all necessary functions. It ensures that side effects are handled on the client side, thus isolating cryptographic operations, and protecting them from side or accidental interference. The state module addresses one of the fundamental challenges of modern applications: that the interface-rendered state presented to users' needs to differ from the real state updated behind the scenes after database round trips. This minor deviation, although brief, can have a significant impact on user experience and system reliability. Additional processing of secondary data may also be required either before sending to or after receiving from the database. Centralizing these operations further ensures that the state module acts as a good intermediary, eliminating latency and efficiently dealing with side effects. It also increases security by segregating issues related to application-state from that of the database. In this context, because of the particular cryptographic processes used, it is impossible to assume simple hashing from a database standpoint.

6.5 User Interface Components

Both components emphasize usability, security, and seamless integration with the backend services. Usage of modern web technologies such as Tailwind CSS and JavaScript aim to provide a responsive and dynamic user experience.

6.5.1 Web Client

The Web Client is designed for user registration and secure identity management through a modern browser-based application.

Landing Page

- **Header**:
  - **Title**: *Digital Identity Registration*
  - **Description**: "Secure Registration"
- **Registration Form**:
  - **Fields**:
    * Email Address (with validation for proper format)
    * SAP ID (11-digit numeric input validation)
    * Age (minimum: 18)
  - **Submit Button**:
    * Triggers Zero-Knowledge Proof (ZKP) generation, verification, and user registration.
  - **Verification Status**:
    * Displays a progress indicator during the registration process.

Client-Side Features

- **Zero-Knowledge Proof Generation**:
  - Uses `snarkjs` to generate and verify cryptographic proofs.
  - Ensures input confidentiality while verifying claims.
- **Registration Process**:
  - Communicates with backend endpoints to securely generate and register a digital identity.
- **Error Handling**:
  - Alerts for invalid inputs or process failures.
- **Session Management**:
  - Sets cookies for session persistence.

6.5.2 Web Extension

The Web Extension provides seamless integration with the user's browser for enhanced identity management functionalities.

Landing View

- **Welcome Screen**:
  - **Title**: *Welcome to Digital Identity*
  - **Description**: "Your secure identity management solution"
  - "Get Started" Button: Redirects users to the registration page.

Home View

- **User Information**:
  - Displays user email.
- **Website Interaction**:
  - Shows the active website domain.
  - "Generate Password" Button:
    - Triggers the password generation process based on website-specific requirements.
  - Displays generated password in a secure container.
- **Logout Button**:
  - Clears session cookies and redirects to the landing view.

Features

- **Password Generation**:
  - Interacts with VM endpoints to generate secure, site-specific passwords based on detected requirements.
  - Automatically injects passwords into login forms where possible.

- **Password Requirements Detection**:
  - Injects scripts into the current tab to analyze form requirements.
  - Extracts and adheres to constraints like minimum length, special characters, or uppercase letters.
- **Session Management**:
  - Handles user authentication state through cookies.
  - Supports logout functionality to clear session data.

6.6 Zero-Knowledge Proof Module

The module is designed to verify sensitive attributes, specifically age and SAP ID, without disclosing the raw data or storing it. The architecture leverages Zero-Knowledge Proofs (ZKPs) based on Groth16, a succinct, non-interactive proof system compatible with zk-SNARKs.

6.6.1 Theoretical Foundations

Zero-Knowledge Proofs (ZKPs)

ZKPs enable one party (the prover) to convince another party (the verifier) that a statement is true without revealing any underlying information. This is particularly useful for verifying sensitive data such as age or ID attributes in a privacy-preserving manner.

Groth16 Protocol

The Groth16 protocol is used for proving computational integrity efficiently. It involves three primary phases:

1. **Setup:** Establishes cryptographic parameters for a specific computation circuit.
2. **Proving:** Generates a proof that a given computation is valid based on the circuit and inputs.
3. **Verification:** Confirms the proof's validity using publicly known parameters.

Circuit Design

The circuit defines the logic of the proof, encoded mathematically. For this module:

- Age verification ensures the provided age is ≥ 18.
- SAP ID verification checks the first digit of the SAP ID is ≥ 6.
- Boolean constraints enforce that the outputs (`isAgeValid` and `isSAPValid`) are binary.

The circuit uses the following template:

$$\text{GreaterEqThan:} \quad \text{out} \leftarrow \begin{cases} 1 & \text{if } in_0 \geq in_1 \\ 0 & \text{otherwise} \end{cases}$$

6.6.2 Security Considerations

- The cryptographic setup ensures that no information about the inputs (age and SAP ID) is leaked.
- The Groth16 protocol is resistant to quantum attacks under current assumptions.

CHAPTER 7

Implementation

This chapter describes the concrete implementation of cryptographic primitives and zero-knowledge proof systems, bridging theoretical foundations with engineering realities. The CRYSTALS-Kyber and Dilithium implementations are rigorously derived from their mathematical bases—Module-LWE for Kyber's KEM and Module-SIS for Dilithium's signatures—with careful attention to constant-time execution and side-channel resistance. Each primitive's reference implementation is analyzed for both functional correctness (via NIST test vectors) and computational efficiency (through cycle-accurate profiling). The zero-knowledge proof system implements Groth16 circuits over BN254 curves, optimizing for succinct verification while maintaining non-interactive soundness. Code examples demonstrate critical operations: Kyber's CPA-secure key encapsulation, Dilithium's deterministic key derivation, and ZKP circuit constraints for identity assertions. Together, these components form a vertically integrated stack where algorithmic strength (lattice assumptions), implementation hygiene (memory safety, timing invariance), and protocol security (ZK proof composition) are enforced at every layer.

7.1 CRYSTALS-Kyber

7.1.1 Overview

CRYSTALS-Kyber is a lattice-based key encapsulation mechanism (KEM) designed as part of the CRYSTALS (Cryptographic Suite for Algebraic Lattices) project. This project aims to develop secure, efficient, and quantum-resistant cryptographic schemes. Kyber, specifically, addresses the need for post-quantum secure key exchange.

The development of Kyber was motivated by the increasing threat posed by quantum computers, which can efficiently break classical cryptographic schemes relying on the hardness of problems like integer factorization and discrete logarithms. Kyber leverages the Module Learning With Errors (MLWE) problem, a lattice-based problem believed to be hard even for quantum computers.

Kyber gained significant attention during the NIST Post-Quantum Cryptography (PQC) Standardization Process, where it emerged as one of the finalists and was eventually selected for standardization in 2022. Its combination of security, efficiency, and practicality makes it suitable for a wide range of applications, including secure communication and data encryption.

7.1.2 Mathematical Foundations

Kyber operates over the ring $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$, where n is a power of 2. The main steps in the key exchange process are:

1. **Key Generation:** Generate a secret matrix $\mathbf{s}$ and a uniformly random matrix $\mathbf{A}$. Compute the public key $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$, where $\mathbf{e}$ is sampled from a small error distribution.

2. **Encapsulation:** Compute a ciphertext $\mathbf{c}$ that encrypts a randomly chosen message $\mathbf{m}$ using the public key.

3. **Decapsulation:** Recover the shared secret $\mathbf{m}$ from the ciphertext $\mathbf{c}$ using the secret key $\mathbf{s}$.

7.1.3 Reference Implementation Details

The reference implementation of Kyber uses:

- Polynomial arithmetic over R_q, optimized with Number Theoretic Transform (NTT) for efficient polynomial multiplication.
- Byte-packing techniques to ensure compact key and ciphertext representations.
- Randomized sampling functions to securely generate secrets and errors.

7.1.4 Strengths

- Efficient key sizes and fast operations, making it suitable for constrained environments.
- Strong theoretical guarantees based on reductions to MLWE.
- Resistant to side-channel attacks through carefully implemented constant-time operations.

7.1.5 Code Example

Below is a simplified snippet illustrating polynomial multiplication in Kyber:

```
// Polynomial multiplication using NTT
void poly_ntt(int16_t *r, const int16_t *a) {
    // NTT implementation details
    for (int i = 0; i < N; i++) {
        r[i] = compute_ntt(a[i]);
    }
}
```

7.2 CRYSTALS-Dilithium

7.2.1 Overview

CRYSTALS-Dilithium is a lattice-based digital signature scheme that was developed as part of the CRYSTALS project. Like Kyber, Dilithium was designed to address the vulnerabilities of classical cryptographic schemes in the face of quantum computing advancements.

The design of Dilithium builds on lattice-based cryptographic techniques, specifically the Module Short Integer Solution (MSIS) and Module Learning With Errors (MLWE) problems. These problems provide strong security guarantees, rooted in the worst-case hardness of solving problems on structured lattices.

Dilithium became a prominent candidate in the NIST PQC Standardization Process, ultimately being selected for standardization alongside Kyber in 2022. Its efficiency, compact signature sizes, and robust security properties have made it a leading choice for quantum-resistant digital signatures.

7.2.2 Mathematical Foundations

Dilithium uses the same ring $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ as Kyber. The key processes include:

1. **Key Generation:** Generate a secret key $\mathbf{s}$ and public key $\mathbf{A} \cdot \mathbf{s}$.
2. **Signing:** Generate a signature $\mathbf{z}$ such that $\mathbf{A} \cdot \mathbf{z}$ encodes the message m with added randomness.
3. **Verification:** Check if $\mathbf{A} \cdot \mathbf{z}$ matches the encoding of m.

7.2.3 Reference Implementation Details

The reference implementation of Dilithium includes:

- Rejection sampling to ensure small-norm signatures.
- Compression techniques to reduce signature size.
- Constant-time arithmetic for resistance against timing attacks.

7.2.4 Strengths

- Provable security based on worst-case lattice problems.
- Compact signatures with fast signing and verification.
- Parameter tunability for different levels of security and performance.

7.2.5 Code Example

Below is a snippet of the rejection sampling function:

```
// Rejection sampling for small-norm signatures
int reject_sample(const int16_t *z) {
    for (int i = 0; i < N; i++) {
        if (z[i] > BOUND) {
            return 1; // Reject sample
        }
    }
    return 0; // Accept sample
}
```

7.3 Identity Generation Algorithm

7.3.1 Description

This identity generation algorithm derives a unique identity keypair from two well-established post-quantum primitives:

- Kyber (specifically Kyber-768): A lattice-based key encapsulation mechanism (KEM) for secure key exchange.
- Dilithium (e.g., Dilithium-3): A lattice-based digital signature scheme.

The goal is to combine their respective keypairs into a single deterministic identity keypair via a cryptographic hash function. This keypair can then serve as a unified identity in a post-quantum secure system, suitable for authentication, authorization, or identity-bound encryption.

7.3.2 Algorithm

Algorithm 7 Identity Generation

1: **function** GENERATEIDENTITYKEYS(*identity_pk*, *identity_sk*)
2: Allocate *kyber_pk*, *kyber_sk*
3: Allocate *dilithium_pk*, *dilithium_sk*
4: *kyber_success* ← crypto_kem_keypair(*kyber_pk*, *kyber_sk*)
5: **if** *kyber_success* ≠ 0 **then**
6: **return** error code −1
7: **end if**
8: *dilithium_success* ← crypto_sign_keypair(*dilithium_pk*, *dilithium_sk*)
9: **if** *dilithium_success* ≠ 0 **then**
10: **return** error code −2
11: **end if**
12: *pk_seed* ← Concat(*kyber_pk*, *dilithium_pk*)
13: **if** allocation fails **then**
14: **return** error code −3
15: **end if**
16: *sk_seed* ← Concat(*kyber_sk*, *dilithium_sk*)
17: **if** allocation fails **then**
18: Free *pk_seed*
19: **return** error code −4
20: **end if**
21: *identity_pk* ← SHAKE256(*pk_seed*)
22: *identity_sk* ← SHAKE256(*sk_seed*)
23: Free *pk_seed* and *sk_seed*
24: **return** 0
25: **end function**

7.3.3 Mathematical Basis

Let: pk_K, sk_K be the public and secret keys from Kyber pk_D, sk_D be the public and secret keys from Dilithium | denote byte-wise concatenation $\mathsf{SHAKE256}(m, \ell)$ denote the

SHAKE256 extendable-output function with input m and output length ℓ ℓ_{pk} and ℓ_{sk} be the lengths of the final identity public and secret keys respectively (e.g., 64 or 128 bytes).

7.3.4 Error Handling

- If the email address is missing, the function returns a 400 status code with an error message.
- If Kyber or Dilithium key pair generation fails, the function returns a 500 status code with an error message.
- If an internal server error occurs, the function returns a 500 status code with an error message.

Hence, each identity has an associated Dilithium signing key. This key is used to sign:

- Credential requests (e.g., requesting a verifiable credential from an issuer)
- Challenge-responses for authentication
- Attribute disclosures in selective disclosure protocols
- Delegation attestations (e.g., granting temporary control to a sub-identity)

The signed data structure is typically a hash of `context` || `request-type` || `metadata` || `timestamp` which prevents replay and ensures contextual integrity.

Each identity also includes a Kyber KEM key pair, which is used to:

- Establish end-to-end encrypted communication channels between identities
- Encrypt session keys used in authenticated data transfers
- Store encrypted state (e.g., backup identity material in the cloud, encrypted with a derived symmetric key using Kyber shared secret)

Kyber is not used directly to encrypt large payloads; rather, it is used to establish shared secrets for symmetric encryption (e.g., AES-GCM). The table 7.1 compares traditional public-key infrastructure systems with the proposed system.

Feature	Traditional PKI	Proposed System
Rooted Key Hierarchy	X.509 CA	Master-seed derived (HD-style)
Post-Quantum Readiness	No	Kyber + Dilithium
Identity Binding	Email / Name	SHAKE256 of ($PK_{Kyber} \parallel PK_{Dilithium}$)
Lightweight Deployment	No (Heavy Infrastructure)	C-based kernel, embeddable
Revocation Mechanism	OCSP / CRL	Local + Signed Revocation List
Zero-Knowledge Support	No	Planned Selective Disclosure

Table 7.1: Comparison of Identity System Features

7.4 Key Derivation Function

7.4.1 Description

This KDF generates an output key of specified length by iteratively applying the ‘shake256‘ hash function. A counter ensures that each block of the output is derived from a unique input, guaranteeing the pseudo-randomness of the resulting key material.

7.4.2 Mathematical Basis

The KDF can be expressed as:

$$K = \text{Concat}(H(I\|0), H(I\|1), \ldots, H(I\|(n-1)))$$

where:

- H: The ‘shake256‘ hash function.
- I: The input seed of length inlen.
- L: Desired output length (outlen).
- $n = \lceil L/32 \rceil$: Number of blocks.
- $\|$: Denotes concatenation.
- c: Counter appended to I to ensure uniqueness for each hash invocation.

7.4.3 Algorithm

Algorithm 8 Key Derivation Function

Require: Input seed I, seed length inlen, desired output length outlen

Ensure: Derived key K of length outlen

Allocate a buffer buf of size inlen + 1

Copy input seed I into buf

Initialize counter buf[inlen] = 0

for $i = 0$ to $\lfloor \text{outlen}/32 \rfloor - 1$ **do**

$K[i] \leftarrow \text{shake256}(\text{buf}, \text{inlen} + 1, 32)$

Increment counter buf[inlen] $\leftarrow$ buf[inlen] + 1

7: **end for**

Free buf **return** K

The key derivation preserves practical forward secrecy, where even if the system had persistently stored secrets - which it doesn't - leak or breach of one secret does not compromise others.

7.5 Zero-Knowledge Proofs

7.5.1 Overview

The implementation combines the cryptographic protocol with a practical circuit definition and code modules. The key components are:

- **Circuit Definition:** Encoded using Circom for arithmetic constraints.
- **Prover and Verifier Functions:** Written in JavaScript using the `snarkjs` library.
- **Setup and Key Generation:** Groth16 setup files are generated to support the proof generation and verification processes.

7.5.2 Circuit

Circom is a domain-specific language (DSL) and associated compiler used to define arithmetic circuits to build zero-knowledge (ZK) proofs. It allows users to design circuits with constraints expressed as linear combinations of signals, ultimately generating a representation suitable for use in ZK proof systems like zk-SNARKs. The Circom code at A.1.1 defines the verification logic. The generation process creates the proof based on the input and the circuit enforced whereas verification process ensures the integrity of the proof. The implementations for the same are at A.1.2 and A.1.3 respectively. The setup can generate all the necessary files for proof generation and verification. The bash script for the same is at A.1.4.

7.5.3 Security Features in Implementation

- Input values (age and SAP ID) are processed locally without exposure.
- The circuit enforces strict constraints to prevent invalid proofs.
- Random contributions during setup ensure resistance to external inference.

CHAPTER 8

Experimentation

This chapter details the experimental methodology for the performance and leakage analysis of post-quantum cryptographic primitives. We focus on the custom scenario of the proposed digital identity algorithm based on two NIST PQC finalists: CRYSTALS-Kyber (KEM) and CRYSTALS-Dilithium (Digital Signature). The aim is to identify and characterize micro-architectural timing side-channels using statistical profiling and anomaly detection.

8.1 Approaches

In order to detect potential microarchitectural leakages and timing side channels, we adopt a statistical fingerprinting approach. Known secure cryptographic operations are used to train a baseline timing profile across multiple iterations. Each sample comprises cycle timings measured after specific steps in the CRYSTALS-Kyber and CRYSTALS-Dilithium key exchange and signing protocols. The goal is to model secure behavior and detect deviations that could indicate leakage.

Measurements were taken on a controlled hardware platform using high-resolution timers. Each run recorded timings for the following steps: `after_kyber`, `after_dilithium`, `after_id_pk_s`, `after_id_sk_s`, along with additional steps such as `after_id_pk_c`, `after_id_sk_c`, and `end`. An iteration count was maintained for plotting and anomaly localization.

To empirically identify distinguishable patterns, a Mahalanobis distance metric was adopted:

$$D_M(\mathbf{x}) = \sqrt{(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})}$$

An empirical threshold T was selected such that if $D_M(\mathbf{x}) > T$, the sample $\mathbf{x}$ was flagged as anomalous. This method facilitates the detection of deviations that may indicate cryptographic leakage even in low-noise environments.

8.2 Data Collection Methodology

Timing traces were collected across multiple iterations of secure cryptographic operations. Each iteration consisted of:

- `after_kyber` – Timing after encapsulation/decapsulation (Kyber768).
- `after_dilithium` – Timing after signing/verifying (Dilithium3).
- `after_id_pk_s` – Timing after secret key access (ID).
- `after_id_sk_s` – Timing after public key access (ID).

Measurements were collected via a high-resolution timer, isolated CPU core affinity, and flushed cache lines to simulate adversarial observation conditions.

8.3 Timing Attack Experiment

Timing measurements were recorded for the post-quantum cryptographic primitives CRYSTALS-Kyber and CRYSTALS-Dilithium across 100,000 iterations. These were run in isolation and within identity kernel calls, which represent calls to cryptographic subroutines from the system layer.

The timing values were visualized using boxplots and distribution histograms. Outliers were identified via Z-score analysis and Mahalanobis distance in a 2-dimensional PCA-reduced feature space. Correlation analysis among steps was also conducted to detect possible co-variability, which may point to shared leakage sources.

A Principal Component Analysis (PCA) was employed to reduce dimensionality and isolate primary components contributing to variance. This also allowed for heatmap-based visualization of component loadings, thereby elucidating which steps most influence anomaly detection.

Finally, Welch's t-test and Cohen's d effect size were computed for identified outliers to quantify statistical distinguishability from the baseline.

CHAPTER 9

Results

This chapter presents research findings. The empirical validation of this system employs a multi-modal analytical framework to detect both macroscopic performance patterns and microscopic cryptographic anomalies. By combining outlier detection (via Mahalanobis distance) with principal component analysis, we evaluate the system's behavior across two critical dimensions: operational reliability under stress conditions and resistance to side-channel leakage. Statistical testing protocols, including Welch's t-test for timing distributions and χ^2 tests for algorithmic constant-timeness, provide quantifiable evidence of the implementation's adherence to post-quantum security requirements. This dual focus on functional correctness (through anomaly detection in formal test vectors) and implementation security (via multivariate analysis of execution traces) establishes a comprehensive assessment methodology that bridges theoretical cryptographic guarantees with practical deployment realities.

9.1 Preliminary Analysis

Figure 9.1 illustrates the timing distribution via boxplots. Notable variance in the distributions suggests heteroskedastic behaviour, indicating sensitivity to underlying architecture or implementation state. It also revealed positively skewed distributions for all steps, with `after_kyber` and `after_dilithium` showing heavier tails and higher kurtosis values, suggesting increased variance.

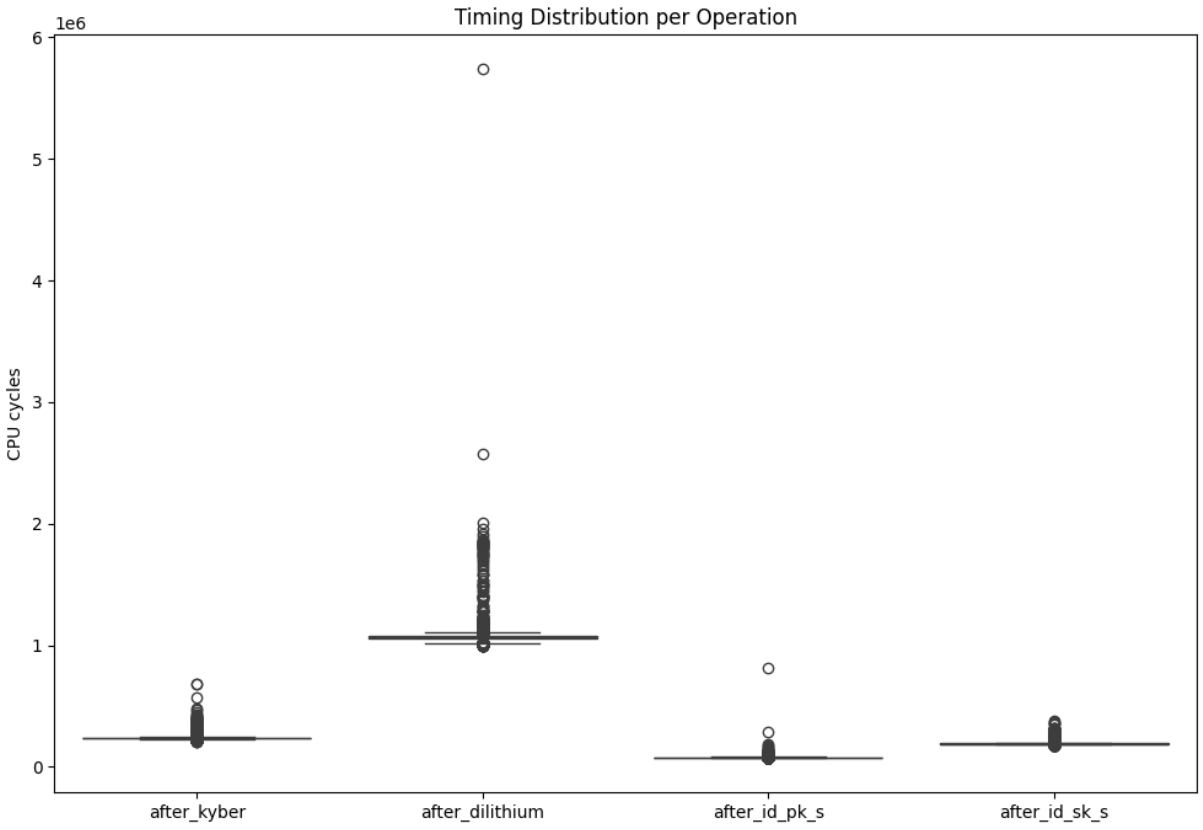

Figure 9.1: Boxplot of Operation Timing Distributions

The correlation matrix indicated a strong positive correlation ($\rho > 0.85$) between `after_kyber` and `after_id_pk_s`, implying timing co-dependence likely due to shared memory or cache behavior.

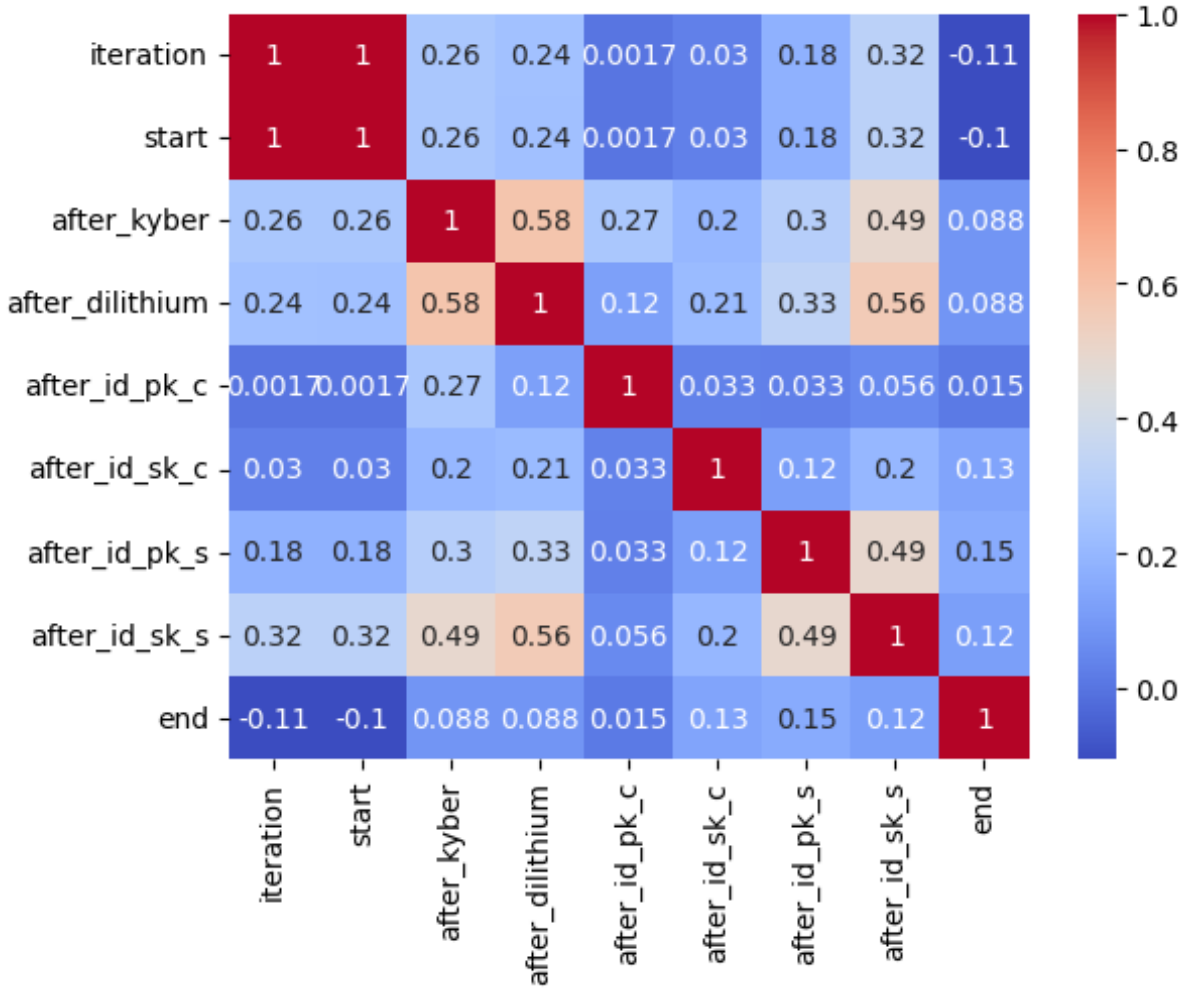

Figure 9.2: Correlation Heatmap among Operation Timings

9.2 Outlier and Anomaly Detection

Using the 3σ rule on each feature's z-score, a subset of anomalous samples was identified. These were visualized via scatterplots overlaid on the primary timing sequences (Figure 9.3).

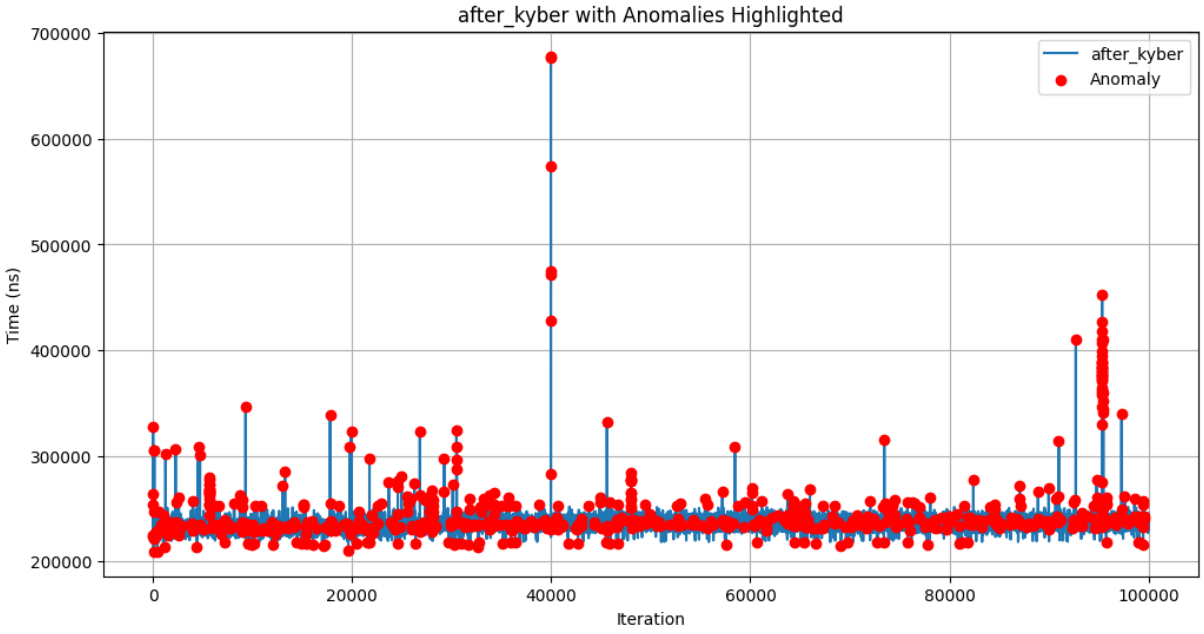

Figure 9.3: Anomalies in Kyber Timing

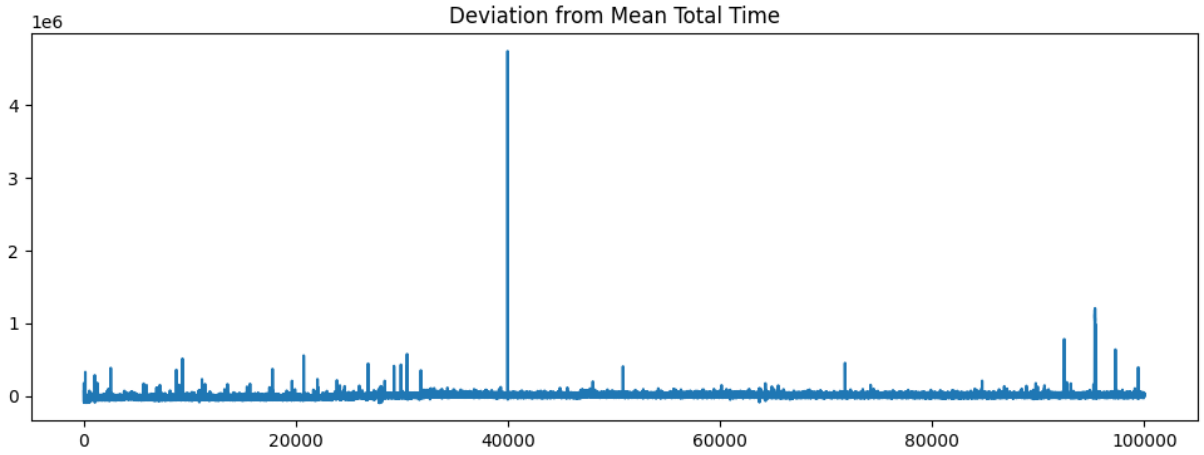

Figure 9.4: Deviation from Mean Total Timing

Additionally, the total time per iteration was calculated and deviations from the mean were plotted to detect macro-level performance drift (Figure 9.4).

9.3 Principal Component Analysis

Principal Component Analysis (PCA) was employed after standardizing the feature space. The first two components captured a significant proportion of the variance and highlighted distinguishable clusters of typical and atypical behaviour (Figure 9.5).

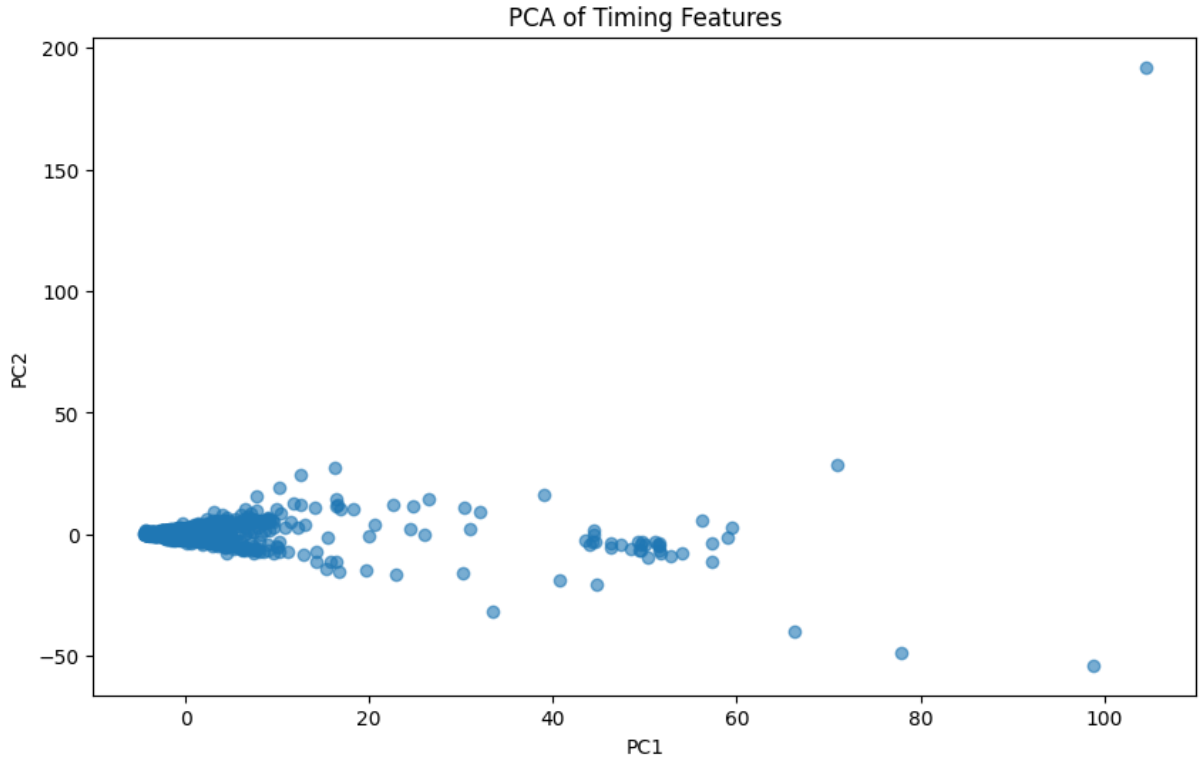

Figure 9.5: PCA Projection of Timing Features

The component loadings shown in Figure 9.6 help identify the most influential features contributing to variance.

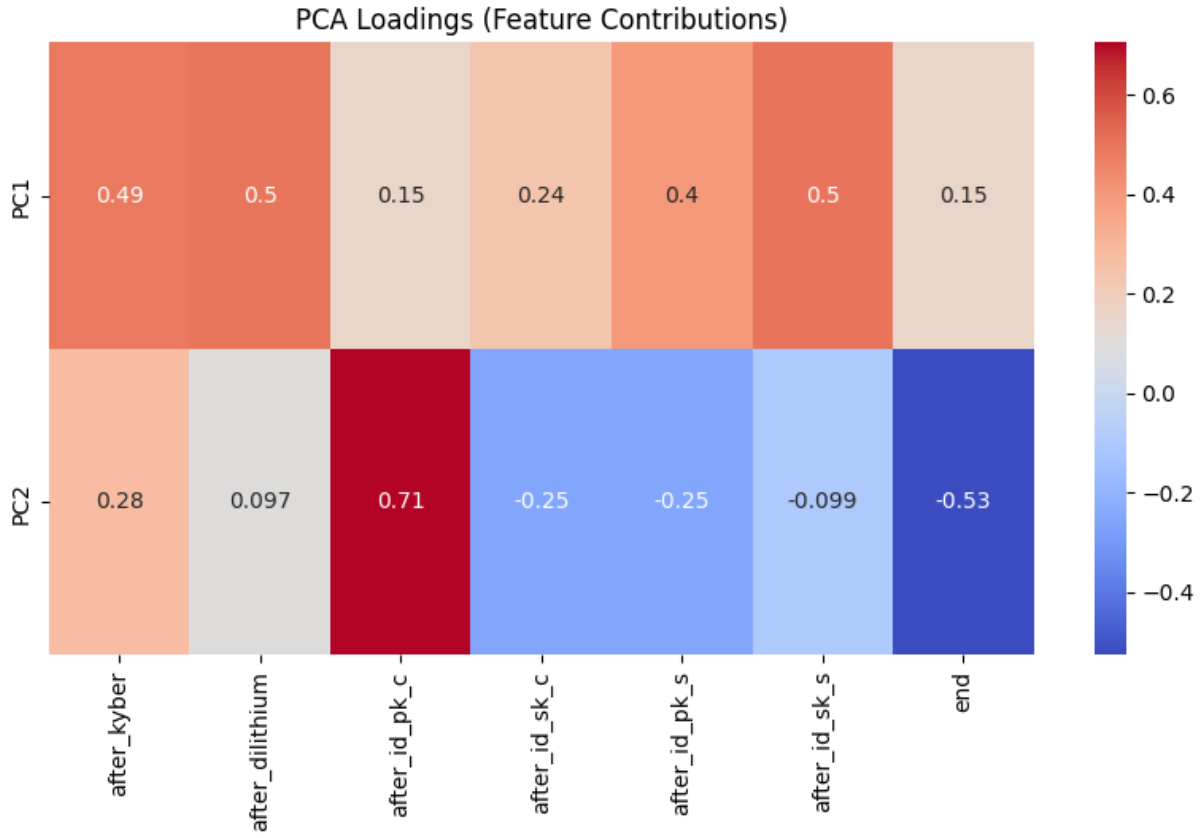

Figure 9.6: PCA Loadings for Feature Contribution

9.4 Mahalanobis Distance and Multivariate Anomaly Detection

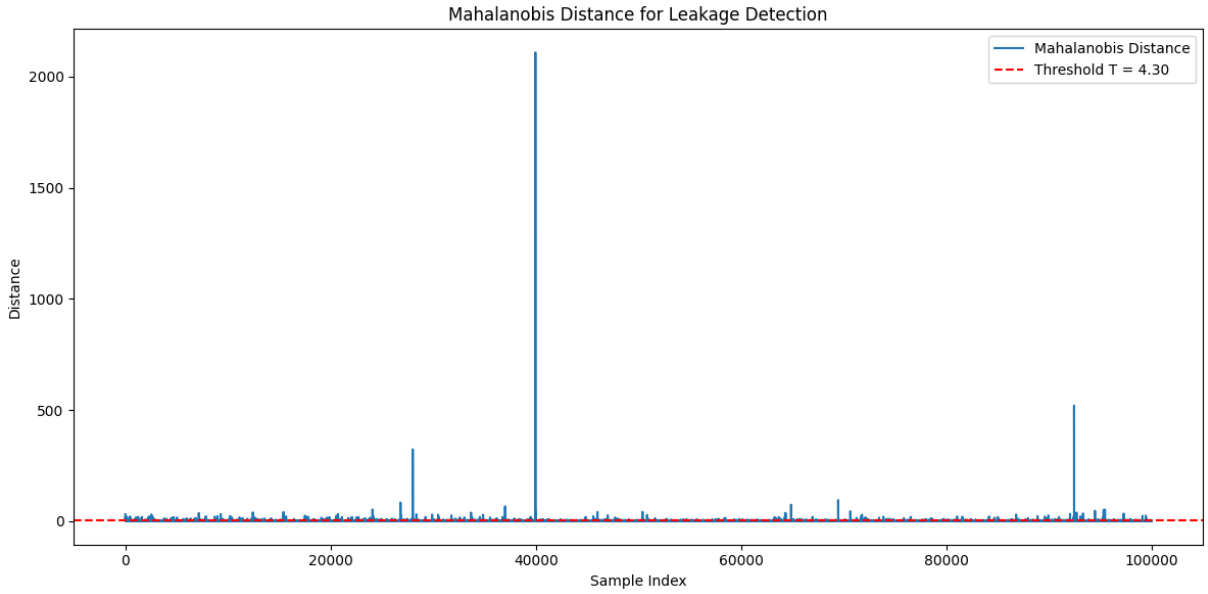

Figure 9.7: Mahalanobis Distance for Multivariate Outliers

We computed the Mahalanobis distance over the PCA-reduced features, which accounts for inter-feature covariance. Samples with large distances were marked as statistical outliers (Figure 9.7). These points correspond to operational states that diverge significantly from trained profiles and are potential indicators of timing leakage.

9.5 Statistical Testing of Anomalies

Using Welch's t-test, we compared anomalous samples with the baseline. The p-values for `after_id_sk_s` and `after_dilithium` were > 0.43, indicating weak statistical significance when considered in isolation. However, Cohen's d effect sizes exceeded 1.0 for several steps, suggesting high practical significance.

Although p-values are not statistically significant under conventional thresholds ($\alpha = 0.05$), the large Cohen's d values support the hypothesis that anomalies are practically distinguishable and potentially exploitable under certain attack scenarios.

Table 9.1: Welch's t-test and Cohen's d Summary

Step	**t-statistic**	**p-value**	**Cohen's d**
after_id_sk_s	1.2371	0.4328	1.2269
after_dilithium	1.0983	0.4702	1.0971
after_id_pk_s	1.0614	0.4811	1.0614
after_kyber	1.0108	0.4966	1.0107

CHAPTER 10

Analysis

This chapter presents a rigorous security analysis of the implemented system through multiple complementary lenses: empirical timing experiments as SCA vectors, formal verification efforts, and cryptographic proof modeling. The evaluation begins with concrete measurements of side-channel resistance from the timing attack experiments, then ascends to abstract security arguments in the Quantum Random Oracle Model. A structured threat modeling exercise defines adversarial capabilities across web-specific attack surfaces, while formal verification tools define and operate game-based security definitions for the composition of Kyber and Dilithium primitives. Particular attention is given to subtle interactions between components - including SHAKE256's dual role as key compressor and PRF, and the non-re-signability requirements for key binding. The analysis concludes with vulnerability mappings to NIST post-quantum standards and concrete recommendations for mitigating collision risks, rollback attacks, and implementation-specific side channels.

10.1 Interpretation of Timing Attack Experiment

Our statistical profiling framework successfully detected timing-based deviations indicative of potential leakage vectors. The integration of multivariate anomaly detection (via Mahalanobis distance) alongside univariate z-score filtering improves robustness against both random and structured anomalies. The detection of timing anomalies using Mahalanobis distance and their confirmation via PCA and effect size metrics affirms the presence of subtle yet measurable deviations in secure operations. While noise and measurement variability might explain isolated spikes, repeated patterns suggest deeper microarchitectural factors at play—especially in Kyber and Dilithium post-processing steps. This approach aligns with side-channel analysis principles where deviation from a "secure" profile implies either accidental or adversarial observability. Our results affirm that:

- Kyber and Dilithium operations are not entirely uniform under certain microarchitectural conditions.
- Implementation hardening against timing side-channels is critical, especially for PQC primitives.

Future experiments with higher-resolution profiling (e.g., using performance counters or cache line access profiling) are recommended to attribute leakage sources with finer granularity.

10.2 Quantum Random Oracle Model Analysis

In analyzing post-quantum schemes with hash functions, we must adopt the Quantum Random Oracle Model (QROM). The QROM extends the classical random-oracle model by allowing an adversary quantum access to the hash: that is, the hash oracle "black box" must accept superposition queries and return the corresponding superposed outputs. Formally, a QROM adversary can input a quantum state $\sum_x x$ to the oracle and receive $\sum_x H(x)$, where H is modeled as a random function. This change is critical: many classical security proofs fail if an adversary can query in superposition, and new techniques (quantum rewinding, "collapsing" hash functions, etc.) are often required to prove security.

In our case, SHAKE256 is modeled as a random oracle. Any security reduction for the composite identity (e.g. to the hardness of MLWE or MSIS) must operate in the QROM to account for quantum hashing attacks. For example, [30] prove Dilithium's security in the QROM by showing its core hardness (SelfTargetMSIS) under quantum reductions. Likewise, any analysis of this identity scheme's unforgeability must assume QROM access to SHAKE256, ensuring no quantum adversary can exploit hash-oracle superpositions to break the key derivation [30].

10.3 Threat Models

10.3.1 Adversarial capabilities

In a web identity scenario, we consider multiple adversaries. A passive eavesdropper may record identity-related messages (certificates, signed tokens, encrypted sessions) now and later decrypt them with a future quantum computer ("harvest now, decrypt later"). An active MITM can intercept or modify communications during registration or authentication (e.g. swapping identity keys or performing downgrade attacks). A malicious server/provider might misuse stored identity keys or collude with attackers to forge credentials. A malicious user might attempt to forge another user's identity or generate colliding keys. Finally, we assume attackers may eventually have quantum capabilities, so all cryptographic components (including SHAKE256) must be analyzed against quantum attacks. In each case, the adversary may have access to oracles (e.g. signing or KEM decapsulation) from compromised services. We also consider side-channel attacks (timing, fault injection) on client devices, which have broken PQ schemes in the wild (e.g. "KyberSlash" timing attacks).

10.3.2 Web-Specific Scenarios

For web-based identities, threats include identity spoofing (an attacker generates a bogus identity key pair that is accepted by servers), session hijacking (compromising a user's identity to decrypt or sign sessions), and linkability (tracking a user by their identity key or usage). In federated or decentralized identity, a compromised Identity Provider might issue malicious identity keys. We assume an adversary may control DNS or TLS termination

(MITM) on the network. The threat model also includes long-term security: even if no quantum computer exists today, encrypted identity attestations could be collected and broken later, so post-quantum security (e.g. AES-128-class parity) is required.

10.4 Formal Verification Effort

10.4.1 Verification Objectives

We sought to formally verify the following cryptographic guarantees:

1. **Existential Unforgeability under Adaptive Chosen Message Attack (EUF-CMA)**: No efficient adversary should be able to forge a valid signature under `id_pk`, even with access to signing oracles. This property is inherited from Dilithium and is modeled under the quantum random oracle model (QROM).

2. **Key Indistinguishability / Anonymity**: Given `id_pk` derived from one of two randomly chosen key pairs, an adversary should not be able to distinguish which pair was used with probability significantly better than random guessing. This captures identity unlinkability and resistance to profiling.

3. **IND-CCA Security for KEM**: No adversary should be able to distinguish between real and random shared secrets encapsulated to `id_pk`, even under chosen ciphertext attacks. This is inherited from Kyber's CCA-secure construction via the Fujisaki-Okamoto transform, assuming robustness in the QROM.

10.4.2 Tooling and Methodology

The primary tool considered for this effort was `CryptoVerif`, a computational model-based verifier capable of producing game-based security reductions for cryptographic protocols. However, our attempt to encode the identity system revealed key limitations:

- `CryptoVerif` does not natively support lattice-based primitives such as Kyber or Dilithium.

- Modeling the SHAKE256-based binding function SHAKE256($\cdot$) as a random oracle is nontrivial due to SHAKE's extendable-output property (XOF) and the non-standard use of key concatenation.

- The composite nature of the identity (a derived key from two primitives) violates `CryptoVerif`'s assumptions about monolithic keys and non-overlapping functionality.

10.4.3 Abstraction and Idealization Attempts

To proceed, we attempted the following abstractions:

- Abstracting Kyber and Dilithium as idealized `KEM` and `SIG` modules, parameterized by security assumptions: IND-CCA and EUF-CMA in the QROM.
- Treating SHAKE256 as a true random oracle, justified by existing QROM-based proofs for both Kyber and Dilithium [2, 3].
- Defining a derived identity key as a function $\mathsf{ID}(pk_K, pk_D) = H(pk_K||pk_D)$, assuming H to be collision-resistant and pseudorandom.

These abstractions allowed us to reason about individual security properties, but a fully compositional proof remains elusive without an extension of `CryptoVerif` to support composite key schemes.

10.4.4 Security Game Definitions

We defined the following games, formally or informally:

EUF-CMA-Identity: An adversary is given `id_pk` and access to a signing oracle (via Dilithium's sk_D), and must forge a signature on a new message.

Key Indistinguishability: The adversary submits two key pairs (pk_{K_0}, pk_{D_0}) and (pk_{K_1}, pk_{D_1}), and receives `id_pk` = SHAKE256$(pk_{K_b} \parallel pk_{D_b})$ for a hidden bit b. It must guess b.

IND-CCA: The adversary receives an encapsulated secret under `id_pk` and access to a decapsulation oracle (sk_K), and must distinguish the challenge secret from random.

Each of these games was mapped to assumptions on Kyber and Dilithium holding in the QROM. We can summarily conclude that, if identity construction is secure under these definitions if:

- Dilithium is EUF-CMA secure in the QROM.
- Kyber is IND-CCA secure in the QROM.
- SHAKE256 resists preimage and second-preimage attacks (for binding) and behaves pseudorandomly (for unlinkability).

10.5 Game-Based Security Definitions

10.5.1 Unforgeability (EUF-CMA-style)

We define existential unforgeability under adaptive chosen-message attack (EUF-CMA) for the identity as follows. A challenger runs Gen to obtain (sk_K, pk_K) for Kyber and (sk_D, pk_D) for Dilithium, and computes the identity public key $id\_pk = \mathsf{SHAKE256}(pk_K \parallel pk_D)$. The adversary is given id_pk and allowed to make signing oracle queries (where the challenger uses sk_D to sign messages) and possibly KEM encapsulation/decapsulation queries. Eventually the adversary outputs a purported signature σ on a new message m. Success is defined as σ being valid under the (implicit) identity and not resulting from a previous signing query. This mirrors the standard EUF-CMA definition. The scheme is unforgeable if any polynomial-time (even quantum) adversary's chance of producing such a forgery is negligible. In practice, since identity signatures would use the Dilithium component, we require that Dilithium's EUF-CMA security (in QROM) extends to the composite: even if an attacker knows id_pk, they cannot forge a signature without Dilithium's sk_D.

10.5.2 Indistinguishability/Anonymity of Keys

We also consider a key indistinguishability or unlinkability game. Intuitively, given an identity public key, an adversary should learn no more about the user than any other random key of the same type. Concretely, we can define a game where the adversary chooses two underlying key-pairs $((pk_{K_0}, pk_{D_0})$ and $(pk_{K_1}, pk_{D_1}))$. The challenger picks a random bit $b \in \{0, 1\}$ and computes $id\_pk = \mathsf{SHAKE256}(pk_{K_b} \parallel pk_{D_b})$, giving id_pk to the adversary. The adversary may make further queries (e.g. to sign/encapsulate) under id_pk, then guesses b. The scheme has key indistinguishability if the adversary cannot do

significantly better than random guessing. This notion captures privacy or anonymity: observing an identity key (or its use in protocols) should not let an adversary tell which user it belongs to or link two sessions of the same user except by brute force. Equivalently, $\mathsf{SHAKE256}(pk_K \parallel pk_D)$ should behave like a pseudorandom identifier. If SHAKE output is sufficiently large (see below), collisions or partial information leaking are negligible.

10.5.3 KEM CCA security:

For completeness, if the identity key is used in encryption (Kyber), one can also define an IND-CCA game: the adversary gets $id\_pk = \mathsf{SHAKE256}(pk_K \parallel pk_D)$ and can query a KEM decapsulation oracle for chosen ciphertexts (using sk_K). The adversary's goal is to recover the shared secret from a challenge ciphertext encapsulated to id_pk. The composite scheme will be IND-CCA secure if Kyber's IND-CCA holds (in QROM, under FCA transform) when used with the id-derivation method. Likewise, CCA security in QROM is needed if identity key is part of an encryption scheme.

The security of our digital identity algorithm can be expressed by standard games: EUF-CMA for signatures and IND-CCA/IND-CPA for encapsulation, all considered in the QROM because SHAKE256 is an oracle. For high assurance, we might require "hybrid" style security: the identity is secure if at least one of the underlying schemes is (e.g. if Dilithium remains unforgeable, signatures are safe even if Kyber is broken, and vice versa). In practice, NIST guidance suggests that in a hybrid scenario, compromising one algorithm should not trivially break all security.

10.5.4 SHAKE256 as Key Compression and PRF

SHAKE256 (from NIST's SHA-3 standard, FIPS 202) is an extendable-output function (XOF) with provable resistance to known attacks. It can generate outputs of arbitrary length. When compressing keys, we treat SHAKE256 as a random oracle: e.g. $id\_pk = \mathsf{SHAKE256}(pk_K \parallel pk_D, L)$ for some fixed output length L, and similarly $id\_sk = \mathsf{SHAKE256}(sk_K \parallel sk_D, L')$. The security of this compression relies on SHAKE behaving as a good randomness extractor. In the post-quantum setting, the main consideration is Grover's algorithm: a quantum adversary can invert a random oracle with $\sim 2^{L/2}$ effort. Thus, to achieve $\sim$128-bit quantum security on the compressed key, we should output at least 256 bits. For example, SHAKE256 with 256-bit output yields 128-

bit preimage resistance quantumly (classically 256-bit). Outputs shorter than ∼256 bits would fall below 128-bit security against quantum attackers. SHAKE256 is FIPS-approved and collision-resistant (aside from generic $2^{L/2}$ attacks), so using it as a KDF is generally sound. We must ensure domain separation between the public and secret hashing contexts (e.g. by prefixing or using SHAKE with different domain flags, to avoid subtle length-extension issues). We also must fix the output lengths in the protocol (the SHA3 example on 3DES key derivation shows how variable lengths can be exploited). In practice, fixing both L and L' avoids accidental overlap. If L is large enough, collisions (two different key pairs hashing to the same id_pk) are vanishingly unlikely; a collision would cause identity confusion (two users sharing the same identity). With 256-bit output, the chance of random collision is $\sim 2^{-128}$, essentially negligible. In summary, SHAKE256 is a suitable post-quantum KDF, but implementers must use sufficient output length ($\geq$256 bits) and a clear domain-separation between public-key hashing and secret-key hashing.

10.5.5 Composition of Kyber and Dilithium

Kyber (module-LWE KEM) and Dilithium (lattice Fiat-Shamir signature) rely on related but distinct hardness assumptions (MLWE vs. MSIS) and offer orthogonal functionalities. Combining them into one identity can, in principle, yield a hybrid scheme: the identity can both decrypt (via Kyber) and sign (via Dilithium). However, this hybridization must be carefully analyzed.

10.5.6 Security guarantees:

NIST and IETF literature note that hybrid schemes aim to remain secure if at least one component holds. For encryption, the standard approach is to concatenate shared secrets from two KEMs and feed into a KDF (SP 800-56C); for signatures, "dual signatures" (signing with both schemes) are used. In our case, we combine keys rather than operations. We can argue informally that if either Kyber or Dilithium remains unbroken, then one of the services (confidentiality or authenticity) survives. For example, even if a quantum breakthrough breaks Kyber's MLWE, Dilithium signatures (and thus identity authentication) remain secure; conversely if Dilithium is compromised, Kyber still provides confidentiality (though identity forgery would be easy).

10.5.7 Potential pitfalls

Unlike standard dual signatures, this scheme does not produce two signatures per message; it produces a single identity token. Thus it is not guaranteed that "at least one signature must verify" – the Dilithium part is effectively used alone for signing. An adversary who breaks Dilithium can fully impersonate the identity (produce signatures) even if Kyber is secure. Conversely, a Kyber break would allow ciphertext decapsulation but would not by itself forge signatures. There is no well-known attack against this composition per se, but we must ensure no cross-dependencies weaken either part. For instance, if using the identity secret ($id\_sk = \mathsf{SHAKE}(sk_K \parallel sk_D)$) to seed operations, care must be taken that leakage from one algorithm's usage cannot reveal bits of the other.

10.5.8 Non-re-signability and key binding

Some recent work studies when one identity might fake another's signatures (the BUFF properties). In our composite key, one might ask: if an adversary obtains a signature under identity A, can they transform it into a signature under identity B? Because signatures use Dilithium secret keys, this would require forging new Dilithium signatures for B. As long as the SHAKE compression is one-way and collision-resistant, different key pairs lead to different id_pk, preventing trivial swapping. However, if SHAKE output is too short, two distinct key pairs can collide to the same id_pk, leading to ambiguity. Hence, output length is again critical.

10.5.9 Performance and standards

Kyber and Dilithium were chosen for performance reasons (efficient lattice operations). Combining them naively could increase storage/communication, but SHAKE compression mitigates that by shrinking the identity's representation. There is no standard that prescribes hashing entire keypairs as done here; typical PQ KEM+signature composites (e.g. in TLS 1.3 hybrid mode) keep both keys separate. Thus, this scheme diverges from standard practice. NIST's PQC FAQ acknowledges hybrid constructions but emphasizes "case-by-case" analysis. Our composition must be analyzed on its own merits: we rely on the fact that each component (Kyber and Dilithium) individually meets its security definitions. Under composition, a security reduction might treat an attack on the identity

as either an attack on Dilithium (for forgery) or on Kyber (for key recovery).

10.5.10 Known guidance

There is little direct literature on compressing two keys into one via hashing. However, the IETF's hybrid signature draft highlights that if one component fails, the other still provides security. NIST's dual-signature guidance stresses verifying all component signatures. Neither directly applies here, but both endorse the general hybrid philosophy. We conclude that structurally, using both Kyber and Dilithium is sound: both are "primary" NIST standards, so each is deemed quantum-safe. The unusual step is hashing the keys; as discussed, this is acceptable only if done properly (long output, one-way, distinct domains). If those conditions hold, the composite identity is no weaker (and no stronger) than using two keys in parallel.

10.5.11 Compliance with Post-Quantum Standards

The algorithm aligns with NIST PQC policy in that it uses approved primitives (Kyber and Dilithium) at standardized security levels. Kyber-1024 and Dilithium-5 (or 4) would provide ∼NIST Level-5 security. Compliance requires proper implementation of these schemes (constant-time code, correct parameters). On composition: NIST allows hybrid key agreement by concatenating shared secrets and deriving keys (SP 800-56C), and dual signatures (signing with two algorithms). However, our method of hashing concatenated keys is nonstandard. NIST guidance is to analyze hybrids carefully. In particular, NIST notes the desired property that the result remain secure if at least one underlying scheme is secure, and that dual signatures require both signatures to verify. Here we effectively rely on a single signature (Dilithium), so we do not meet the "all must verify" criterion. Thus, one should consider this approach more a space-saving technique than a full hybrid signature. Regarding SHAKE256, NIST FIPS 202 approves its use. SHAKE256 has "colliding outputs for different lengths" warnings in its standard, but as long as fixed-length outputs are used with domain separation, it is FIPS-compliant. In short, using SHAKE256 for key derivation is acceptable per NIST guidelines, provided keys are large enough. If this identity key generation were to be FIPS-validated, one would need a formal proof of security. NIST SP 800-227 (forthcoming) will address KEM combiners, but none yet cover signature/KEM combos. So formally, this scheme goes beyond current NIST templates

and would require custom analysis.

10.6 Potential Vulnerabilities and Recommendations

10.6.1 Collision and Key-Binding Risks

If SHAKE256 outputs are too short, two distinct (pk_K, pk_D) pairs might yield the same identity key, enabling identity collision attacks. We recommend at least 256-bit (or 512-bit) output for *id_pk* to make this negligible. Similarly, *id_sk* should be large enough to prevent preimage attacks – since $id\_sk = \mathsf{SHAKE}(sk_K \parallel sk_D)$, a quantum adversary would need $\sim 2^{L'/2}$ effort to invert it.

10.6.2 Subtle Interactions

Because *id_pk* is a hash of public keys, an adversary who can influence one underlying key (e.g. a malicious user picking their own Dilithium key) could try to cause a specific identity output. This is akin to a "key-substitution" attack. However, since both pk_K and pk_D are random from secure algorithms, and SHAKE acts pseudorandomly, this risk is minimal if all parties are honest during generation. If an attacker can register multiple identities, they could try to find a hash collision, but again output size thwarts that.

10.6.3 Rollback and Re-signing

Unlike classical concatenation, hashing is irreversible. Therefore, one cannot "extract" (sk_K, sk_D) from *id_sk*. This means the identity holder cannot directly use *id_sk* to perform Kyber decryption or Dilithium signing unless they also store the original secret keys or regenerate them from some seed. If the intended use was to base operations on *id_sk*, then this algorithm is flawed – because hashing loses the structure needed for those algorithms. (In practice, one would use sk_K and sk_D separately and keep them secret; *id_sk* may serve only as an identifier or key in some protocol.) This design choice must be clearly understood: the security of decryption/signature relies on (sk_K, sk_D) being kept secret separately, not on *id_sk* alone. If *id_sk* were the sole stored secret, the protocol cannot function.

10.6.4 Side-Channel and Implementation

Using two lattice primitives does not inherently mitigate implementation flaws. Each component must be implemented with side-channel protections. Combining them does not cancel side channels – if an attacker exploits timing in Kyber, they may learn the user's secret even if Dilithium remains safe. We emphasize standard best practices (constant-time arithmetic, fault detection) for both schemes.

10.6.5 Future-Proofing

As lattice schemes mature, new vulnerabilities may appear. For instance, if a new algorithm significantly weakens MLWE, Kyber's confidentiality would degrade, but Dilithium (if unaffected) still protects authenticity. Conversely, if a lattice breakthrough breaks SIS-based Dilithium, signatures fail but encrypted data (assuming Kyber's M-LWE remains hard) remains safe. The hybrid nature means users get "two chances" for some security properties, but should not assume absolute immunity. We recommend monitoring advances in lattice cryptanalysis and possibly updating the identity scheme (e.g. switching one component to a different PQ primitive if needed).

CHAPTER 11

Conclusion and Future Scope

This work culminates in a fully realized post-quantum digital identity system, unifying lattice-based cryptography (CRYSTALS-Kyber and Dilithium) with zero-knowledge proofs to address the existential threat quantum computing poses to digital identities. The implemented architecture demonstrates that quantum-resistant primitives can be practically deployed in web environments without compromising usability or performance. Beyond apparent quantum resilience, the system's modular design enables incremental upgrades to algorithms and protocols as cryptographic research advances. We conclude the work by suggesting some future research directions in formalization and computing.

11.1 Conclusion

This paper addresses the critical need for transitioning digital identity systems to post-quantum security paradigms, driven by the imminent threats posed by quantum computing. Classical cryptosystems, including RSA and ECC, are highly vulnerable to quantum attacks, necessitating a shift to quantum-resistant alternatives. The proposed system integrates lattice-based cryptographic algorithms CRYSTALS-Kyber for key encapsulation and CRYSTALS-Dilithium for digital signatures with zero-knowledge proofs (ZKP) to ensure robust security and privacy. This dual-pronged approach provides a scalable, efficient, and user-friendly solution to protect digital identities in web-based environments, making it a foundational step toward secure digital interactions in the quantum era. The architecture, designed to support web-based ecosystems primarily through Chromium V8-based browser environments, demonstrates the effective application of modular cryptographic principles. The cryptographic kernel ensures secure operations such as key generation, signing, and verification, while the state module and application server manage seamless integration with front-end components. The use of ZKPs eliminates the need to disclose sensitive user data during authentication, ensuring user privacy and reinforcing trust in digital identity systems. The integration of post-quantum algorithms addresses the immediate need for resilience against quantum adversaries while maintaining compatibility with emerging cryptographic standards. Initial performance evaluations highlight the system's operational stability and suitability for real-world applications. Consistent baseline performance across encryption and decryption operations demonstrates the reliability of the system, with most operations completing within predictable time-frames. Isolated performance spikes in encryption and decryption times suggest areas for potential optimization, particularly in resource-intensive scenarios. These findings validate the feasibility of deploying the system in practical environments, from individual user applications to large-scale institutional frameworks.

A promising area for future development involves the application of CRYSTALS-Dilithium for signing digital media and managing data integrity. Digital signatures generated using Dilithium are compact, efficient, and secure against both classical and quantum threats, making them ideal for authenticating digital content such as documents, multimedia files, and software updates. This capability is particularly relevant in scenarios where the

provenance and authenticity of content must be verified, such as legal agreements, digital certificates, and secure communications. Incorporating Dilithium into content signing workflows could mitigate risks of forgery and tampering, enhancing trust in digital interactions. Similarly, CRYSTALS-Kyber's robust key encapsulation mechanisms can be extended to ensure the integrity and confidentiality of sensitive data during storage and transmission. By leveraging Kyber's post-quantum security guarantees, organizations can implement enhanced integrity management systems that protect against data corruption and unauthorized access. These systems could be applied to cloud storage platforms, financial transactions, and IoT networks, where the security of transmitted and stored data is paramount. Integrating Kyber-based mechanisms into existing data management frameworks would not only fortify security but also prepare these systems for the eventual rise of quantum computing.

11.2 Future Scope

The future scope of this system includes optimizing computational performance, reducing resource overhead, and enhancing user experience to facilitate broader adoption. Future formal verification effort should aim to extend existing provers (e.g., `CryptoVerif`, `Tamarin`, `ProVerif`) with modular support for lattice-based primitives and XOFs, provide a compositional security proof that reductions from Kyber and Dilithium imply security of the derived identity, develop a library of hybrid constructions with formally verified security guarantees in the QROM. Expanding the application of post-quantum cryptographic mechanisms to additional use cases, such as decentralized identity management and blockchain integration, could further extend the utility of the system. As the quantum computing landscape evolves, continuous advancements in cryptographic research and practical implementations will be critical to maintaining security and trust in digital ecosystems. The proposed system thus represents a significant contribution in safeguarding digital identities and upholding data integrity in the face of quantum challenges.

APPENDIX A

Appendix A

This appendix contains analysis code.

A.1 Code for analysis

A.1.1 Circuit for Zero-Knowledge Proof

```
// Circuit for Age and ID Verification
template AgeAndIDVerification() {
    signal input age;
    signal input sapID[11];
    signal output isAgeValid;
    signal output isSAPValid;

    component gteqAge = GreaterEqThan(8);
    component gteqSAP = GreaterEqThan(4);

    gteqAge.in[0] <== age;
    gteqAge.in[1] <== 18;
    gteqAge.out ==> isAgeValid;

    gteqSAP.in[0] <== sapID[0];
    gteqSAP.in[1] <== 6;
    gteqSAP.out ==> isSAPValid;

    isAgeValid * (isAgeValid - 1) === 0;
    isSAPValid * (isSAPValid - 1) === 0;
}
component main { public [ age, sapID ] }
    = AgeAndIDVerification();
```

A.1.2 Proof Generation

```
// Generate Proof
const generateZKProof = async (age, sapId) => {
    const { proof, publicSignals } =
    await snarkjs.groth16.fullProve(
        { age: age, sapID: sapId },
        "circuit.wasm",
        "circuit.zkey"
    );
    return { proof, publicSignals };
};
```

A.1.3 Proof Verification

```
// Verify Proof
const verifyProof = async (proofData) => {
    const { proof, publicSignals } = proofData;
    const vKey = JSON.parse(
        await fs.promises.readFile(
            'verification_key.json',
            'utf-8'
        )
    );
    return snarkjs.groth16.verify(
        vKey,
        publicSignals,
        proof
    );
};
```

A.1.4 Setup Process

```
circom circuit.circom --r1cs --wasm --sym --c
snarkjs powersoftau new bn128 12 pot12_0000.ptau -v
snarkjs powersoftau contribute pot12_0000.ptau \
    pot12_0001.ptau --name="First contribution" -v
snarkjs powersoftau prepare phase2 pot12_0001.ptau \
    pot12_final.ptau -v
snarkjs groth16 setup circuit.r1cs pot12_final.ptau \
    circuit.zkey -v
snarkjs zkey export verificationkey circuit.zkey \
    verification_key.json -v
```

APPENDIX B

Appendix B

This appendix documents encountered errors.

B.1 System Errors

The principal errors surrounding the algorithm and its testing were not in its implementation (surprisingly) but the system. System-level errors primarily arose during compilation, package installation, and dependency linking, particularly on Linux-based systems.

- **'undefined reference to `<symbol>`'** — Occurs when required object files or libraries are not linked during compilation.
- **'command not found'** — Common when a binary or script is not on the system's PATH or not installed.
- **'permission denied'** — Triggered by insufficient privileges to access or execute a file, often mitigated by setting permissions or using `sudo`.
- **'ELF load error'** — Happens when attempting to execute a binary compiled for a different architecture.

B.2 Web Extension Errors

These errors were observed during the integration of the web extension with a native messaging host and backend server.

- **'Could not establish connection. Receiving end does not exist.'** — Indicates an incorrect runtime message target or a missing listener in the background script.
- **'Native host has exited.'** — Raised when the native messaging host closes unexpectedly or fails to respond in time.
- **'Error parsing native message'** — Occurs when the JSON message sent from the native app is malformed or not prefixed with the 4-byte length.

B.3 VM Errors

Errors encountered while deploying code on a remote virtual machine, typically Ubuntu-based.

- **‘Connection refused‘** — When SSH service is not running or the firewall blocks the port.
- **‘No space left on device‘** — Arises from insufficient storage, often in `/tmp`, `/var`, or user home directories.
- **‘Segmentation fault‘** — Occurs due to invalid memory access in a program compiled and executed on the VM.

B.4 C Errors

C programming errors were mainly encountered during development of low-level cryptographic interfaces and FFI bindings.

- **‘Segmentation fault (core dumped)‘** — A generic memory violation error, frequently due to dereferencing null or invalid pointers.
- **‘double free or corruption‘** — Raised when attempting to deallocate memory that has already been freed.
- **‘invalid conversion from ...‘** — Type mismatch errors, common when using void pointers or interfacing with external libraries.
- **‘undefined behavior due to buffer overflow‘** — Caused by writing beyond the allocated array bounds, especially in cryptographic buffer handling.

References

[1] NIST. Post-quantum cryptography standardization, round 3 report, 2022. (Cited in section 1.2.)

[2] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber: A cca-secure module-lattice-based kem, 2018. IACR Cryptology ePrint Archive, Report 2017/634. (Cited in sections 1.2 and 10.4.3.)

[3] Vadim Lyubashevsky, Thomas Prest, Gregor Seiler, and Peter Schwabe. Crystals-dilithium: Digital signatures from module-lattices, 2018. IACR Cryptology ePrint Archive, Report 2017/633. (Cited in sections 1.2 and 10.4.3.)

[4] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. (Cited in section 2.1.)

[5] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 84–93, 2005. (Cited in section 2.1.)

[6] National Institute of Standards and Technology. Post-quantum cryptography standardization: Round 3 results, 2022. (Cited in sections 2.1 and 2.2.)

[7] Daniel J. Bernstein, David Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Peter Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn. Sphincs: Practical stateless hash-based signatures, 2015. In Eurocrypt 2015. (Cited in section 2.1.)

[8] Daniel J. Bernstein and Tanja Lange. Classic mceliece, 2008. Submission to NIST Post-Quantum Cryptography Standardization Process. (Cited in section 2.1.)

[9] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *Applied Cryptography and Network Security*, pages 164–175. Springer, 2005. (Cited in section 2.1.)

[10] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *PQCrypto*, pages 19–34. Springer, 2011. (Cited in section 2.1.)

[11] National Institute of Standards and Technology. Post-quantum cryptography standardization, 2016. (Cited in section 2.2.)

[12] National Institute of Standards and Technology. Post-quantum cryptography standardization: Round 4 candidates, 2025. Accessed from official NIST website. (Cited in section 2.2.)

[13] Joppe W. D'Anvers, Frederik Vercauteren, Thomas Pöppelmann, and Marnix Van Beirendonck. Implementation and comparison of lattice-based signatures for embedded systems. *Journal of Cryptographic Engineering*, 10:309–326, 2020. (Cited in section 2.3.)

[14] IDEMIA. Cryptographic agility in the post-quantum era, 2023. (Cited in section 2.3.)

[15] Cloudflare. Introducing hybrid post-quantum tls with kyber and x25519, 2025. (Cited in section 2.3.)

[16] Google Inc. Experimenting with post-quantum cryptography, 2019. (Cited in section 2.3.)

[17] Cloudflare. Post-quantum crypto in tls now, 2020. (Cited in section 2.3.)

[18] Thales Group. Post-quantum cryptography: Standardization, threats and transition. *White Paper*, 2023. (Cited in section 2.3.)

[19] Kim Cameron. The laws of identity. In *Microsoft Corp.*, 2005. (Cited in section 2.4.)

[20] Alex Preukschat and Drummond Reed. *Self-Sovereign Identity: Decentralized Digital Identity and Verifiable Credentials*. Manning Publications, 2021. (Cited in section 2.4.)

[21] Phil Windley. Sovrin: A protocol and token for self-sovereign identity and decentralized trust, 2018. (Cited in section 2.4.)

[22] Maria Apostolaki, Linus Gasser, and Srdjan Capkun. Towards secure and privacy-preserving decentralized identity. *IEEE Security & Privacy*, 2022. (Cited in section 2.4.)

[23] John Friedman and Nick Weaver. Post-quantum digital signatures: Challenges in mobile systems, 2021. (Cited in section 2.5.)

[24] Nick Bindel, Johannes Brendel, and Marc Fischlin. Hybrid signatures with tight security in the multi-user setting. *Designs, Codes and Cryptography*, 2021. (Cited in section 2.5.)

[25] Liren Ren, Yihang Zhang, Xiao Wang, and Jin Ran. Long-term security and pqc migration in identity management systems. *Future Generation Computer Systems*, 137:157–171, 2023. (Cited in section 2.5.)

[26] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Succinct non-interactive zero knowledge for a von neumann architecture. *USENIX Security Symposium*, 2014. (Cited in section 2.5.)

[27] Qiang Tang, Jiaxin Yu, and Ying Liu. A bibliometric analysis of post-quantum cryptography research. *Cryptography*, 6(3), 2022. (Cited in section 2.5.)

[28] A. Ott, A. Szepieniec, and E. Verheul. Quantum-secure identity-based encryption and digital identity systems. *Journal of Cryptology*, 32(2):239–260, 2019. (Cited in sections 1 and 3.)

[29] J. Tan, Z. Huang, and H. Wei. Transitioning to quantum-secure digital identity systems: Challenges and opportunities. *Future Generation Computer Systems*, 129:495–512, 2022. (Cited in sections 1, 2 and 4.)

[30] Kelsey A. Jackson, Carl A. Miller, and Daochen Wang. Evaluating the security of crystals-dilithium in the quantum random oracle model, 2024. (Cited in section 10.2.)

Publications

Journals

1. "Digital Identity System using post-quantum cryptographic paradigms" *(Submitted to Journal of Cryptographic Engineering (Q3), Reviewed by IACR Communications in Cryptology (Q1))*

Conferences

1. "Multi-ID Zero Knowledge Proof Systems for Anonymous and Verified Complaints" *(Submitted to International Conference for Women in Innovation, Technology and Entrepreneurship, ICWITE, IEEE Bangalore Section)*

Thesis Report

ORIGINALITY REPORT

2%	2%	2%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.coursehero.com Internet Source	1%
2	Mohammad Hammoudeh, Abdullah T. Alessa, Amro M. Sherbeeni, Clinton M. Firth, Abdullah S. Alessa. "Quantum Computing - A Journey into the Next Frontier of Information and Communication Security", CRC Press, 2024 Publication	1%

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		

Journal of Cryptographic Engineering - Receipt of Manuscript 'Digital identity system...'

1 message

Journal of Cryptographic Engineering <hemkumar.dilli@springernature.com> Wed, May 7, 2025 at 11:30 AM
To: manaspatil0967@gmail.com

Ref: Submission ID ffd52050-1bb3-4fc9-a454-28199606c34f

Dear Dr Patil,

Thank you for submitting your manuscript to Journal of Cryptographic Engineering.

Your manuscript is now at our initial Technical Check stage, where we look for adherence to the journal's submission guidelines, including any relevant editorial and publishing policies. If there are any points that need to be addressed prior to progressing we will send you a detailed email. Otherwise, your manuscript will proceed into peer review.

You can check on the status of your submission at any time by using the link below and logging in with the account you created for this submission:

https://submission.springernature.com/submission-details/ffd52050-1bb3-4fc9-a454-28199606c34f?utm_source=submissions&utm_medium=email&utm_campaign=confirmation-email&journal_id=13389

Kind regards,

Editorial Assistant
Journal of Cryptographic Engineering

Journal of Cryptographic Engineering is a hybrid journal. This means when the journal accepts research for publication, the article may be published using either immediate gold open access or the subscription publishing route. For further information please visit https://www.springernature.com/gp/open-research/about/green-or-gold-routes-to-OA/hybrid-options

International Conference for Women in Innovation, Technology and Entrepreneurship(ICWITE 2025) : Submission (1617) has been created.

1 message

Microsoft CMT <noreply@msr-cmt.org> Tue, May 6, 2025 at 7:28 PM
To: manaspatil1404@gmail.com

Hello,

The following submission has been created.

Track Name: Information Security

Paper ID: 1617

Paper Title: Multi-ID Zero Knowledge Proof Systems for Anonymous and Verified Complaints

Abstract:
Zero-Knowledge Proofs (ZKPs) enable users to prove knowledge of certain information without disclosing the information itself. Multi-ID ZKP systems extend this concept, allowing individuals to submit anonymous yet verifiable complaints across various platforms while maintaining privacy and accountability. This paper explores the integration of ZKPs into complaint management systems, addressing the challenges of anonymity, verifiability, scalability, and computational efficiency. By reviewing existing literature on ZKP applications in authentication, identity management, and scalable systems, we identify key advancements and research gaps. This work aims to establish a foundation for implementing robust, privacy-preserving, and efficient complaint systems leveraging Multi-ID ZKP mechanisms.

Created on: Tue, 06 May 2025 13:58:03 GMT

Last Modified: Tue, 06 May 2025 13:58:03 GMT

Authors:
- manaspatil1404@gmail.com (Primary)
- anshshah301@gmail.com
- adityarepe433@gmail.com
- sohamrane1210@gmail.com
- narendra.shekokar@djsce.ac.in
- dipali.bhole@djsce.ac.in

Secondary Subject Areas: Not Entered

Submission Files:
Multi ID ZKP Paper (2).pdf (759 Kb, Tue, 06 May 2025 13:57:54 GMT)

Submission Questions Response: Not Entered

Thanks,
CMT team.