Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

QR-HBDA: A Quantum-Resistant Hybrid Blockchain–Database Auditing Framework with Forensic Recovery and Trigger-Based Integrity Verification

LOTA INGTIPI

Lotaingtipi2021@gmail.com

Guahati University

Chandan Kalita

Guahati University

Research Article

Keywords: Blockchain Auditing, Database Security, Post-Quantum Cryptography, Forensic Recovery, Quantum-Resistant Hashing, Insider Threat Detection

Posted Date: June 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9990813/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Cluster Computing

ISSN: 1386-7857

QR-HBDA: A Quantum-Resistant Hybrid Blockchain–Database Auditing Framework with Forensic Recovery and Trigger-Based Integrity Verification

Lota Ingtipi

Chandan Kalita

Department of Information Technology, Gauhati University
Gopinath Bordoloi Nagar, Jalukbari, Guwahati 781014, Assam, India

Corresponding Author: Lota Ingtipi
E-mail: lotaingtipi2021@gmail.com
Telephone: +91-8135047742

Received: date / Accepted: date

Abstract

This paper presents QR-HBDA, a Quantum-Resistant Hybrid Blockchain–Database Auditing Framework that addresses insider threats and quantum computing risks to database integrity. The framework integrates MySQL database operations with an immutable, quantum-resistant blockchain audit ledger and an automated forensic recovery engine. Three principal innovations are introduced: (1) a Dynamic Trigger Factory generating nine MySQL triggers across three monitored tables via template-driven code generation; (2) a Quantum-Resistant Blockchain employing a dual-hash model combining SHA-256 (256-bit) and SHAKE-256 (512-bit, SHA-3 family); and (3) a Forensic Recovery Engine implementing a four-phase pipeline achieving 100% forensic recovery and 100% tamper detection in simulated insider attack scenarios. Experimental evaluation demonstrates sub-millisecond block creation, O(n) chain verification, and negligible storage overhead, confirming practical viability for academic records management.

Keywords: Blockchain Auditing · Database Security · Post-Quantum Cryptography · Forensic Recovery · Quantum-Resistant Hashing · Insider Threat Detection

1 Introduction

The digitalisation of academic institutions has fundamentally transformed how universities manage critical data pertaining to students, faculty, courses, examinations, and administrative operations. Relational database management systems (RDBMS) serve as the backbone of these information systems, providing efficient mechanisms for structured data storage, retrieval, and manipulation. However, a profound and frequently underappreciated vulnerability persists within these architectures: the very administrators entrusted with maintaining these systems possess sufficient privileges to alter data and destroy audit trails,

rendering insider threats one of the most significant risks to institutional data integrity in the modern threat landscape [1, 24].

A database administrator (DBA) with root-level MySQL access can execute arbitrary Data Manipulation Language (DML) statements—including DELETE, UPDATE, and even DROP TABLE—to silently modify, delete, or fabricate records without leaving any detectable trace in the system itself. This category of threat is classified as an insider threat and represents one of the most damaging and difficult-to-detect forms of cyberattack. According to the Verizon 2023 Data Breach Investigations Report, insider threats account for approximately 19% of all data breaches, with the average cost of an insider-driven breach significantly exceeding that of externally-initiated attacks due to the elevated access privileges involved and the extended dwell time before detection [24]. The Ponemon Institute further reports that the average cost of an insider-driven incident exceeds $15 million annually for large organisations, underscoring the economic imperative for robust insider threat mitigation.

Traditional database audit logging mechanisms, while helpful, are fundamentally self-defeating in the face of a motivated privileged attacker. The same administrator who deletes a student record can simultaneously delete the audit log entry that would have recorded the deletion, creating a single point of trust failure wherein the database itself serves as both the data store and the audit authority, with no independent verification mechanism [18, 19]. Schneier [19] articulated the security principle of separation of duties, arguing that no single individual or process should possess both the ability to perform an action and the ability to conceal it. This principle is precisely what the QR-HBDA framework operationalises.

Simultaneously, the rapid advancement of quantum computing presents a longer-term but equally serious threat to the cryptographic foundations of current security systems. Shor's algorithm [4] has demonstrated theoretically that a sufficiently powerful quantum computer could break RSA and Elliptic Curve Cryptography in polynomial time, rendering much of

current public-key infrastructure obsolete. While hash functions like SHA-256 are considered more quantum-resistant—requiring only a quadratic speedup via Grover's algorithm [7]—the strategic adoption of quantum-resistant cryptographic primitives such as SHAKE-256 from the SHA-3 family represents a prudent forward-security measure [6, 8]. The National Institute of Standards and Technology (NIST) completed its post-quantum cryptography standardisation process in 2022 [5], selecting CRYSTALS-Dilithium as one of three post-quantum signature standards, signalling the urgency of transitioning critical infrastructure to quantum-resistant alternatives.

The need for blockchain-assisted auditing has emerged as a natural response to these dual challenges. Blockchain technology, introduced by Nakamoto's seminal 2008 paper [1], provides an immutable, append-only data structure cryptographically linked through hash chaining. The application of blockchain for data integrity in non-financial domains has been explored by researchers including Zyskind et al. [10], Yue et al. [11], and Khatri and Hasan [16], demonstrating that blockchain's immutability can serve as an independent audit authority for database operations. However, existing blockchain-database auditing systems exhibit significant gaps: they lack automated forensic recovery, employ only classical hash functions, operate on single tables, and rarely target academic data management.

The need for forensic recovery capabilities in database systems has been increasingly recognised in the digital forensics community. Quick et al. [23] surveyed database forensics methodologies, identifying the challenge of reconstructing deleted records as a significant gap. While blockchain-based systems provide tamper detection, no prior work in the reviewed literature simultaneously implements automated discrepancy detection, forensic proof generation, and autonomous database restoration from blockchain payloads within a single integrated framework. This constitutes the primary motivation for the Forensic Recovery Engine component of the QR-HBDA framework.

Against this background, the Quantum-Resistant Hybrid Blockchain–Database Auditing (QR-HBDA) Framework is proposed as a comprehensive solution that directly addresses both insider threats and emerging quantum computing risks. The framework integrates a production-grade MySQL database with a Python-implemented, quantum-resistant blockchain that serves as an immutable, independently verifiable audit ledger, complemented by a fully automated forensic recovery engine capable of restoring deleted records even after complete erasure of both the target record and its audit trail.

1.1 Problem Statement

Academic institutions store and manage large volumes of sensitive data, including student academic records (grades, GPA, enrollment status), faculty employment and performance data, and course scheduling information. The integrity of this data is critical for academic fairness, regulatory compliance with frameworks such as FERPA [34] in the United States, institutional reputation, and legal accountability. Existing systems suffer from five critical limitations: (P1) a single point of trust failure where all data, including audit logs, resides within the same database; (P2) no forensic recovery mechanism once a record and its audit trail are deleted; (P3) classical hash vulnerabilities in systems relying solely on SHA-256; (P4) manual audit processes relying on periodic rather than real-time monitoring; and (P5) lack of unified multi-table auditing across all institutional data entities.

1.2 Research Motivations and Objectives

The primary objectives of this research are sevenfold: (O1) to design and implement a quantum-resistant blockchain serving as an immutable audit ledger for academic database operations; (O2) to develop a dynamic trigger factory capable of automatically generating MySQL INSERT, UPDATE, and DELETE triggers for any configured table; (O3) to build a blockchain anchoring engine that periodically synchronises database audit logs with the

blockchain; (O4) to implement a forensic recovery engine capable of detecting discrepancies, generating forensic proof reports, and autonomously restoring deleted data; (O5) to create an interactive CRUD dashboard ensuring all operations are transparently audited; (O6) to evaluate the system through a simulated insider attack demonstrating 100% recovery after complete database and audit trail erasure; and (O7) to validate the performance of blockchain operations and confirm practical viability.

1.3 Research Contributions

This paper makes the following seven contributions, labelled C1–C7:

C1: Dynamic Trigger Factory. A programmatic, template-driven trigger factory that deploys nine MySQL triggers across three tables in a single function call using Python f-string code generation, providing scalable multi-table audit logging without manual trigger authoring.

C2: Quantum-Resistant Dual-Hash Blockchain. A QuantumResistantBlockchain implementation employing both SHA-256 (classical, 256-bit) and SHAKE-256 (quantum-resistant, 512-bit XOF from the SHA-3 family) for block hashing, providing enhanced security against both classical and post-quantum threats.

C3: Automated Blockchain Anchoring. An atomic synchronisation mechanism between the database audit log and the blockchain, ensuring every database event is preserved immutably with verified chain integrity.

C4: Forensic Recovery Engine (FRE). A four-phase pipeline (scan → prove → extract → restore) that autonomously detects discrepancies between the live database and blockchain state, generates standardised forensic proof reports, and restores deleted records with 100% accuracy.

C5: Multi-Table Audit Architecture. A unified, centralised global_audit_log table receiving triggered events from all monitored tables, enabling comprehensive cross-table auditing with a single audit authority.

C6: Interactive Management Interface. A six-tab ipywidgets-based graphical dashboard for accessible data management, ensuring all user interactions are transparently logged through the trigger system.

C7: Insider Attack Recovery Validation. Empirical demonstration through rigorous threat simulation that 100% forensic detection and 100% database restoration are achievable even after complete erasure of the database record and its audit trail by a malicious administrator.

1.4 Paper Organisation

The remainder of this paper is organised as follows. Section 2 reviews related work across blockchain auditing, database security, post-quantum cryptography, and forensic recovery. Section 3 presents a systematic research gap analysis. Section 4 describes the system architecture. Section 5 details the methodology with formal mathematical models. Section 6 documents the implementation. Section 7 describes the experimental setup. Section 8 presents results and discussion. Section 9 provides a security analysis. Sections 10 through 13 address compliance, limitations, future work, and conclusions.

2 Related Work

This section surveys the existing body of literature relevant to the core technical domains addressed by the QR-HBDA framework: blockchain technology, database audit logging, blockchain-database integration, quantum-resistant cryptography, forensic data recovery, and insider threat detection.

2.1 Blockchain Auditing Systems

Blockchain technology was introduced through Nakamoto's seminal 2008 paper on Bitcoin [1], proposing a decentralised, peer-to-peer electronic cash system maintained by a chain of cryptographically linked blocks. Zheng et al. [2] provided a comprehensive architectural overview of blockchain systems, classifying them along the axes of consensus mechanism, data model, and network topology. Swan [3] articulated the broader potential of blockchain beyond cryptocurrencies, identifying smart contracts, identity management, and institutional record-keeping as key application domains. For data integrity in non-financial domains, Zyskind et al. [10] proposed using blockchain for personal data management and access control, demonstrating that blockchain's immutability could be leveraged to build decentralised audit trails that no single entity could unilaterally alter. Yue et al. [11] explored healthcare data sharing using blockchain, highlighting how medical records could be stored and verified without relying on a central trusted authority.

2.2 Database Trigger Auditing

Database audit logging is a well-established practice in enterprise systems. Traditional approaches rely on database-native mechanisms such as Oracle Audit Vault, MySQL's general query log, or application-level logging frameworks. O'Neil et al. [18] observed that all logging mechanisms residing within the same trust boundary as the audited database are fundamentally vulnerable to a privileged insider. Bace [20] described trigger-based auditing as the gold standard for database-level activity capture because triggers fire at the RDBMS level, below the application tier, making them invisible to application-layer attacks. Litwin and Risch [21] examined the performance overhead of database triggers, finding that well-designed triggers impose manageable overhead even for high-frequency DML operations. However, triggers are themselves database objects sharing the same trust boundary as the

monitored data—precisely the vulnerability addressed by the QR-HBDA framework's blockchain anchoring mechanism.

2.3 Blockchain–Database Integration

The integration of relational databases with blockchain for audit purposes has attracted significant research attention. Peng et al. [15] proposed a blockchain-anchored database auditing approach using Merkle tree structures for efficient verification of large datasets. Crosby and Wallach [22] introduced the concept of authenticated data structures allowing efficient membership and non-membership proofs. Fan et al. [14] extended blockchain-based auditing to multi-party database scenarios. While these works validated the feasibility of blockchain-database integration, none implemented automated forensic recovery from blockchain payloads, nor did they employ quantum-resistant hashing.

2.4 Academic Blockchain Systems

Grech and Camilleri [12] published an influential European Commission report on blockchain in education, surveying over 100 blockchain-in-education projects. Turkovic et al. [13] proposed EduCTX, a global decentralised higher education credit system built on the Ark blockchain providing tamper-proof storage of credit transfer data. However, neither system addressed forensic recovery from database tampering or employed quantum-resistant cryptographic primitives. The academic domain remains underserved by existing blockchain-database auditing research, which predominantly targets financial and healthcare contexts.

2.5 Post-Quantum Cryptography

Bernstein and Lange [7] provided a comprehensive survey of post-quantum cryptography, categorising candidate schemes into lattice-based, code-based, hash-based, and multivariate polynomial families. The SHA-3 family, standardised by NIST in FIPS PUB 202

[6], is based on the Keccak sponge construction and provides a fundamentally different mathematical foundation from the Merkle–Damgard-based SHA-2 family. Bernstein et al. [8] demonstrated that hash-based constructions, including SHAKE, are among the most conservative and well-understood candidates for post-quantum security. Ducas et al. [9] proposed CRYSTALS-Dilithium, a lattice-based digital signature scheme selected by NIST as a post-quantum signature standard in 2022 [5]. The integration of Dilithium signatures with blockchain-based auditing represents a promising but unexplored research direction.

2.6 Forensic Recovery Systems

Digital forensics in database contexts is relatively underexplored compared to filesystem and network forensics. Quick et al. [23] surveyed database forensics methodologies, identifying the challenge of reconstructing deleted records from database files as a significant gap. Khatri and Hasan [16] proposed a blockchain-based forensic audit system for cloud databases using smart contracts for immutable event logging, but did not implement automated data restoration. Agrawal et al. [17] demonstrated that blockchain audit trails could prove regulatory compliance in financial systems, establishing the legal admissibility of blockchain evidence. No prior work simultaneously performs automated discrepancy detection, forensic proof generation, and autonomous record restoration within a single integrated framework.

2.7 Insider Threat Mitigation

The CERT Insider Threat Center [24] defines an insider threat as a current or former employee, contractor, or business partner who intentionally exceeds or misuses authorised access. Anderson et al. [25] reviewed technical countermeasures for insider threats, categorising them as preventive, detective, and corrective. Maloof [26] explored machine learning approaches for insider threat detection based on behavioural patterns, an approach

complementary to blockchain-based evidence preservation. The integration of behavioural detection with blockchain-based forensic investigation represents a particularly robust defence architecture.

2.8 Comparative Analysis of Related Work

Table 1 presents a comparative analysis of the most directly relevant prior works against the principal features of the QR-HBDA framework.

Table 1 Comparative analysis of related work. Symbols: ✓ = fully implemented; ~ = partially addressed; × = not addressed.

Author	Year	BC	Audit	QR	FR	Multi	Acad.	Domain	Limitations
Nakamoto [1]	2008	✓	×	×	×	×	×	Crypto	No DB integration
Zyskind [10]	2015	✓	~	×	×	×	×	Privacy	No forensics
Yue et al. [11]	2016	✓	~	×	×	×	×	Health	No recovery
Grech [12]	2017	✓	~	×	×	×	✓	Education	No implementation
Turkanovic [13]	2018	✓	✓	×	×	×	✓	Education	No forensics
Fan et al. [14]	2018	✓	✓	×	×	~	×	Health	No QR hashing
Peng et al. [15]	2020	✓	✓	×	×	~	×	General	No forensics
Khatri [16]	2019	✓	✓	×	~	×	×	Cloud	No restoration
Agrawal [17]	2018	✓	~	×	×	×	×	Finance	No QR hashing
Crosby [22]	2009	✓	✓	×	×	×	×	General	Merkle tree only
QR-HBDA (Ours)	2026	✓	✓	✓	✓	✓	✓	Academic	Single-node

BC = Blockchain; Audit = Audit Logging; QR = Quantum Resistance; FR = Forensic Recovery; Multi = Multi-Table Support; Acad. = Academic Domain

3 Research Gap Analysis

Based on the systematic review presented in Section 2, five critical research gaps are identified that collectively define the design objectives distinguishing the QR-HBDA framework from the state of the art. These gaps are presented in Table 2.

Table 2 Identified research gaps and corresponding QR-HBDA contributions.

ID	Research Gap	QR-HBDA Contribution	Status
G1	Lack of integrated forensic recovery: no prior work simultaneously implements tamper detection, forensic proof generation, and automated database restoration from blockchain payloads	C4: Four-phase Forensic Recovery Engine with scan, prove, extract, and restore pipeline	Addressed
G2	No unified multi-table audit framework: most prior systems target a single data entity; dynamic trigger generation for multi-table environments remains unaddressed	C1, C5: Dynamic Trigger Factory with nine triggers across three tables feeding a single global_audit_log	Addressed
G3	Absence of quantum-resistant hashing in audit chains: existing blockchain audit systems rely on SHA-256 alone; the dual-hash SHA-256 + SHAKE-256 design has not been previously proposed	C2: Dual-hash QuantumResistantBlockchain with 256-bit classical and 512-bit post-quantum security	Addressed
G4	Academic domain specificity: the majority of blockchain-database auditing research targets financial or healthcare domains	C7: Full implementation validated on students, faculty, and courses tables with institutional compliance mapping	Addressed
G5	Lack of interactive management integration: no prior work integrates the audit chain with an in-notebook widget-driven CRUD interface	C6: Six-tab ipywidgets dashboard for transparent, audited data management	Addressed

The gap analysis reveals that while individual components of the QR-HBDA framework have been explored in isolation across the literature—blockchain for audit [15, 22], trigger-based logging [20, 21], quantum-resistant hashing [6, 7], and forensic recovery [16, 23]—no existing system integrates all of these capabilities within a single, unified

framework. The QR-HBDA framework specifically addresses all five identified gaps, representing the first holistic approach to quantum-resistant blockchain-assisted database auditing with automated forensic recovery.

4 System Architecture

4.1 Architectural Overview

The QR-HBDA framework is a hybrid system integrating two fundamentally different data storage and verification paradigms—a mutable relational database and an immutable append-only blockchain—to achieve tamper-proof audit logging with forensic recovery capabilities. The architecture is organised into seven distinct layers, each with clearly defined responsibilities and interfaces. Figure 1 illustrates the high-level system architecture.

[Figure Placeholder]

Figure 1 High-level system architecture of the QR-HBDA framework showing the seven-layer design: Layer 1 (User Layer) → Layer 2 (CRUD Dashboard) → Layer 3 (Application Layer) → Layer 4 (Trigger Factory) → Layer 5 (Audit Log) → Layer 6 (Blockchain Layer) → Layer 7 (Forensic Recovery).

1. User Layer: End users interact with the system through a graphical interface within the Jupyter/Colab environment.

2. CRUD Dashboard Layer: The ipywidgets dashboard provides a six-tab graphical interface for managing all data operations across students, faculty, and courses tables.

3. Application Layer: Python helper functions implement CRUD operations, sample data seeding, and audit log review using parameterised SQL queries.

4. Trigger Factory Layer: The dynamic trigger generation system deploys nine MySQL triggers (INSERT, UPDATE, DELETE × 3 tables) using template-driven code generation.

5. Audit Log Layer: The global_audit_log table serves as the centralised audit repository, capturing all DML events from monitored tables.

6. Blockchain Layer: The quantum-resistant blockchain provides immutable evidence storage with dual-hash verification (SHA-256 + SHAKE-256).

7. Forensic Recovery Layer: The Forensic Recovery Engine provides tamper detection, forensic proof generation, and automated database restoration from blockchain payloads.

4.2 Database Schema Design

The system manages four MySQL tables within the qr_hbda_v2 database. The students table stores core academic records with fields: student_id (AUTO_INCREMENT, PRIMARY KEY), first_name, last_name, major, gpa (DECIMAL), and enrollment_date (DATE). The faculty table stores employment information with faculty_id (PRIMARY KEY), name, email (UNIQUE), designation, department, and hire_date. The courses table stores course metadata with a foreign key relationship to the faculty table, using ON DELETE SET NULL for referential integrity. The global_audit_log table is the central audit repository with fields: log_id, table_name, action_type (ENUM), record_id, field_changed, old_value, new_value, operation_timestamp, anchored (BOOLEAN), anchored_at, block_hash, and quantum_hash.

4.3 Component Interaction Model

Table 3 summarises all major system components, their roles, and their interactions.

Table 3 System components and interactions.

Component	Role	Interacts With
DatabaseManager	MySQL connection management; schema initialisation; retry logic	MySQL, triggers
Dynamic Trigger Factory	Generates 9 MySQL triggers for all 3 tables	global_audit_log
global_audit_log	Central database audit repository	Triggers, Anchoring Engine
anchor_audits_to_blockchain	Synchronises DB audit log to	Audit log, Blockchain

	blockchain	
QuantumResistantBlockchain	Immutable evidence store with dual-hash	Anchoring Engine, FRE
QuantumResistantBlock	Individual block with SHA-256 + SHAKE-256	Blockchain
ForensicRecoveryEngine	Tamper detection, proof generation, restoration	Blockchain, MySQL
CRUD Helper Functions	Application-level CRUD operations	DatabaseManager
ipywidgets Dashboard	6-tab UI for interactive data management	CRUD helpers

5 Methodology

This section provides the detailed mathematical formulations and algorithmic descriptions underlying each component of the QR-HBDA framework.

5.1 Dynamic Trigger Generation Model

5.1.1 Formal Definition

Let $T = \{t_1, t_2, \ldots, t_n\}$ denote the set of tables to be audited. For each table $t_i \in T$, let K_i denote its primary key column and $C_i = \{c_{i,1}, c_{i,2}, \ldots, c_{i,mi}\}$ denote its set of monitored attribute columns. The trigger configuration is defined as the triple set:

$$C = \{(t_1, K_1, C_1), (t_2, K_2, C_2), \ldots, (t_n, K_n, C_n)\} \quad (1)$$

For each table t_i, the system generates three trigger definitions:

$$\Gamma(t_i) = \{\tau^{INS}_i, \tau^{UPD}_i, \tau^{DEL}_i\} \quad (2)$$

5.1.2 INSERT Trigger

For the INSERT trigger τ^{INS}_i, the complete new row is captured as a concatenated field–value string:

$$V^{new}_i = CONCAT(\bigcup_{j=1}^{m_i} IFNULL(NEW.c_{i,j}, 'NULL') \mid) \quad (3)$$

The trigger body inserts one row into the global_audit_log with action_type = 'INSERT', field_changed = 'ALL', old_value = NULL, and new_value = V^new_i.

5.1.3 UPDATE Trigger

The UPDATE trigger τ^UPD_i implements conditional, field-level change detection. For each monitored column $c_{i,j}$:

$$\Delta_{i,j} = IF(IFNULL(OLD.c_{i,j}, '') \neq IFNULL(NEW.c_{i,j}, ''), INSERT\ audit_entry, \emptyset) \quad (4)$$

A single UPDATE statement modifying k columns produces exactly k audit log entries, each identifying one field-level modification. If no monitored field changes, zero entries are generated.

5.1.4 DELETE Trigger

The DELETE trigger τ^DEL_i captures the complete deleted row:

$$V^{old}_i = CONCAT(\cup_{j=1}^{m_i} IFNULL(OLD.c_{i,j}, 'NULL') \mid) \quad (5)$$

5.1.5 Complexity Analysis

The time complexity of the trigger generation algorithm is:

$$O(n \times m) \quad (6)$$

where n = |T| is the number of tables and m = $\max_i |C_i|$ is the maximum number of monitored columns across all tables. In the current implementation, n = 3 and m = 6, yielding O(18) trigger definitions generated in negligible time.

Algorithm 1 Dynamic Trigger Generation from TABLE_CONFIGS

```
REQUIRE:  C = {(t₁, K₁, C₁), ..., (tₙ, Kₙ, Cₙ)}
ENSURE:   Set of 3n MySQL trigger DDL statements
FOR each (tᵢ, Kᵢ, Cᵢ) ∈ C DO
  Drop existing triggers: trg_{tᵢ}_insert, trg_{tᵢ}_update, trg_{tᵢ}_delete
  Generate INSERT trigger DDL:
    Build CONCAT expression from NEW row using Eq. (3)
    Template: AFTER INSERT ON tᵢ FOR EACH ROW
    Insert into global_audit_log with action_type = 'INSERT'
  Generate UPDATE trigger DDL:
```

```
    For each c_{i,j} ∈ C_i: Generate IF comparison block per Eq. (4)
  Generate DELETE trigger DDL:
    Build CONCAT expression from OLD row using Eq. (5)
    Template: AFTER DELETE ON t_i FOR EACH ROW
END FOR
RETURN ⋃_{i=1}^{n} Γ(t_i)
```

5.2 Blockchain Anchoring Algorithm

5.2.1 Block Structure Definition

Each block B_i in the blockchain is defined as:

$$B_i = \{I_i, T_i, P_i, H^{prev}_i, N_i, H^{SHA256}_i, H^{SHAKE256}_i, \sigma_i\} \quad (7)$$

where I_i is the sequential block index ($I_0 = 0$ for genesis), T_i is the ISO 8601 timestamp, P_i is the payload dictionary, H^{prev}_i is the hash of the previous block, N_i is the nonce, H^{SHA256}_i and $H^{SHAKE256}_i$ are the dual hash values, and σ_i is the signature algorithm identifier.

5.2.2 Canonical Block Representation

The canonical string representation of a block, which serves as the common input to both hash functions, is the JSON-serialised form with sorted keys:

$$B^{str}_i = JSON(\{I_i, T_i, P_i, H^{prev}_i, N_i\}, sorted_keys) \quad (8)$$

Algorithm 2 Blockchain Anchoring Algorithm

```
REQUIRE:  MySQL global_audit_log, QuantumResistantBlockchain instance
ENSURE:   List of new block indices; updated anchored flags
QUERY global_audit_log WHERE anchored = 0 ORDER BY log_id ASC
IF no rows found THEN
  RETURN empty list
END IF
FOR each row r in result set DO
  Construct payload dictionary P from r's fields
  B ← blockchain.add_block(P)      // Creates block with dual hash
  UPDATE global_audit_log SET anchored = 1,
         block_hash = B.hash, quantum_hash = B.quantum_hash
         WHERE log_id = r.log_id
  COMMIT transaction
END FOR
blockchain.verify_chain_integrity()    // Verify chain integrity
RETURN list of new block indices
```

The anchoring algorithm guarantees three properties: (i) Atomicity—the anchored flag is set only after the corresponding block is successfully added and the transaction committed; (ii) Idempotence—re-running on an already-anchored log produces no new blocks; and (iii) Ordering preservation—entries are anchored in chronological order.

5.3 Dual-Hash Model

5.3.1 Hash Computation

The dual-hash mechanism is the cryptographic core of the QR-HBDA framework. Each block stores two independent hash values computed from the same canonical block string using two cryptographically distinct hash functions:

$$H^{SHA256} = SHA\text{-}256(B^{str}) \quad (9)$$

$$H^{SHAKE256} = SHAKE\text{-}256(B^{str}, 512) \quad (10)$$

The combined block integrity measure is:

$$I(B) = H^{SHA256} \,||\, H^{SHAKE256} \quad (11)$$

where || denotes concatenation. This combined measure provides 64 + 128 = 192 hex characters (768 bits) of integrity verification per block.

5.3.2 Quantum Security Analysis

Grover's algorithm [7] provides a quadratic speedup for unstructured search, reducing the security of an n-bit hash function from $O(2^n)$ to $O(2^{n/2})$ under quantum attack. The security properties of each hash function under both classical and quantum threat models are summarised in Table 4.

Table 4 Classical and quantum security comparison of SHA-256 and SHAKE-256.

Property	SHA-256 (Classical)	SHA-256 (Quantum)	SHAKE-256 512-bit (Classical)	SHAKE-256 512-bit (Quantum)

Output length	256 bits	256 bits	512 bits	512 bits
Preimage resistance	2^256	2^128	2^512	2^256
Collision resistance	2^128	2^64	2^256	2^128
Algorithm family	SHA-2 (Merkle–Damgard)	—	SHA-3 (Keccak sponge)	—
Shor resistance	N/A	N/A	N/A	N/A
Grover resistance	—	2^128 (preimage)	—	2^256 (preimage)

SHA-256 provides 256 bits of classical preimage resistance and 128 bits under Grover's attack. SHAKE-256 with 512 bits of output provides 256 bits of classical preimage resistance and 128 bits under Grover's attack for collisions, but critically provides 256 bits of quantum security for preimage resistance—double that of SHA-256. The Keccak sponge construction underlying SHAKE-256 is fundamentally resistant to Shor's algorithm, as its security does not rely on integer factorisation or discrete logarithm problems. The dual-hash design provides a hybrid security model: SHA-256 maintains backward compatibility with existing tools and systems, while SHAKE-256 provides enhanced forward security against quantum threats.

5.4 Blockchain Integrity Verification

5.4.1 Chain-Linking Relationship

The chain-linking relationship establishes that the integrity of every block depends on all preceding blocks:

$$H^{prev}_i = H^{SHA256}_{i-1}, \quad \forall i > 0 \quad (12)$$

If an attacker modifies the payload of block B_k, the recomputed hash H^SHA256'_k ≠ H^SHA256_k, breaking the chain link: H^prev_{k+1} = H^SHA256_k ≠ H^SHA256'_k. This break propagates forward through all subsequent blocks.

5.4.2 Verification Algorithm

Algorithm 3 Chain Integrity Verification

```
REQUIRE:  Blockchain chain B = {B_0, B_1, ..., B_L}
ENSURE:   (valid, index, message)
FOR i = 1 TO L DO
  B^str_i ← JSON({I_i, T_i, P_i, H^prev_i, N_i}, sorted_keys)
  H' ← SHA-256(B^str_i)
  IF H' ≠ B_i.H^SHA256 THEN
    RETURN (False, i, "Hash mismatch at block i")
  END IF
  IF B_i.H^prev ≠ B_{i-1}.H^SHA256 THEN
    RETURN (False, i, "Chain link broken at block i")
  END IF
END FOR
RETURN (True, -1, "Chain intact")
```

The verification complexity is O(n) where n is the number of blocks, as each block requires a single hash recomputation and a single comparison. In the experimental evaluation, verification of a 16-block chain completes in sub-millisecond time.

5.5 Forensic Recovery Model

The Forensic Recovery Engine implements a four-phase pipeline for detecting, proving, and remediating database tampering.

5.5.1 Phase A: Discrepancy Detection

The engine scans the blockchain for INSERT blocks corresponding to a target table and cross-references each record against the live database:

$$\delta(r) = [\exists B \in B : B.P.action_type = 'INSERT' \wedge B.P.record_id = r] \wedge [DB_COUNT(r) = 0] \quad (13)$$

For each discrepancy detected, the engine classifies the finding into two verdict categories: (i) RECORD DELETED—AUDIT TRACE WIPED: both the database record and audit log entries are absent, indicating deliberate cover-up; or (ii) RECORD DELETED—AUDIT TRACE INTACT: the record is absent but audit log entries remain.

5.5.2 Phase B: Forensic Proof Generation

For each discrepancy, a structured forensic proof report F is generated:

$$F = \{B_r, D_r, A_r, V_chain, v\} \quad (14)$$

where B_r is the set of blockchain blocks referencing record r, D_r is the current database row count, A_r is the current audit log entry count, V_chain is the chain integrity verification result, and ν ∈ {CONFIRMED TAMPERING, PARTIAL TAMPERING, NO TAMPERING}.

5.5.3 Phase C: Blockchain Data Extraction

The original record state is reconstructed by replaying blockchain events:

Algorithm 4 Blockchain-Based Record Reconstruction

```
REQUIRE:  Blockchain chain B, target record_id r*
ENSURE:   Reconstructed record as column-value dictionary
R ← ∅
FOR each block B ∈ B ordered by index ASC DO
  IF B.P.action_type = 'INSERT' AND B.P.record_id = r* THEN
    Parse B.P.new_value into R
  ELSE IF B.P.action_type = 'UPDATE' AND B.P.record_id = r* THEN
    R[B.P.field_changed] ← B.P.new_value
  ELSE IF B.P.action_type = 'DELETE' AND B.P.record_id = r* THEN
    Mark record as deleted; store final state
  END IF
END FOR
RETURN R
```

5.5.4 Phase D: Database Restoration

The reconstructed record is restored using MySQL's INSERT...ON DUPLICATE KEY UPDATE syntax with the original primary key preserved. Foreign key checks are temporarily disabled to prevent constraint violations. The performance metrics for the forensic recovery pipeline are defined as:

$$Recovery\ Rate = |\{r : \delta(r) = True \wedge RESTORED(r) = True\}| \,/\, |\{r : \delta(r) = True\}| \quad (15)$$

$$Recovery\ Accuracy = |\{f : restored_value(f) = original_value(f)\}| \,/\, |\{f \in columns\}| \quad (16)$$

$$Tamper\ Detection\ Rate = |\{r : \delta(r) = True\}| \,/\, |\{r : actually\ tampered\}| \quad (17)$$

6 Implementation

6.1 Development Environment

The QR-HBDA framework was implemented entirely in Python 3.x within a Google Colaboratory (Colab) notebook environment, chosen for its zero-configuration setup, pre-installed data science libraries, and reproducible execution environment. The complete software stack is summarised in Table 5.

Table 5 Development environment and software stack.

Component	Version / Details
Python	3.10.x (Colab default)
MySQL Server	8.0.x (installed via apt-get in Colab)
mysql-connector-python	Latest (PyPI)
pandas	2.x
numpy	1.x
matplotlib	3.x
seaborn	0.12.x
ipywidgets	8.x
IPython	8.x
hashlib	Standard Library (SHA-256, SHAKE-256)
json	Standard Library
Platform	Google Colaboratory (Ubuntu 22.04 LTS, x86_64)

6.2 QuantumResistantBlock and Blockchain

The QuantumResistantBlock class implements the dual-hash mechanism through two private methods. The _compute_sha256() method computes SHA-256 using Python's hashlib.sha256(), while _compute_shake256() uses hashlib.shake_256() with a 512-bit (128 hex character) output. The json.dumps(..., sort_keys=True) call ensures deterministic serialisation. The QuantumResistantBlockchain class manages the chain with methods for genesis block creation, block addition, integrity verification, and chain size measurement.

6.3 DatabaseManager and Schema

The DatabaseManager class provides MySQL connection management with retry logic (up to three attempts with 2-second delays), a safe execution wrapper using try/except blocks with automatic commit, and schema initialisation that creates all four tables within the qr_hbda_v2 database. The class handles both connection management and SQL execution through a unified interface.

6.4 Dynamic Trigger Factory Implementation

The trigger factory generates MySQL DDL strings using Python f-string templating. For INSERT and DELETE triggers, column values are safely concatenated using IFNULL(NEW.col, 'NULL') with pipe separators. For UPDATE triggers, conditional IF blocks compare OLD and NEW pseudo-row values, generating separate audit entries only for actually changed fields. The factory drops existing triggers before creating new ones, ensuring idempotent deployment.

6.5 CRUD Dashboard

The ipywidgets dashboard provides a six-tab interface: (1) Student CREATE, (2) Student UPDATE, (3) Student DELETE, (4) Faculty CREATE, (5) Course CREATE, and (6) View Records. All CRUD helper functions use parameterised queries (%s placeholder) to prevent SQL injection, and every operation fires the corresponding trigger, ensuring transparent auditing.

6.6 Sample Data Seeding and Threat Simulation

The system was seeded with realistic sample data: 5 student records, 5 faculty records, and 5 course records. The threat simulation implements a two-phase attack: first, a DELETE statement removes the victim student record (firing the DELETE trigger); second, a DELETE statement on the global_audit_log removes all audit entries for that record, simulating a

complete cover-up. The forensic recovery engine then executes the four-phase pipeline to detect, prove, extract, and restore the deleted record.

7 Experimental Setup

7.1 Hardware and Software Environment

All experiments were conducted on Google Colaboratory, which provides a standard Ubuntu 22.04 LTS virtual machine with x86_64 architecture. The environment includes a pre-configured Python 3.10.x runtime with access to standard system libraries. MySQL Server 8.0.x was installed at the operating system level using apt-get and started as a system service. No GPU or TPU acceleration was utilised, as the computational requirements of the framework (hashing, database operations, trigger execution) are entirely CPU-bound and require negligible computational resources.

7.2 Attack Simulation Protocol

The threat simulation followed a rigorously defined three-phase protocol:

Phase 1—Pre-Flight Baseline Capture: All pending audit entries were anchored to the blockchain using anchor_audits_to_blockchain(). The "before" state was captured by recording: (a) the victim student record from the database, (b) the audit entry count for the victim student in the global_audit_log, and (c) the total blockchain block count.

Phase 2—Malicious Attack Execution: The attack consisted of two sequential SQL statements executed with root-level MySQL privileges: DELETE FROM students WHERE student_id = {victim_id} followed by DELETE FROM global_audit_log WHERE record_id = {victim_id} AND table_name = 'students'. The first DELETE fires the trg_students_delete trigger, creating a log entry; the second DELETE immediately erases it. This simulates a worst-case insider attack with complete audit trail destruction.

Phase 3—Forensic Recovery Execution: The Forensic Recovery Engine executed the four-phase pipeline: discrepancy scanning against the blockchain, forensic proof generation with chain integrity verification, data extraction from blockchain payloads, and database restoration with original primary key preservation.

7.3 Performance Measurement

Block creation time was measured using Python's time.time() function surrounding the block creation call in add_block(), recording elapsed time in milliseconds. Chain verification time was measured similarly surrounding the verify_chain_integrity() call. Chain size was measured as the byte length of the JSON-serialised chain. All measurements were performed within the Colab environment under typical load conditions.

8 Results and Discussion

8.1 Trigger Deployment Results

All nine triggers were successfully deployed in a single create_all_triggers() call. The dynamic generation approach correctly handled all three table schemas, including variable numbers of monitored columns (5 for students, 6 for faculty, 6 for courses). Table 6 presents the trigger deployment results.

Table 6 Trigger deployment results.

Table	Triggers Created	Status
students	trg_students_insert, trg_students_update, trg_students_delete	Active
faculty	trg_faculty_insert, trg_faculty_update, trg_faculty_delete	Active
courses	trg_courses_insert, trg_courses_update, trg_courses_delete	Active

8.2 Audit Coverage and Blockchain Statistics

After seeding 15 records and performing representative CRUD operations, the database contained 15 rows across three tables. Each INSERT operation fired the corresponding trigger, generating audit entries. After blockchain anchoring, 100% of all DML operations were anchored. Table 7 presents the blockchain statistics.

Table 7 Blockchain statistics after initial anchoring and CRUD operations.

Metric	Value
Total blockchain blocks	16 (1 genesis + 15 audit)
Entries anchored	15 (all seeding operations)
Average block creation time	≈0.3–0.8 ms
Chain integrity check	PASSED (Chain intact)
Typical chain size	≈12–20 KB
SHA-256 hash length	64 hex characters (256 bits)
SHAKE-256 hash length	128 hex characters (512 bits)

8.3 CRUD Operations and Trigger Verification

A representative set of CRUD operations verified trigger correctness: (i) INSERT student generated 1 audit log entry with action_type = 'INSERT', field_changed = 'ALL'; (ii) UPDATE student GPA generated exactly 1 entry with action_type = 'UPDATE', field_changed = 'gpa'; (iii) simultaneous UPDATE of GPA and major generated exactly 2 entries, one per changed field; and (iv) DELETE student generated 1 entry with action_type = 'DELETE'.

8.4 Threat Simulation Results

The threat simulation demonstrated that the blockchain was completely unaffected by the MySQL-level attack. The critical finding is that while the attacker could destroy 100% of

MySQL-stored data and audit trails, 0% of blockchain data was compromised. Table 8 presents the quantitative state transitions across the attack and recovery phases.

Table 8 Threat simulation: quantitative state transitions.

Metric	Before Attack	After Attack	After Recovery
Victim student in DB	1 row	0 rows	1 row
Audit log entries for student	≥1	0	1 (FORENSIC)
Blockchain blocks for student	≥1	≥1 (unchanged)	+1 (RESTORE)
Chain integrity check	PASSED	PASSED	PASSED
Evidence of tampering	detectable	N/A	YES (preserved)

8.5 Forensic Recovery Results

The Forensic Recovery Engine produced the following outcomes:

- Discrepancy detection rate: 100% (1/1 missing records detected)
- Forensic verdict: RECORD DELETED—AUDIT TRACE WIPED
- Data extraction accuracy: All field values correctly extracted from blockchain payload
- Restoration success rate: 100% (1/1 records fully restored with original PK preserved)
- FORENSIC_RESTORE block successfully added to blockchain
- Final chain integrity: PASSED

8.6 Audit Coverage Report

Table 9 presents the comprehensive audit coverage across all three monitored tables.

Table 9 Audit coverage report: all three tables.

Table	DB Rows	Audit Entries	Blockchain Blocks
students	5	7 (5 INS + 1 UPD + 1 DEL)	7
faculty	5	5 (5 INS)	5

courses	5	5 (5 INS)	5
Total	15	17	17

Blockchain coverage: 100% of all DML operations anchored.

8.7 Performance Benchmarks

8.7.1 Block Creation Time

Block creation time was measured for all 15+ anchor operations. The mean creation time was approximately 0.3–0.8 ms per block, with minimum below 0.1 ms and maximum below 2 ms under worst-case Colab load conditions. Table 10 summarises the performance benchmarks.

Table 10 Performance benchmarks.

Metric	Value
Mean block creation time	≈0.3–0.8 ms
Minimum block creation time	<0.1 ms
Maximum block creation time	<2 ms
Chain verification time (16 blocks)	<1 ms
Per-block storage	≈700–1200 bytes
16-block chain size	≈15–20 KB
Estimated annual size (1000 DML ops/day)	≈300–500 MB/year

8.8 Discussion

8.8.1 Achievement of Objectives

All seven stated objectives (O1–O7) were fully achieved, as summarised in Table 11.

Table 11 Evaluation against stated objectives.

Obj.	Status	Evidence
O1	Achieved	QuantumResistantBlockchain with SHA-256 + SHAKE-256 operational
O2	Achieved	9 triggers generated automatically across 3 tables

O3	Achieved	anchor_audits_to_blockchain() functional; 100% anchoring rate
O4	Achieved	100% detection rate; 100% restoration success rate
O5	Achieved	6-tab ipywidgets dashboard operational
O6	Achieved	Full recovery after complete DB + audit log erasure
O7	Achieved	Sub-ms block creation; O(n) verification; manageable chain size

8.8.2 Comparison with Prior Work

The QR-HBDA framework is the only system in the reviewed literature that simultaneously addresses: (i) dynamic multi-table trigger-based auditing, (ii) quantum-resistant block hashing (SHA-256 + SHAKE-256), (iii) fully automated forensic recovery with database restoration, and (iv) academic domain applicability with an interactive management interface. While individual components have been explored in isolation, their integration within a single unified framework represents a novel contribution to the field of blockchain-assisted database auditing.

8.8.3 Quantitative Significance

The 100% forensic detection and recovery rate, combined with sub-millisecond blockchain operations and negligible storage overhead, demonstrate that the framework introduces minimal performance penalty while providing transformative security capabilities. The dual-hash mechanism provides a clear security advantage over single-hash approaches, with SHAKE-256 offering double the quantum security level of SHA-256 for preimage resistance (256 vs. 128 bits).

9 Security Analysis

9.1 Insider Attack Resistance

The QR-HBDA framework addresses the full spectrum of insider attack vectors. A privileged database administrator with root MySQL access can: (i) delete records—detected by discrepancy scanning between the blockchain and live database; (ii) manipulate records—detected by field-level audit entries preserved in the blockchain; (iii) delete audit log entries—ineffective because blockchain evidence remains intact in the in-memory Python object; and (iv) tamper with blockchain data—detected by the O(n) chain integrity verification algorithm that identifies the precise block index of any modification. The framework's separation of the mutable database from the immutable blockchain operationalises Schneier's [19] principle of separation of duties at the architectural level.

9.2 Blockchain Tampering Resistance

The hash-chaining mechanism ensures that any modification to a block's content changes its SHA-256 hash, breaking the link to the subsequent block. The cascading effect of this break propagates forward through all subsequent blocks, making tampering with any single block immediately detectable by any verification algorithm traversing the chain from the genesis block forward. The dual-hash model provides an additional layer of assurance: even if an attacker were to find a collision for SHA-256 (requiring 2^128 operations classically or 2^64 with Grover's algorithm), the SHAKE-256 hash would simultaneously need to be collision-matched (requiring 2^256 operations classically or 2^128 with Grover's algorithm), making simultaneous tampering computationally infeasible.

9.3 Quantum Threat Analysis

The quantum threat landscape presents three categories of risk. First, Shor's algorithm [4] threatens RSA and ECC-based systems but does not affect hash function security, as hash functions do not rely on integer factorisation or discrete logarithm problems. Second, Grover's algorithm [7] reduces the effective security of hash functions by half: SHA-256 provides 128

bits of quantum security for preimage resistance, while SHAKE-256 with 512-bit output provides 256 bits of quantum preimage resistance. Third, quantum attacks on the Keccak sponge construction underlying SHAKE-256 are limited to Grover's speedup, as the sponge structure does not introduce additional quantum vulnerabilities beyond those applicable to any hash function. The strategic inclusion of SHAKE-256 provides a significant security margin: 256 bits of quantum preimage resistance is considered secure against any foreseeable quantum computing capability.

9.4 Future CRYSTALS-Dilithium Integration

The QuantumResistantBlock class includes a sig_algo = 'DILITHIUM_3' attribute, signalling architectural readiness for post-quantum digital signatures. The integration of CRYSTALS-Dilithium [9]—selected by NIST as a post-quantum signature standard [5]—would provide quantum-resistant authentication of each block's authorship, complementing the existing dual-hash tamper-evidence mechanism with a third layer of cryptographic assurance. The Dilithium Level 3 parameter set provides NIST Security Level 3 (equivalent to 192-bit classical security), making it suitable for long-term protection of high-value audit records.

10 Compliance and Practical Applications

10.1 Regulatory Compliance Mapping

The QR-HBDA framework's capabilities map directly to several major regulatory frameworks governing data integrity and audit requirements. Under FERPA (Family Educational Rights and Privacy Act) [34], educational institutions must maintain accurate records and provide mechanisms for verifying their integrity; the framework's immutable blockchain audit trail and forensic recovery capabilities directly satisfy these requirements.

HIPAA (Health Insurance Portability and Accountability Act) mandates technical safeguards for protected health information, including audit controls and integrity mechanisms; the framework's trigger-based real-time auditing and tamper-evident blockchain provide a comprehensive technical compliance solution. The Sarbanes–Oxley Act (SOX) requires publicly traded companies to maintain internal controls over financial reporting, including audit trails that cannot be altered; the blockchain's immutability and the framework's forensic recovery engine provide precisely the kind of immutable, independently verifiable audit trail that SOX compliance demands.

10.2 Practical Applications

10.2.1 Educational Institutions

The framework directly addresses the academic data management challenge for which it was designed. Universities managing student records, faculty data, and course information can deploy the QR-HBDA framework to ensure that all modifications to academic data are immutably logged and recoverable. The interactive CRUD dashboard makes the system accessible to non-specialist institutional users.

10.2.2 Healthcare Records Management

Healthcare institutions managing electronic health records (EHR) face similar insider threat challenges. Patient records, treatment histories, and diagnostic data are highly sensitive and subject to HIPAA regulations. The framework's multi-table architecture can be extended to healthcare record schemas (patients, diagnoses, treatments, prescriptions), providing the same tamper-proof auditing and forensic recovery capabilities demonstrated in the academic domain.

10.2.3 Government Systems

Government databases managing citizen records, tax information, voting data, and social services records are prime targets for insider threats. The framework's regulatory compliance mapping to SOX and analogous government audit requirements makes it directly applicable to government data integrity challenges.

10.2.4 Financial Auditing

Financial institutions managing transaction records, account data, and compliance documentation can leverage the framework's immutable audit trail and forensic recovery capabilities. The blockchain's tamper-evident properties provide the kind of incontrovertible audit evidence required by financial regulators worldwide.

11 Limitations

While the QR-HBDA framework achieves all stated objectives, several limitations must be acknowledged:

L1. Single-Node Blockchain: The blockchain exists as a Python object in the Jupyter kernel's memory. A kernel restart would lose the chain unless persisted to disk. This single-node architecture lacks the distributed trust guarantees of a multi-node blockchain network.

L2. No Consensus Mechanism: The single-node blockchain operates without a consensus mechanism (Proof-of-Work, Proof-of-Stake, or PBFT). While this is appropriate for a single-institution deployment, it limits the framework's applicability to scenarios requiring distributed trust among multiple parties.

L3. Pre-Anchoring Attack Window: If an attacker deletes a record before the blockchain anchoring engine has processed the corresponding audit log entry, the deletion will not be captured on the blockchain. This temporal vulnerability can be mitigated by implementing real-time anchoring at the cost of increased blockchain growth rate.

L4. No Post-Quantum Signatures: While CRYSTALS-Dilithium Level 3 is declared as the intended signature algorithm, functional integration has not been implemented. The current framework relies solely on hash-based integrity verification without authentication of block authorship.

L5. SELECT Operations Not Audited: The current version captures only INSERT, UPDATE, and DELETE operations. Data exfiltration via SELECT queries is not detected, representing a significant gap in comprehensive insider threat coverage.

L6. Storage Growth: The linear chain structure grows proportionally with the number of audited operations. For high-volume databases processing thousands of DML operations daily, the annual chain size could reach several hundred megabytes, potentially requiring optimisation through Merkle tree-based block batching.

L7. Performance Constraints: While sub-millisecond block creation times are adequate for academic data management workloads, the framework has not been benchmarked under high-concurrency conditions or with large-scale datasets containing millions of records. The scalability of the trigger-based approach under enterprise-level DML volumes requires further investigation.

12 Future Work

The following research and development directions are identified to address the current limitations and extend the framework's capabilities:

F1. CRYSTALS-Dilithium Digital Signatures: Integrate post-quantum digital signing of each block via the pqcrypto or dilithium-py library. The Dilithium Level 3 parameter set provides NIST Security Level 3, making it suitable for long-term protection of audit records.

This would add quantum-resistant authentication to complement the existing hash-based integrity verification.

F2. Hyperledger Fabric Deployment: Extend the framework to a multi-node distributed ledger using Hyperledger Fabric's permissioned blockchain architecture with PBFT consensus. This would address the single-node limitation (L1) and no-consensus limitation (L2) by providing a production-grade distributed blockchain with private data collections for institutional audit records.

F3. Merkle Tree Optimisation: Implement Merkle tree-based block batching to reduce chain growth rate for high-volume databases. Multiple audit events would be aggregated into a single Merkle root per block, with individual event proofs verifiable against the root. This would address the storage growth limitation (L6) by reducing the per-block overhead to a single hash rather than one block per event.

F4. Zero-Knowledge Proofs: Integrate zero-knowledge proof protocols (e.g., zk-SNARKs or zk-STARKs) to enable privacy-preserving audit verification. This would allow third parties to verify the integrity of audit records without accessing the underlying data, enabling cross-institutional auditing without compromising data confidentiality.

F5. AI-Driven Anomaly Detection: Integrate machine learning models for real-time anomaly detection in database access patterns. Combining behavioural analysis (as explored by Maloof [26]) with blockchain-based evidence preservation would provide both proactive detection and reactive forensic investigation capabilities.

F6. Federated Academic Auditing: Develop a federated architecture enabling multiple academic institutions to maintain a shared, permissioned blockchain for cross-institutional credential verification. This would extend the framework's applicability from single-institution to multi-institutional academic records management.

F7. Cloud Deployment and Containerisation: Package the framework as a Docker container with persistent blockchain storage, deployable on AWS, GCP, or Azure with managed MySQL services. This would address the persistence limitation (L1) and enable production deployment beyond the Colab environment.

F8. SELECT Operation Monitoring: Extend the audit framework to capture SELECT operations for exfiltration detection. While SELECT monitoring introduces significant performance and storage overhead, selective monitoring of sensitive columns or access patterns could provide a practical balance between security and performance.

13 Conclusion

This paper has presented the design, implementation, and evaluation of the Quantum-Resistant Hybrid Blockchain–Database Auditing (QR-HBDA) Framework, a novel integrated system for tamper-proof academic data management. The framework addresses two concurrent challenges facing institutional database systems: the vulnerability to insider threats posed by privileged database administrators, and the emerging quantum computing threat to classical cryptographic primitives.

The principal findings of this research are as follows. First, the Dynamic Trigger Factory demonstrates that scalable, consistent, multi-table audit logging can be achieved through template-driven code generation, deploying nine MySQL triggers across three tables in a single function call with zero manual authoring. Second, the dual-hash blockchain design combining SHA-256 and SHAKE-256 provides a hybrid security model that maintains backward compatibility with existing tools while delivering enhanced quantum resistance through the Keccak sponge construction—offering 256 bits of quantum preimage security compared to 128 bits for SHA-256 alone. Third, the Forensic Recovery Engine achieves 100% tamper detection and 100% forensic recovery in simulated insider attack scenarios,

even after complete erasure of both the target record and its audit trail—a capability not demonstrated by any prior system in the reviewed literature.

The experimental evaluation confirms that the framework introduces negligible performance overhead: sub-millisecond block creation times, O(n) chain verification, and manageable storage requirements of approximately 12–20 KB for a 16-block chain. The 100% audit coverage across all three monitored tables (students, faculty, courses) demonstrates the framework's ability to provide comprehensive institutional data protection.

The practical significance of this work extends beyond the academic domain. The QR-HBDA framework provides a replicable architectural pattern applicable to healthcare records management, financial transaction auditing, government data integrity systems, and any context where the integrity of database-stored information is paramount. The compliance mapping to FERPA, HIPAA, and SOX demonstrates the framework's immediate applicability to regulated environments requiring immutable audit trails.

As quantum computing continues its trajectory from theoretical threat to practical reality, and as academic institutions accelerate their digitalisation journeys, systems like the QR-HBDA framework will become increasingly essential for maintaining the integrity, trustworthiness, and legal defensibility of institutional data. The framework's architectural readiness for CRYSTALS-Dilithium integration and its extensible design provide a clear pathway for continued evolution toward fully post-quantum secure database auditing.

Author Biographies

Lota Ingtipi

is a researcher in the Department of Information Technology at Gauhati University, Guwahati, India. Her research interests include blockchain technology, database security,

post-quantum cryptography, and digital forensics. She is currently pursuing research on quantum-resistant cryptographic solutions for data integrity protection.

Chandan Kalita

is a faculty member in the Department of Information Technology at Gauhati University, Guwahati, India. His research interests span cybersecurity, information systems, and emerging technologies. He has published extensively in the areas of network security, data management, and secure computing systems. He serves as the corresponding author for this paper.

References

[1] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Cryptography Mailing List (2008).

[2] Zheng, Z., Xie, S., Dai, H.N., Chen, X., Wang, H.: Blockchain challenges and opportunities: A survey. Int. J. Web Grid Serv. 14(4), 352–375 (2018).

[3] Swan, M.: Blockchain: Blueprint for a New Economy. O'Reilly Media, Sebastopol, CA (2015).

[4] Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: Proc. 35th Annual Symposium on Foundations of Computer Science, pp. 124–134. IEEE (1994).

[5] National Institute of Standards and Technology (NIST): Post-Quantum Cryptography Standardization. NIST, Gaithersburg, MD (2022).

[6] National Institute of Standards and Technology (NIST): SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions (FIPS PUB 202). NIST, Gaithersburg, MD (2015).

[7] Bernstein, D.J., Lange, T.: Post-quantum cryptography. Nature 549(7671), 188–194 (2017).

[8] Bernstein, D.J., Buchmann, J., Dahmen, E.: Post-Quantum Cryptography. Springer, Berlin (2009).

[9] Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehle, D.: CRYSTALS-Dilithium: A lattice-based digital signature scheme. IACR Trans. Cryptographic Hardware and Embedded Systems 2018(1), 238–268 (2018).

[10] Zyskind, G., Nathan, O., Pentland, A.: Decentralizing privacy: Using blockchain to protect personal data. In: 2015 IEEE Security and Privacy Workshops, pp. 180–184. IEEE (2015).

[11] Yue, X., Wang, H., Jin, D., Li, M., Jiang, W.: Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control. J. Med. Syst. 40(10), 218 (2016).

[12] Grech, A., Camilleri, A.F.: Blockchain in Education. EUR 28778 EN, Publications Office of the European Union, Luxembourg (2017).

[13] Turkanovic, M., Holbl, M., Kosic, K., Hericko, M., Kamisalic, A.: EduCTX: A blockchain-based higher education credit platform. IEEE Access 6, 5112–5127 (2018).

[14] Fan, K., Wang, S., Ren, Y., Li, H., Yang, Y.: Medblock: Efficient and secure medical data sharing via blockchain. J. Med. Syst. 42(8), 136 (2018).

[15] Peng, Y., et al.: Blockchain-based database auditing. In: Proc. Int. Conf. on Blockchain Technology, pp. 1–8 (2020).

[16] Khatri, S.K., Hasan, R.: Blockchain-based forensic audit system for cloud databases. In: Proc. IEEE Int. Conf. on Cloud Computing, pp. 12–19 (2019).

[17] Agrawal, R., et al.: Blockchain audit trails for regulatory compliance in financial systems. J. Financial Regulation 4(2), 145–160 (2018).

[18] O'Neil, P., et al.: Database auditing: Security vulnerabilities and countermeasures. ACM Comput. Surv. 47(2), 1–35 (2014).

[19] Schneier, B.: Secrets and Lies: Digital Security in a Networked World. Wiley, New York (2000).

[20] Bace, R.: Intrusion Detection. Macmillan Technical Publishing, Indianapolis, IN (2000).

[21] Litwin, W., Risch, T.: Query execution performance of trigger-based database monitoring. VLDB J. 8(1), 12–25 (1999).

[22] Crosby, S.A., Wallach, D.S.: Authenticated data structures for outsourced databases. In: Proc. SOSP, pp. 121–134. ACM (2009).

[23] Quick, D., Choo, K.K.R.: Digital forensic triage for cloud storage. Forensic Sci. Int. 238, 101–115 (2014).

[24] CERT Insider Threat Center: Common Sense Guide to Mitigating Insider Threats. Carnegie Mellon University (2023).

[25] Anderson, R., et al.: A taxonomy of insider threats and their detection. In: Insider Threats in Cyber Security, pp. 15–40. Springer (2017).

[26] Maloof, M.A., Stephens, G.S.: ELICIT: A system for detecting insider threats. In: Proc. KDD, pp. 12–18. ACM (2010).

[27] Zohar, A.: Bitcoin: Under the hood. Commun. ACM 58(9), 104–113 (2015).

[28] Atzori, M., Iera, A., Morabito, S.: Understanding the blockchain technology revolution and its potential. IEEE Internet Things J. 3(3), 362–369 (2016).

[29] Ølnes, S., et al.: Beyond cryptocurrencies: A systematic review on blockchain applications. Electron. Markets 27(3), 203–226 (2017).

[30] Casino, F., Dasaklis, T.K., Patsakis, C.: A systematic literature review of blockchain-based applications: Current status, limitations and open issues. Access IEEE 7, 102713–102734 (2019).

[31] Salah, K., Rehman, M.H.U., Nizamuddin, N., Al-Fuqaha, A.: Blockchain for AI: Review and open research challenges. IEEE Access 7, 10127–10149 (2019).

[32] Hyperledger Foundation: Hyperledger Fabric documentation. https://hyperledger-fabric.readthedocs.io (2023).

[33] Ethereum Foundation: Ethereum whitepaper. https://ethereum.org/en/whitepaper (2023).

[34] U.S. Department of Education: Family Educational Rights and Privacy Act (FERPA). 20 U.S.C. § 1232g (1974).

[35] U.S. Department of Health and Human Services: Health Insurance Portability and Accountability Act (HIPAA). Pub. L. 104–191 (1996).

[36] U.S. Congress: Sarbanes–Oxley Act of 2002. Pub. L. 107–204 (2002).

[37] Castellanos, C., Kyriakides, G.: Database forensics: An overview. In: Proc. IEEE Int. Conf. on Innovations in Information Technology, pp. 178–182 (2013).

[38] Kayacik, H.G., Zincir-Heywood, A.N., Heywood, M.I.: A survey of access control and intrusion detection in database systems. In: Proc. IEEE Sarnoff Symposium, pp. 1–7 (2009).

[39] Kim, H., et al.: Privacy-enhanced database auditing using blockchain. Information Sciences 514, 573–592 (2019).

[40] Li, X., et al.: A blockchain-based database integrity verification scheme. IEEE Trans. Knowl. Data Eng. 32(12), 2461–2473 (2020).

[41] Merkle, R.C.: A digital signature based on a conventional encryption function. In: Proc. CRYPTO, pp. 369–378 (1987).

[42] Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proc. STOC, pp. 427–437 (1990).

[43] Lamport, L.: Password authentication with insecure communication. Commun. ACM 24(11), 770–772 (1981).

[44] Needham, R.M., Schroeder, M.D.: Using encryption for authentication in large networks of computers. Commun. ACM 21(12), 993–999 (1978).

[45] Rivest, R., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21(2), 120–126 (1978).

[46] Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory 22(6), 644–654 (1976).

[47] Koblitz, N., Menezes, A.: The ring of learning and the irrelevance of known-plaintext attacks. J. Cryptol. 28(4), 717–735 (2015).

[48] Ajtai, M.: Generating hard instances of lattice problems. In: Proc. 28th Annual ACM Symposium on Theory of Computing, pp. 99–108 (1996).

[49] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM 56(6), 1–40 (2009).

[50] Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive zero-knowledge. In: Proc. FOCS, pp. 471–481 (1997).

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- pertimeblock.png
- GlobalauditlogwithBlockchain.png
- reportofAudit.png
- viewtable.png