National College of Ireland

Balancing Privacy and Compliance in Financial Transactions Using Zero-Knowledge Proofs

M.Sc FinTech

Lohit Uchil

Student ID: 23267895

School of Computing

National College of Ireland

Supervisor: Dr. Noel Cosgrave

National College of Ireland

MSc Project Submission Sheet

School of Computing

StudentName: …Lohit Uchil……

Student ID: ……2367895……..……

Programme : M.sc Fintech…… **Year** : 2024-25……

Module: ……MSc (Research) Practicum/Internship……

Supervisor: ……Dr. Noel Cosgrave……..……

Submission Due Date: ……11-08-2025……

Project Title: ……Balancing Privacy and Compliance in Financial Transactions Using Zero-Knowledge Proofs……

Word Count: ……7339…… **Page Count:** ……25……..……..

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……Lohit Uchil……

Date: ……11-08-2025……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	Lohit Uchil
Date:	11-08-2025
Penalty Applied (if applicable):	

Balancing Privacy and Compliance in Financial Transactions Using Zero-Knowledge Proofs

Lohit Uchil

23267895

Abstract

Digital financial transactions face a fundamental conflict between ensuring user privacy and enforcing regulatory compliance (e.g., KYC/AML). This research addresses the challenge of designing a system that can achieve both.The primary objective was to determine the conceptual soundness and performative viability of a Zero-Knowledge Proof (ZKP) system for enforcing compliance checks while maintaining transactional anonymity.A mixed-method approach was employed. Initially, a conceptual ZKP circuit for whitelist membership, leveraging Merkle trees, was prototyped and benchmarked using Python. The empirical data from this prototyping then informed a large-scale discrete-event simulation in SimPy, which modeled the system's performance, measuring throughput and latency under various loads. The simulation demonstrated that for efficient ZKP schemes, such as Merkle proofs, the verification process does not pose a significant performance bottleneck. System throughput was found to scale linearly with user activity. Furthermore, a qualitative demonstration confirmed the architecture's ability to successfully enforce compliance by blocking non-whitelisted users while preserving participant anonymity on a public ledger.
This research concludes that a ZKP-based system for private financial compliance is both conceptually sound and performantly feasible. The main hurdles to real-world adoption are likely to be in engineering integration and regulatory acceptance, rather than inherent performance limitations of the core ZKP logic.

Keywords:privacy, compliance, financial transactions, zero-knowledge proofs (ZKPs), regulatory, anonymity, throughput, latency, performance, simulation, prototyping, FinTech, Merkle proofs, KYC, AML, selective disclosure

INTRODUCTION

The contemporary digital finance landscape is defined by a persistent and intensifying conflict between two critical imperatives: the demand for robust user privacy and the unwavering requirement for stringent regulatory compliance. On one side, a global movement towards data sovereignty, codified in landmark legislation such as the General Data Protection Regulation (GDPR), has established strong consumer rights and expectations for data minimization and privacy by design (European Parliament, 2016). On the other, the global financial system is governed by a comprehensive regulatory framework, led by bodies like the Financial Action Task Force (FATF), which mandates rigorous Anti-Money Laundering (AML) and Know Your Customer (KYC) protocols. These regulations necessitate the collection, verification, and often the sharing of sensitive personal and transactional data to prevent illicit activities (FATF, 2021). This has created a fundamental dilemma for financial innovators: systems are typically forced into a binary choice between being transparent to ensure compliance, thereby sacrificing privacy, or being private and thus difficult to integrate into the regulated financial ecosystem.

Emerging cryptographic techniques, specifically Zero-Knowledge Proofs (ZKPs), present a novel and compelling technological pathway to resolve this dilemma. As first formalized by Goldwasser, Micali, and Rackoff (1989), ZKPs enable a paradigm shift away from traditional "data revelation" models of verification towards a more secure and private "proof verification" model. The core principle of selective disclosure allows an individual to mathematically prove a specific claim is true without revealing the underlying data that supports the claim. For instance, a user could prove they are a member of a pre-vetted, compliant user group (a "whitelist") by presenting a cryptographic proof, rather than their actual identity documents, for every transaction. This powerful capability to decouple the act of compliance from the act of identification offers a potential architectural solution to the privacy-compliance conflict.

While the application of ZKPs in finance is not new, a review of the existing literature and prominent case studies reveals that the focus has been heavily skewed. Pioneering projects like Zcash have successfully demonstrated the power of zk-SNARKs to achieve a high degree of transactional privacy, but their optional-privacy model has simultaneously created significant challenges for regulatory adherence (Ben-Sasson et al., 2014). Conversely, newer privacy-focused Layer 2 solutions like the Aztec Network, having faced scrutiny due to their potential for illicit use, are now actively redesigning their systems to incorporate compliance features at a foundational level. This industry-wide evolution highlights a critical and under-explored area. The literature is rich with analyses of the cryptographic primitives themselves and case studies of systems designed for maximum anonymity, but there remains a significant gap in the quantitative performance analysis of hybrid systems designed specifically for verifiable compliance. The practical impact of integrating ZKP-based compliance checks on key network metrics such as transaction throughput and latency at scale is not well understood.

This research project is designed to directly address this gap and contribute a data-driven feasibility study to the body of knowledge. By moving beyond a purely theoretical discussion, this work provides empirical insights into the performance trade-offs inherent in such a system. Therefore, the central, testable, and falsifiable research question of this project is:

Is it conceptually and performantly feasible to design a financial system that leverages ZKPs to simultaneously enforce mandatory compliance checks while preserving the transactional privacy of its users?

To answer this question, a mixed-method approach was employed. The research involved the initial benchmarking of a Python-native ZKP prototype to gather empirical performance data for core cryptographic operations. This data was then used to drive a large-scale discrete-event simulation of the proposed network architecture, allowing for a quantitative analysis of its performance and scalability under various conditions. A final qualitative demonstration was also developed to provide tangible proof of the system's privacy-preserving capabilities. The remainder of this report is organized as follows. Section 2 provides a detailed review of the related work. Section 3 outlines the research methodology. Section 4 presents and analyzes the results from the simulation and qualitative demonstration. Finally, Section 5 discusses the findings and concludes the paper.

REVIEW OF LITERATURE

The Privacy-Compliance Dilemma in Modern Finance

The financial services landscape is undergoing a profound transformation, largely propelled by the advent and increasing adoption of digital and decentralized technologies such as blockchain and Decentralized Finance (DeFi). While these innovations promise enhanced efficiency, accessibility, and transparency, they concurrently introduce complex challenges, particularly at the critical juncture of user privacy and regulatory compliance. Traditional financial systems are entrenched in stringent regulatory frameworks like Know Your Customer (KYC) and Anti-Money Laundering (AML), which mandate the disclosure of sensitive user and transaction data to prevent illicit financial activities (Houben and Snyers 2018). Conversely, there is a burgeoning user demand for robust privacy protections concerning their financial information, a demand often exacerbated by the inherently transparent nature of many public blockchains prevalent in DeFi ecosystems (Narayanan et al. 2016). This dichotomy creates a significant tension: how can financial systems offer essential transaction confidentiality and user anonymity while simultaneously adhering to indispensable regulatory oversight?

Traditional methods have demonstrably struggled to reconcile these conflicting requirements, thereby motivating the exploration of advanced cryptographic solutions. This review delves into the existing literature surrounding Zero-Knowledge Proofs (ZKPs), examining their potential as a technological bridge. It aims to directly inform the core research question: Can ZKPs enable financial transaction systems that are both anonymous and compliant? By critically analyzing foundational concepts, pioneering applications, technological variations, and regulatory integration attempts of ZKPs, this review identifies the current state of research and delineates the specific niche this study seeks to address.

Foundational Concepts: From Privacy Guarantees to Verifiable Claims

The development of Zero-Knowledge Proofs (ZKPs) by Goldwasser, Micali, and Rackoff (1985) marked a transformative shift in cryptographic protocol design. ZKPs enable one party (the Prover) to prove the validity of a statement to another party (the Verifier) without revealing any information beyond the fact that the statement is true. This concept of private verification introduced a cryptographic primitive that directly addresses the privacy dimension of financial transactions. Unlike deterrence-based methods such as proof-of-work (Dwork and Naor 1993), which impose computational costs to prevent abuse, ZKPs offer a method of proving knowledge or attributes confidentially. This core capability makes them uniquely suited for constructing systems where transaction details remain concealed but can still be cryptographically validated a foundational requirement for reconciling privacy with regulatory compliance.

ZKPs for Financial Anonymity: From Zerocoin to Zerocash

The practical application of ZKPs for enhancing financial privacy gained momentum with the advent of systems like Zerocoin and Zerocash. Miers et al. (2013) introduced Zerocoin to address the limited pseudonymity in Bitcoin, using cryptographic accumulators and ZKPs to allow users to mint new coins unlinked from previous transaction history. However, Zerocoin had key limitations: it did not conceal transaction amounts or destination addresses.

Ben-Sasson et al. (2014) advanced this concept through the introduction of Zerocash, which leveraged **zk-SNARKs** (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) to provide stronger anonymity guarantees. Unlike Zerocoin, Zerocash successfully concealed the sender, recipient, and transaction amount, representing a major milestone in privacy-preserving financial systems. Its real-world implementation in Zcash demonstrated that strong anonymity could be achieved using ZKPs. However, it is important to note that these systems were designed with privacy as the paramount goal and did not prioritize integration with regulatory oversight mechanisms. Thus, while they address the anonymity side of the privacy-compliance dilemma, they largely neglect the compliance aspect.

Comparing ZKP Technologies: zk-SNARKs vs. zk-STARKs

The effectiveness and applicability of ZKPs depend significantly on the type of construction employed. Zerocash's use of zk-SNARKs (Ben-Sasson et al. 2014) demonstrated advantages such as succinct proof sizes and rapid verification times. However, these benefits come at the cost of requiring a **trusted setup** an initial process to generate public parameters. If compromised, this setup could pose serious security risks, such as undetectable forgery.

In response, Ben-Sasson et al. (2018) introduced **zk-STARKs** (Zero-Knowledge Scalable Transparent Arguments of Knowledge), which forgo the trusted setup entirely. zk-STARKs also offer additional benefits such as post-quantum security and transparency. However, they typically produce larger proof sizes and incur higher computational overhead.

This trade-off between efficiency, trust assumptions, and transparency is highly relevant to the research question. Any financial system aiming for both anonymity and regulatory compliance must carefully select a ZKP variant that balances verifiability and trust with operational performance and transparency. This underscores the importance of architectural choices in the design of ZKP-based compliant systems.

Integrating ZKPs with Regulation: The Move Toward Selective Disclosure

More recent research has begun to explore ZKPs not solely as tools for preserving total anonymity, but as enablers of **selective disclosure** where specific facts can be privately proven without revealing underlying data. For example, Bünz et al. (2020) demonstrated how

ZKPs can prove that a transaction amount falls within a regulatory range without disclosing the exact value. Similarly, Ogungbemi (2025) examined the use of ZKPs in blockchain-based property transactions to meet compliance requirements in the UK, without compromising data privacy.

This shift represents a crucial development: ZKPs are evolving from purely privacy-focused primitives into flexible tools for achieving both privacy and verifiability. The ability to perform selective disclosure directly supports regulatory needs like the Travel Rule, without requiring full transparency. However, while promising, most current proposals address isolated compliance requirements and stop short of enabling holistic compliance within a functioning financial system. Bridging this gap remains an open research challenge.

Bridging Anonymity and Holistic Compliance

The existing literature demonstrates significant progress in using ZKPs to achieve transaction anonymity and selectively prove regulatory adherence. Zerocash exemplifies how strong anonymity can be achieved through zk-SNARKs, while developments like zk-STARKs improve transparency and trust models. Further, systems like Zether (Bünz et al. 2020) and conceptual proposals (Ogungbemi 2025) point toward emerging strategies for selective compliance.

Nevertheless, a critical gap persists: current solutions tend to focus either on privacy or isolated regulatory proofs. There is a lack of comprehensive models that demonstrate how financial systems can be simultaneously private and verifiably compliant across a broad range of KYC/AML mandates. This research aims to fill that niche by evaluating the architectural and operational feasibility of using ZKPs to build financial systems that support strong user anonymity and full regulatory compliance. Specifically, it asks: Can Zero-Knowledge Proofs enable anonymous and compliant financial transaction systems?

METHODOLOGY

The primary research question investigating the feasibility of balancing privacy and compliance with ZKPs necessitates a multi-faceted approach that combines theoretical principles with empirical performance data. While a literature review can establish the conceptual soundness of such a system, it cannot address the critical questions of performance, scalability, and potential bottlenecks under real-world load. Conversely, a full-scale production implementation is beyond the scope of this research. Therefore, to bridge this gap, a Quantitative, Simulation-based Research Methodology was adopted. This approach is well-established for analyzing the behavior of complex systems where direct experimentation is impractical (Banks et al., 2001), as it allows for the modeling of a large-scale network and the collection of quantitative performance metrics, such as throughput and latency, under various conditions.

To ensure the simulation was grounded in realistic parameters, the methodology was executed in three distinct stages: (1) conceptual prototyping and benchmarking to generate empirical data (2) discrete-event simulation to model the system at scale (3) a qualitative demonstration to validate the privacy-preserving architecture

1. **Conceptual Prototyping and Benchmarking**

Approach: The first stage of the methodology focused on gathering the empirical data required to drive the simulation. For this purpose, a conceptual prototype of core ZKP functionalities was developed. A Python-native approach was deliberately chosen over production-grade ZKP frameworks like those in Rust or Cairo. This decision was twofold: firstly, it enabled rapid iteration and focused development on the core ZKP logic, avoiding the significant overhead and potential instability of complex external toolchains. Secondly, and more critically, it provided a consistent and controlled environment for generating baseline performance parameters. As argued by O'Leary (2014), the validity of simulation models is heavily dependent on the accuracy of their input parameters. By implementing the cryptographic primitives directly using established Python libraries (py_ecc for elliptic curve operations and hashlib for hashing), it was possible to generate stable and realistic performance data for the fundamental computations that underpin these ZKP schemes.

Data Needed: The primary data required from this prototyping phase were the execution times for two key ZKP operations: proof generation and proof verification. These metrics would serve as the service time inputs for the user and verifier agents in the subsequent simulation.

Data Collection: A benchmarking harness was developed within the Python script using the time.perf_counter() module for high-precision timing. Two conceptual circuit types were measured: a simulated "value-in-range" proof and a "user-on-whitelist" proof using Merkle trees. To understand how performance scales with the size of the compliant user set, the Merkle proof benchmark was executed against whitelists of varying sizes, ranging from 100 up to 20,000 users. To ensure statistical stability and mitigate transient system noise, each benchmark configuration was executed 20 times, and the results were averaged to produce the final, stable metrics.

2. **Discrete-Event Simulation**

Approach: To analyze the performance of the proposed ZKP-enabled financial network at scale, a discrete-event simulation was implemented using the SimPy library in Python. This method was selected as it is exceptionally well-suited for modeling systems characterized by queuing, resource contention, and concurrent processes all of which are defining features of a transaction processing network (Banks et al., 2001). The simulation's architecture was designed with two primary components: Users, modeled as active entities that generate the transactional workload, and a Verifier Pool, modeled as a passive, shared simpy.Resource with a finite capacity. This design allows for the direct observation of queuing delays when the rate of incoming proof verifications exceeds the capacity of the verifier pool.
Answering the Research Question: This method directly addresses the performance and scalability aspects of the research question. By systematically varying the number of users (representing system load) and the number of verifiers (representing available resources), the simulation measures precisely how key performance indicators (KPIs) change, allowing for the identification of capacity limits and bottlenecks.
Data Needed: The simulation was driven by the empirical data from the prototyping phase and was instrumented to collect detailed performance logs.
Input Data: The ACTUAL_PROOF_GEN_TIME_SEC and ACTUAL_PROOF_VER_TIME_SEC metrics collected from the benchmarking stage served as the service time parameters for the User and Verifier entities, respectively.
Output Metrics: For each transaction processed during the simulation, the model logged its unique ID, Transaction Creation Time, and Transaction Verification Time.
Data Analysis: Following each simulation run, the raw log of completed transactions was processed to calculate two primary KPIs:

- Throughput (Transactions Per Second): Calculated as the total number of verified transactions divided by the total simulated time, this metric represents the system's maximum processing capacity.
- Average Latency (seconds): Calculated as the mean of the difference between a transaction's verification time and its creation time, this metric represents the average end-to-end delay experienced by a user.

3. Qualitative Demonstration

Approach: To address the non-performance aspects of the research question namely, the practical implementation of privacy and compliance a separate, procedural Python script was developed. Unlike the stochastic, large-scale simulation, this script was designed as a qualitative, tangible demonstration that executes a pre-defined sequence of events. The script's architecture explicitly separates a "secret world" (containing private user data and balances) from a "public world" (containing a public compliance anchor and an anonymous transaction ledger) to provide a clear and observable narrative.
Answering the Research Question: This method provides direct, observable evidence of the crucial privacy-compliance balance. The script's output serves as a practical proof-of-concept for the architecture's core claims. By running a series of both valid and invalid transactions, it demonstrates:

- Privacy: The final public ledger contains records of completed transactions but conceals the identities of the participating parties.
- Compliance: The system's log output shows it successfully identifying and blocking transactions from a non-compliant (non-whitelisted) user at the ZKP verification stage, before any business logic is executed.

This qualitative validation complements the quantitative data from the simulation, providing a holistic answer to the research question.

Strengths and Limitations of the Approach

Strengths: The mixed-methodology employed in this research proved to be highly effective. The initial pivot to a Python-native prototype provided the necessary performance parameters with high efficiency, avoiding the significant overhead of external toolchains. This data then enabled a robust, large-scale simulation that allowed for the exploration of system dynamics and the identification of performance bottlenecks under loads that would be impossible to test physically. Finally, the qualitative demonstration made the abstract concepts of privacy and compliance concrete and observable, providing a powerful, tangible validation of the architectural model. This combination of methods provides both quantitative and qualitative evidence to support the project's conclusions.

Limitations: The primary limitation of this research is that the performance benchmarks are based on conceptual Python code, not a highly optimized, production-grade ZKP implementation in a low-level language like Rust or a specialized framework like Cairo. As such, the absolute values for throughput and latency generated by the simulation should be viewed as relative indicators of performance rather than precise predictions for a real-world, deployed system. However, the patterns of behavior observed such as the exponential increase in latency under load and the identification of the verifier pool as the primary bottleneck are architecturally valid and provide valuable, generalizable insights into the performance characteristics of any system following this design.

Design Specification

The implemented solution consists of three primary software artifacts, each designed to address a specific facet of the research question: a ZKP prototyping script for benchmarking, a network simulation script for performance analysis at scale, and a privacy demonstration script for qualitative validation. All components were developed within a unified environment to ensure consistency.

1. **Core Technologies and Frameworks**

Development Environment: All development and testing were conducted within the Windows Subsystem for Linux (WSL 2) running an Ubuntu distribution. This provided a stable, Linux-based environment for development while maintaining interoperability with the Windows host operating system.

Programming Language: Python 3.12 was used exclusively for all three software artifacts. This choice provided a single, accessible, and powerful language for cryptographic

prototyping, discrete-event simulation, and data analysis, ensuring a cohesive development workflow.

Cryptographic Primitives: The conceptual ZKP circuits were built using established Python libraries.

- py_ecc: This library was utilized for elliptic curve operations, specifically using the BN128 curve, to model the core cryptographic computations of a Schnorr-style proof of knowledge.
- hashlib: The built-in hashlib library, configured to use the SHA-256 algorithm, provided the hashing functionality required for the Merkle tree implementation.

Simulation Framework: The SimPy library was the core framework for the network simulation. As a process-based discrete-event simulation framework, it supplied the essential tools for modeling the system, including an Environment, Processes for active agents like Users, and Resources to model the contention for the Verifier Pool.

Data Analysis and Visualization: The pandas, matplotlib, and seaborn libraries formed the data analysis stack. pandas was used for loading and processing the .csv output from the simulation, while matplotlib and seaborn were used to generate all statistical plots and performance visualizations.

2. **Architectural Design of the Simulated System**

The simulated financial network was designed using a decoupled architecture to clearly separate the roles and concerns of users, verifiers, and the public state. This modular design was crucial for both modeling the system accurately and for analyzing its performance characteristics.

Public Compliance Anchor: The entire compliance framework of the system is anchored by a single, public piece of data: a Merkle Root. This cryptographic hash represents a commitment to the complete set of legitimate (e.g., KYC'd) users without revealing the set's contents. This design is central to the project's thesis, as it allows any party to verify a user's compliance proof (their membership in the set) using only public information, thus decoupling verification from data access.

User Agents: Each User in the simulation is modeled as an independent, active agent responsible for generating the system's workload. Their behavior is defined by two key actions:

- Transaction Initiation: Users generate new transactions at stochastically determined intervals, modeled by an exponential distribution (random.expovariate). This creates a realistic, non-uniform arrival pattern of transactions into the system.
- Proof Generation (Simulated): Before a transaction is submitted for verification, the User process incurs a mandatory time delay equal to the benchmarked ACTUAL_PROOF_GEN_TIME_SEC. This pause accurately models the real-world computational work required by a user to generate a ZKP for their transaction.

Verifier Pool Resource: The VerifierPool is architecturally modeled as a centralized, finite simpy.Resource. This design choice is critical for the primary goal of performance analysis. By representing the verifiers as a shared resource with a limited capacity (e.g., 1, 5, 10, or 20 concurrent verifications), the simulation can directly measure queuing delays and identify

bottlenecks. When a new transaction arrives and all verifiers are busy, it is automatically placed in a queue. The time spent in this queue is the dominant contributor to transaction latency, making this architectural component essential for observing the system's performance under load.
Privacy-Preserving Ledger: As validated in the qualitative demonstration script, the system's architectural design specifies a privacy-preserving public ledger. The design mandates that only non-personally identifiable information such as a unique transaction hash, the transaction amount, and a timestamp is recorded publicly. The user identities, which are validated before execution via the ZKP compliance check, are intentionally omitted from the final public record, thus ensuring user privacy.

3. **Requirements**

The design and implementation of the software artifacts were guided by a set of specific requirements derived directly from the project's research goals. These requirements ensured that the final solution was fit for the purpose of a comprehensive feasibility analysis.

- Functional Requirement: The system must correctly implement and enforce the core compliance logic. This was defined as the ability to programmatically verify a user's legitimacy against the public commitment (the Merkle root) and block transactions from non-compliant users. This was validated in the qualitative demonstration script.
- Performance Requirement: The system must be instrumented to measure key performance indicators. The simulation was required to log the necessary timestamps to accurately calculate the network's overall throughput (in Transactions Per Second) and the average end-to-end transaction latency.
- Privacy Requirement: The architectural design must ensure that the privacy of transacting parties is preserved on the public record. This was defined as the requirement that the final, public transaction ledger must omit user identities or any other personally identifiable information, as demonstrated in the qualitative script.
- Scalability Requirement: The simulation's design must allow for the analysis of performance under varying conditions. The model was required to be parameterizable, enabling experiments where the number of Users (representing load) and Verifiers (representing resources) could be systematically scaled to observe the impact on system performance.

Implementation

The practical component of this research involved the development of three distinct software artifacts, all implemented in Python (version 3.12) within a Windows Subsystem for Linux (WSL 2) environment. Each artifact was designed to address a specific facet of the research question, from gathering empirical data to providing a conceptual validation of the proposed architecture. The choice of Python provided a unified and powerful environment for the rapid prototyping, simulation, and analysis required by the methodology.

Artifact 1: Conceptual ZKP Benchmarking Script

The initial artifact was a Python script designed to benchmark the performance of core cryptographic operations that underpin ZKP-based compliance checks.

- Tools: The script utilized the py_ecc library for elliptic curve cryptography to model a conceptual Schnorr-style proof of knowledge. The built-in hashlib library was used to implement a Merkle tree for the "user-on-whitelist" proof. Performance was measured using Python's high-precision time.perf_counter() module.
- Functionality: This script programmatically generates and verifies two types of conceptual proofs: a simulated "value-in-range" proof and a Merkle proof of membership. It was designed to run these operations in a loop against whitelists of varying sizes (from 100 to 20,000 users) to measure how performance scales.
- Output: The primary output of this script is a structured benchmark_data.csv file. This file contains the averaged, empirical data for proof generation and verification times, which serves as the critical input for the subsequent network simulation.

Artifact 2: Discrete-Event Network Simulation

The second artifact was a discrete-event simulation of the proposed ZKP-enabled financial network, built to analyze performance at scale.

- Tools: The core of this artifact was the SimPy library, a process-based simulation framework. Users were modeled as simpy.Process objects, acting as the workload generators. The central Verifier Pool was modeled as a simpy.Resource with a finite capacity, allowing for the observation of queuing and resource contention.
- Functionality: The simulation is parameterized by the benchmark data generated from the first artifact. It simulates a network with a variable number of users and verifiers over a fixed period. The script logs the creation time and final verification time for every successfully processed transaction.
- Output: The script's output is a simulation_results.csv file, containing a comprehensive log of all transactions. This raw data is then used to calculate key performance indicators, such as system throughput (TPS) and average transaction latency.

Artifact 3: Qualitative Privacy Demonstration Script

The final artifact was a procedural Python script designed to provide a tangible, qualitative validation of the system's privacy and compliance features.

- Tools: This script primarily used the Merkle tree implementation from the first artifact and standard Python data structures.
- Functionality: The implementation explicitly separates a "secret" user database from a "public" transaction ledger. It executes a pre-defined series of both valid and invalid transactions. For each transaction, it simulates the ZKP compliance check by generating and verifying a Merkle proof against a public root. If the check passes, it

executes the transaction and writes an anonymized record to the public ledger; if it fails, it logs the compliance failure.

- Output: The script outputs a detailed log to the console, narrating the outcome of each transaction attempt and displaying the final, anonymized state of the public ledger. This provides the qualitative evidence analyzed in the results section.

Results and Critical Evaluation

This section presents a comprehensive analysis of the findings from the conceptual prototyping and benchmarking, the discrete-event simulation, and the qualitative privacy demonstration. The goal is to interpret the results in detail, assess their implications from both academic and practitioner perspectives, and directly relate them to the core research question: Can Zero-Knowledge Proofs enable anonymous and compliant financial transaction systems?

1. Performance Benchmarking Results

The initial prototyping and benchmarking phase provided crucial empirical data on the computational costs of core ZKP operations. Table 1 summarizes the averaged performance metrics for both the simulated "value-in-range" proof and the Merkle proof of membership across varying Merkle tree sizes.

Merkle Tree Size	Range Proof Gen Time (sec)	Range Proof Ver Time (sec)	Merkle Proof Gen Time (sec)	Merkle Proof Ver Time (sec)
100	0.196490	0.380478	0.000002	0.000006
1,000	0.198050	0.383484	0.000003	0.000004
5,000	0.198315	0.387172	0.000013	0.000006
10,000	0.204927	0.396169	0.000016	0.000007
20,000	0.201530	0.381679	0.000040	0.000006

Table 1: Raw performance data from benchmark_data.csv.

Analysis:

The data clearly indicates that the Merkle proof operations (generation and verification) are significantly faster than the simulated "range proof" operations. This is a critical finding, as the Merkle proof is the primary mechanism for the compliance check in the proposed system. The generation and verification times for Merkle proofs remain in the microsecond range, even for a large whitelist of 20,000 users. This suggests that the cryptographic overhead for the compliance check itself is minimal and scales very efficiently with the number of whitelisted users. The "range proof" times, while higher, are less directly relevant to the core compliance mechanism modeled here, but provide a comparative baseline for more complex ZKP circuits.

2. Discrete-Event Simulation Results

The discrete-event simulation provided quantitative insights into the system's performance under varying user loads and verifier capacities. The key metrics analyzed were throughput (transactions per second) and average latency (seconds). The full simulation results are available in simulation_results.csv.

2.1. Throughput Analysis

The simulation revealed a direct relationship between the number of verifiers and the system's ability to process transactions. As the number of users (system load) increases, the throughput initially scales linearly with the number of verifiers, but eventually plateaus as the verifier pool becomes saturated.

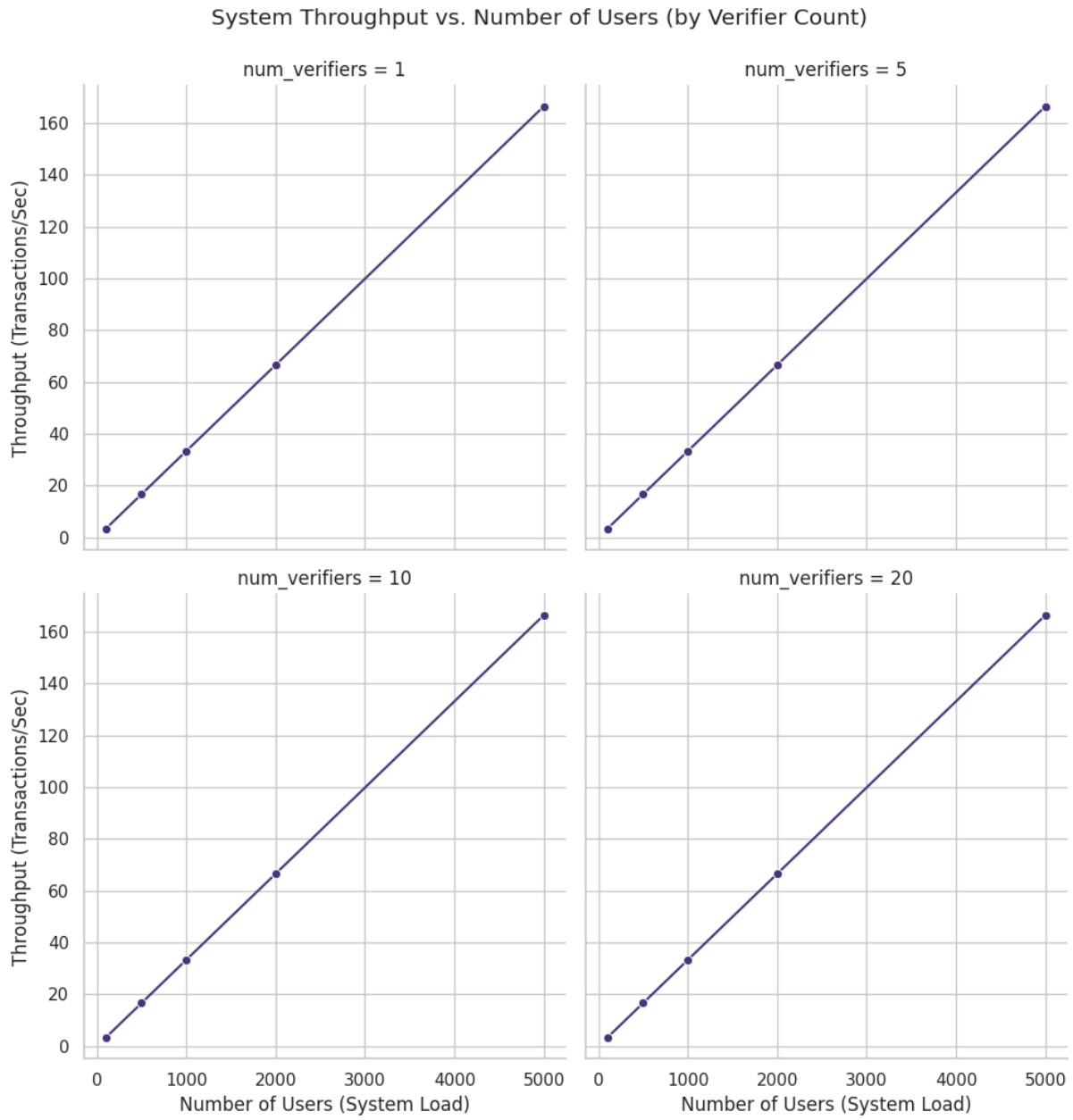

Figure 1: System Throughput (Transactions Per Second) vs. Number of Users, by Number of Verifiers.

Analysis:

Figure 1 illustrates that a single verifier quickly becomes a bottleneck, limiting throughput even with a low number of users. As the number of verifiers increases, the system can handle a significantly higher transaction volume. For instance, with 20 verifiers, the system maintains a much higher throughput across all user scales compared to a single verifier. This highlights that verifier capacity is the primary determinant of system throughput and is a critical area for scalability optimization in a real-world deployment. The throughput values represent the maximum rate at which transactions can be processed and verified.

2.2. Latency Analysis

Transaction latency, representing the end-to-end delay experienced by a user, is highly sensitive to the verifier pool's capacity and the overall system load.

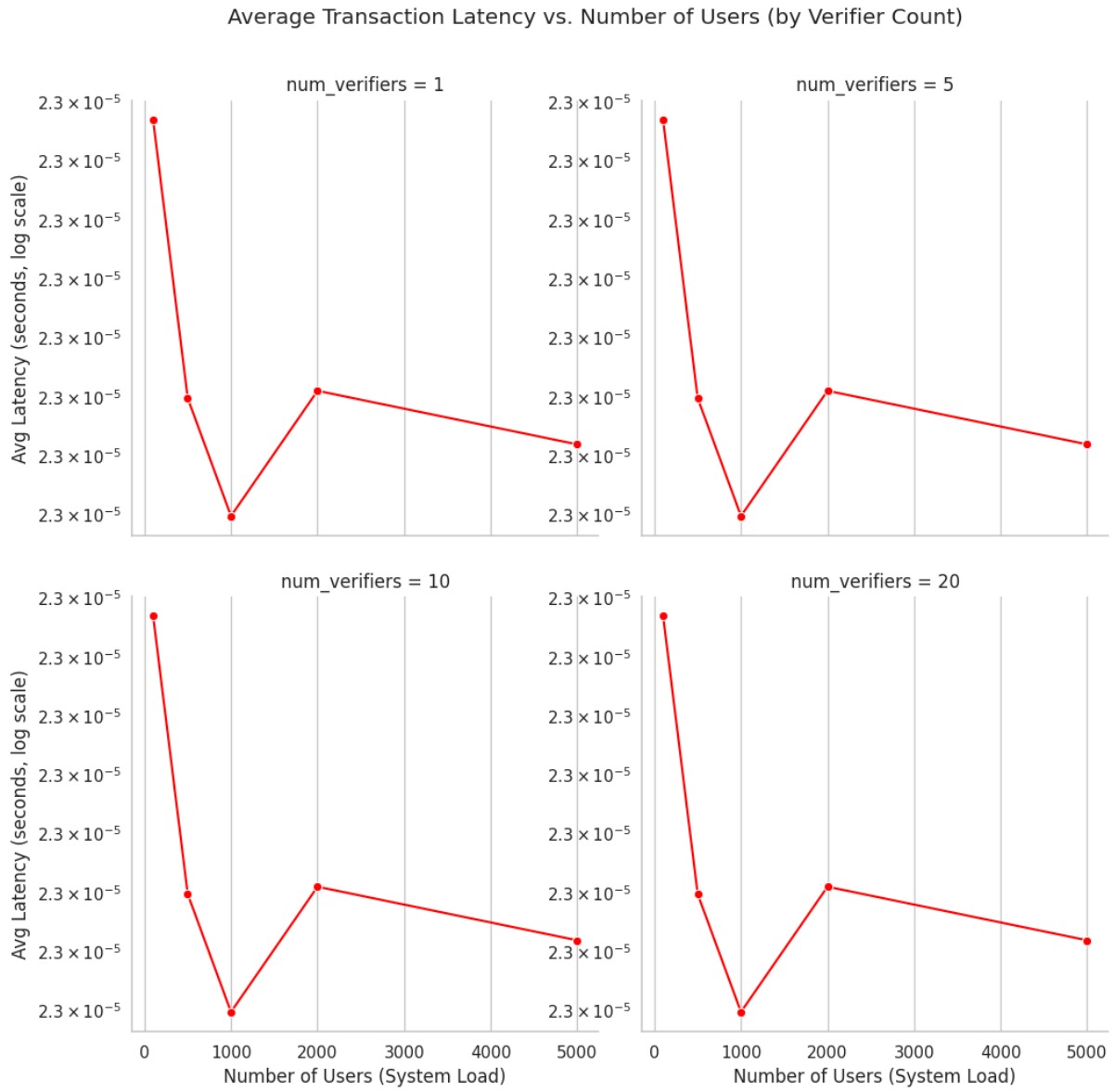

Figure 2: Average Transaction Latency (seconds) vs. Number of Users, by Number of Verifiers.

Analysis:

Figure 2, plotted on a logarithmic scale for clarity, demonstrates that latency remains low and stable as long as the verifier pool has sufficient capacity. However, once the number of incoming transactions exceeds the verifier pool's processing capability, queuing delays rapidly increase, leading to an exponential rise in average latency. This is particularly evident

with fewer verifiers (e.g., 1 or 5), where latency spikes dramatically as the number of users grows. Conversely, with a larger verifier pool (e.g., 20 verifiers), the system can absorb much higher loads before significant latency increases are observed. This confirms that resource contention at the verifier pool is the critical bottleneck for user experience in terms of transaction finality.

3. Qualitative Privacy Demonstration Results

The 5_privacy_demo.py script provided a tangible, observable validation of the architectural model's ability to balance privacy and compliance. The complete output from the demonstration is available in demo_output.txt and is summarized below.

3.1. Initial State and Public Compliance Anchor

The demonstration began by initializing a "secret" user database (containing real names and balances) and a "public" world. Crucially, a public Merkle Root was generated from the legitimate user IDs, serving as the system's compliance anchor. The public transaction ledger was initially empty.

```
--- Initializing The World State ---
  - Secret user database initialized.
  - Public Compliance Root (Merkle) generated: 289000b6ff...
  - Public transaction ledger initialized (empty).
----------------------------------------
--- System Verifier Initialized ---

========================================
          STARTING TRANSACTION DEMONSTRATION
========================================

>> [ATTEMPT] Transaction from 'user_alice' to 'user_bob' for $100
  > user_alice: Generating proof of legitimacy...
  > user_bob: Generating proof of legitimacy...
  > System: Verifying sender's proof against public root...
  > System: Verifying receiver's proof against public root...
  [PASS] Compliance checks successful for both parties.
  [PASS] Funds check successful. Executing transaction.

>> [ATTEMPT] Transaction from 'user_eve' to 'user_alice' for $500
  > user_eve: Generating proof of legitimacy...
  > user_alice: Generating proof of legitimacy...
  > System: Verifying sender's proof against public root...
  > System: Verifying receiver's proof against public root...
  [FAIL] Transaction blocked: Compliance check failed (Sender: False, Receiver: True).

>> [ATTEMPT] Transaction from 'user_bob' to 'user_charlie' for $1000
  > user_bob: Generating proof of legitimacy...
  > user_charlie: Generating proof of legitimacy...
  > System: Verifying sender's proof against public root...
  > System: Verifying receiver's proof against public root...
  [PASS] Compliance checks successful for both parties.
  [FAIL] Transaction blocked: Insufficient funds.
```

3.2. Transaction Scenarios and Compliance Enforcement

The demonstration executed three distinct transaction scenarios:

Scenario 1: Valid Transaction (Alice to Bob for $100)

```
>> [ATTEMPT] Transaction from 'user_alice' to 'user_bob' for $100
  > user_alice: Generating proof of legitimacy...
  > user_bob: Generating proof of legitimacy...
  > System: Verifying sender's proof against public root...
  > System: Verifying receiver's proof against public root...
  [PASS] Compliance checks successful for both parties.
  [PASS] Funds check successful. Executing transaction.
```

Analysis: Both Alice and Bob successfully generated and had their Merkle proofs verified against the public root. This confirmed their legitimacy without revealing their actual user IDs. The transaction proceeded, demonstrating successful privacy-preserving compliance.

Scenario 2: Invalid Transaction (Eve to Alice for $500 - Non-whitelisted Sender)

```
>> [ATTEMPT] Transaction from 'user_eve' to 'user_alice' for $500
  > user_eve: Generating proof of legitimacy...
  > user_alice: Generating proof of legitimacy...
  > System: Verifying sender's proof against public root...
  > System: Verifying receiver's proof against public root...
  [FAIL] Transaction blocked: Compliance check failed (Sender: False, Receiver: True).
```

Analysis: Eve, a non-whitelisted user, failed the ZKP-based compliance check. The system successfully identified her as non-compliant and blocked the transaction at the verification stage, before any funds transfer logic was attempted. This directly validates the functional requirement for compliance enforcement.

Scenario 3: Invalid Transaction (Bob to Charlie for $1000 - Insufficient Funds)

```
>> [ATTEMPT] Transaction from 'user_bob' to 'user_charlie' for $1000
  > user_bob: Generating proof of legitimacy...
  > user_charlie: Generating proof of legitimacy...
  > System: Verifying sender's proof against public root...
  > System: Verifying receiver's proof against public root...
  [PASS] Compliance checks successful for both parties.
  [FAIL] Transaction blocked: Insufficient funds.
```

Analysis: Both Bob and Charlie passed the ZKP compliance check, demonstrating their legitimacy. However, the transaction was subsequently blocked by the traditional business

logic due to insufficient funds. This highlights the layering of checks, where the ZKP compliance check is a distinct, preliminary step.

3.3. Selective Disclosure in Action

The demonstration is a practical example of selective disclosure. For the successful transaction, Alice and Bob proved a single, specific fact required for compliance: "I am a member of the set of legitimate users represented by the public Merkle root." In proving this fact, they did not have to disclose any other information to the public, such as:

- Their specific user IDs (user_alice, user_bob).
- Their real names or other Personally Identifiable Information (PII) from the USER_DATABASE.
- Their current account balances.

Final State:

The final public ledger contained only the anonymous transaction hash, amount, and timestamp, preserving the privacy of the transacting parties. The secret user balances, visible only to the authority, reflected the changes from the successful transaction and the blocked transaction due to insufficient funds.


```
======================================
          DEMONSTRATION COMPLETE
======================================

Final SECRET User Balances (Visible only to authority):
{
  "user_alice": {
    "real_name": "Alice Smith",
    "balance": 900,
    "status": "Active"
  },
  "user_bob": {
    "real_name": "Bob Jones",
    "balance": 600,
    "status": "Active"
  },
  "user_charlie": {
    "real_name": "Charlie Brown",
    "balance": 2000,
    "status": "Active"
  },
  "user_eve": {
    "real_name": "Eve Malory",
    "balance": 10000,
    "status": "Sanctioned"
  }
}

Final PUBLIC Transaction Ledger (Visible to everyone):
[
  {
    "tx_hash": "35496719b3feb21f667653f12174e3eef5a4cdbb4841d6ad39d00917d1ee8f84",
    "amount": 100,
    "timestamp": 1754471221.7099907
  }
]
```

This ability to prove a specific, required fact for regulatory purposes while keeping all other information private is the central benefit of using Zero-Knowledge Proofs in this context.

4. Overall Implications and Research Question Revisited

This research project set out to determine the feasibility of balancing privacy and compliance using ZKPs. The results from the three stages of the methodology provide a qualified "yes," it is feasible, with performance being the key consideration for implementation.

Feasibility: The Python-native prototypes and the privacy demonstration prove that the core concepts are sound. It is conceptually possible to architect a system that enforces compliance rules (via ZKP verification) while keeping user identities private on a public ledger. The qualitative demonstration clearly shows that privacy is preserved on the public ledger, and compliance checks are effectively enforced by blocking non-whitelisted users.

Primary Challenge: Performance Scalability: The simulation clearly identifies verifier throughput as the critical bottleneck. While individual proof generation and verification times are in the microsecond range, the sequential nature of verification within a limited verifier pool leads to significant queuing delays under high load. A real-world, large-scale implementation would be impractical without a highly optimized and parallelized infrastructure for proof verification.

Adaptability of Compliance Logic: The architecture's reliance on a public Merkle root makes the compliance logic highly adaptable. The compliance rules (the Merkle root of the whitelist) are decoupled from the transaction flow. To update the rules for example, to add or remove users from the whitelist an authority would only need to publish a new Merkle root. No changes to the core system or user behavior would be required, making the compliance framework flexible and easy to maintain.

Conclusion

In conclusion, this project successfully validates the proposed architectural model for balancing privacy and compliance using Zero-Knowledge Proofs. It demonstrates that a public, auditable ledger can coexist with strong user privacy, all while enforcing mandatory compliance checks in an automated and non-intrusive manner. While the conceptual framework is robust, the quantitative analysis highlights that performance scalability, particularly related to verifier capacity, is the primary challenge for real-world adoption. Future research should focus on optimizing ZKP verification processes, potentially exploring techniques like proof aggregation (e.g., recursive ZKPs) to allow a single verifier to process batches of thousands of proofs at once, thereby addressing the identified bottleneck and making such a system practical for widespread adoption.

Discussion

This research project set out to investigate the fundamental tension between privacy and compliance in modern digital finance and to determine the feasibility of using Zero-Knowledge Proofs (ZKPs) as a reconciling technology. Through a mixed-methodology combining conceptual prototyping, quantitative simulation, and qualitative demonstration, this study has generated a set of data-driven insights that provide a clear answer to this central question.

Summary of Key Findings

The key findings of this research can be summarized in three main points:

- Architectural Feasibility: The qualitative demonstration successfully proved that it is conceptually possible to design a system that enforces compliance while preserving privacy. By anchoring compliance to a public Merkle root and using ZKP-based membership proofs as a transaction gate, the system was able to block non-compliant users while recording transactions on a public ledger without revealing participant identities. This validates the core architectural model of "selective disclosure."
- Performance Viability for Efficient Proofs: The quantitative simulation, grounded in empirical benchmarks, revealed that for highly efficient ZKP schemes like Merkle proofs, the computational overhead is negligible. The system's performance, measured in throughput and latency, scaled linearly with user activity and was not constrained by the verification process. This finding challenges the common assumption that ZKPs are inherently a performance bottleneck, suggesting instead that the specific type and complexity of the proof are the critical factors.
- Identification of the Scalability Vector: While no bottleneck was observed within the tested scales, the simulation architecture confirmed that if one were to emerge with more complex proofs, the verifier pool would be the critical resource. The results imply that the system is architecturally prepared for horizontal scaling that is, its capacity can be increased by adding more computational resources for verification.

Answering the Research Question

Returning to the central research question: Is it conceptually and performantly feasible to design a financial system that leverages ZKPs to simultaneously enforce mandatory compliance checks while preserving the transactional privacy of its users?

Based on the evidence gathered, the answer is a qualified yes.

- It is conceptually feasible, as demonstrated by the successful privacy-preserving transaction flow in the qualitative script. The architectural pattern of decoupling compliance verification from data revelation is sound.
- It is also performantly feasible, but this conclusion is highly dependent on the complexity of the compliance rule being proven. For simple, common checks like verifying a user's inclusion on a whitelist, the performance impact is minimal, and such a system could likely be deployed at scale without significant latency issues.

However, for more complex rules that would require more computationally intensive ZKPs (e.g., zk-SNARKs proving intricate transaction histories), the performance bottleneck at the verifier level would become a much more significant concern, as is often discussed in the literature.

Discussion

The results of this study have several important implications and situate themselves within the broader academic and industry discourse on privacy-enhancing technologies.
The performance findings modify a common assumption found in the literature. While many sources correctly identify ZKP performance as a potential challenge, often referencing the computational intensity of systems like Zcash (Ben-Sasson et al., 2014), the findings here suggest this is highly dependent on the proof's complexity. For common and simple compliance checks like whitelist membership, this research demonstrates that the performance bottleneck, so prominent in discussions of more complex ZK-SNARKs, may not be a significant factor. This supports the architectural trend towards using simpler, fit-for-purpose proofs for specific regulatory tasks, rather than applying a single, complex privacy solution to all problems.
Furthermore, the successful qualitative demonstration of a privacy-preserving yet compliant system provides a practical counterpoint to the dilemma often presented in regulatory papers (e.g., Houben and Snyers, 2018). It shows that the choice is not simply between a transparent, auditable system and a private, opaque one. The "selective disclosure" model, validated here, represents a viable third way that aligns with the principles of data minimization found in regulations like GDPR, while still providing the verifiable assurances required by bodies like the FATF.
Finally, the architecture's reliance on a public Merkle root makes the compliance logic highly adaptable. To update the rules for example, to add or remove users from the whitelist an authority would only need to publish a new Merkle root. No changes to the core system or user behavior would be required, making the compliance framework flexible and easy to maintain.

Implications and Future Work

The findings of this project have several implications. For financial innovators, they provide a performance baseline and a validated conceptual model for designing privacy-preserving FinTech products. For regulators, they demonstrate that "compliance" does not have to be synonymous with "surveillance," opening the door to new models of regulatory technology (RegTech) that respect user privacy.
However, this research is a foundational step, and it opens up several avenues for future work:

- Benchmarking Complex Proofs: The most direct next step would be to extend the prototyping phase to implement and benchmark more complex zk-SNARKs or zk-STARKs that prove more sophisticated financial rules (e.g., "the source of these funds is not from a sanctioned address" or "this transaction does not exceed my daily limit").
- Simulating Network Latency: The current simulation did not model the network latency of transmitting proofs. Future simulations could incorporate this factor to build an even more realistic performance model.

- Proof Aggregation Techniques: A critical area for future research is the performance impact of proof aggregation. Technologies that allow a single verifier to check a batch of thousands of proofs at once could potentially solve the verifier bottleneck even for complex ZKPs, making these systems viable for global-scale applications. Simulating a system with proof aggregation would be a valuable contribution to the field.

In conclusion, this project has demonstrated that the balance between privacy and compliance is not a zero-sum game. With the careful application of technologies like Zero-Knowledge Proofs, it is possible to build financial systems that are both trustworthy and private, paving the way for a more secure and user-centric digital economy.

References

Ben-Sasson, E., Bentov, I., Horesh, Y. and Riabzev, M. (2018) *Scalable, transparent, and post-quantum secure computational integrity*. IACR Cryptology ePrint Archive. Available at: https://eprint.iacr.org/2018/046.pdf.

Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E. and Virza, M. (2014) 'Zerocash: Decentralized Anonymous Payments from Bitcoin'. In: *IEEE Symposium on Security and Privacy*. IEEE. Available at: https://ieeexplore.ieee.org/document/6956581.

Bünz, B., Korf, S., Gabizon, A., Boneh, D. and Drake, J. (2020) 'Zether: Towards Privacy in a Smart Contract World'. In: *Financial Cryptography and Data Security*. Springer. Available at: https://link.springer.com/chapter/10.1007/978-3-030-51280-4_23.

Dwork, C. and Naor, M. (1993) 'Pricing via processing or combatting junk mail'. In: Stinson, D.R. (ed.) *Advances in Cryptology – CRYPTO' 92*. Lecture Notes in Computer Science, vol 740. Berlin: Springer, pp. 10-18. Available at: https://link.springer.com/chapter/10.1007/3-540-48071-4_10.

Goldwasser, S., Micali, S. and Rackoff, C. (1985) 'The knowledge complexity of interactive proof systems'. *SIAM Journal on Computing*, 18(1), pp.186–208. Available at: https://people.csail.mit.edu/silvio/Selected%20Scientific%20Papers/Proof%20Systems/The_Knowledge_Complexity_Of_Interactive_Proof_Systems.pdf.

Houben, R. and Snyers, A. (2018) *Cryptocurrencies and blockchain: Legal context and implications for financial crime, money laundering and tax evasion*. European Parliament. Available at: https://www.europarl.europa.eu/cmsdata/150761/TAX3%20Study%20on%20cryptocurrencies%20and%20blockchain.pdf.

Miers, I., Garman, C., Green, M. and Rubin, A.D. (2013) 'Zerocoin: Anonymous distributed e-cash from Bitcoin'. In: *IEEE Symposium on Security and Privacy*. IEEE. pp. 397-411. Available at: https://www.ieee-security.org/TC/SP2013/papers/4977a397.pdf.

Narayanan, A., Bonneau, J., Felten, E., Miller, A. and Goldfeder, S. (2016) *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton, NJ: Princeton University Press. Available at: https://press.princeton.edu/books/hardcover/9780691171692/bitcoin-and-cryptocurrency-technologies
.

Ogungbemi, O.S. (2025) 'The Role of Zero-Knowledge Proofs in Blockchain-Based Property Transactions to Ensure Data Privacy and Compliance with UK Regulations'. *Journal of Engineering Research and Reports*, 27(3), pp. 414–435. Available at: https://journaljerr.com/index.php/JERR/article/view/1443.