Scalable Blockchain-based Healthcare Consent Management with Automated Compliance Verification

***Abstract*—Managing consent for Electronic health record (EHR) faces critical challenges in modern healthcare systems. Paper-based consent mechanisms lack granular permission control and cannot scale to manage millions of patients. While blockchain technology offers immutable audit trails essential for sensitive medical data, its low transaction throughput makes cross-patient aggregation queries infeasible. We present a novel protocol that maintains blockchain's cryptographic guarantees while delegating queries to high-performance lookup tables, enabling fast category lookups across millions of patients. Our system combines hierarchical consent trees, blockchain audit trails, and event-driven synchronization with reconciliation mechanisms. Additionally, we implement automated GDPR compliance verification through zero-knowledge proofs, enabling processors to cryptographically prove compliance without exposing processed data. Evaluation demonstrates practical query performance enabling real-time dataset compilation while maintaining cryptographic integrity and sub-second proof verification.**

***Index Terms*—Electronic health records, Blockchain, Consent management, GDPR compliance, Zero-knowledge proofs**

I. Introduction

HEALTHCARE systems face a fundamental challenge in patient consent management. Patients currently provide consent through paper-based forms signed at hospital admission, creating binary "all-or-nothing" permission structures that lack the fine-grained control required for modern healthcare data usage [1]. As healthcare systems scale to manage tens of millions of patients, traditional consent management approaches become infeasible.

The challenge extends beyond scale: research institutions regularly need to compile datasets by querying across the entire patient population. For example, a cardiovascular research study might require identifying all patients who have consented to sharing their cardiology records, laboratory results, and imaging data. Such queries must scan millions of consent records to assemble the cohort [2]. These cross-patient aggregation queries are fundamental to modern medical research, yet they require detecting consent changes across millions of patients, managing fine-grained permissions at the category level, and providing sub-second query responses.

Beyond consent management, GDPR defines two key roles in healthcare data sharing: data controllers (hospitals that own and govern patient data) and data processors (pharmaceutical companies, research institutes that analyze data under contractual agreements). When controllers share patient data with processors, regulations require Data Processing Agreements (DPAs) that specify GDPR compliance requirements. However, verifying that processors actually adhere to these negotiated terms —such as k-anonymity thresholds, data minimization requirements, and processing time constraints— relies on manual inspection and periodic audits, creating significant compliance overhead and introducing verification delays that can span weeks or months [3].

Blockchain technology offers compelling properties for healthcare consent management. Immutable audit trails provide tamper-proof records of consent changes, distributed consensus enables transparent verification without central authorities, and cryptographic hashing ensures data integrity. However, blockchain systems face a fundamental performance limitation: consensus-based transaction processing achieves throughput of only a few hundred transactions per second (TPS) in production deployments like Hyperledger Fabric [4], compared to 10,000+ TPS in traditional relational databases.

This throughput gap creates a critical architectural challenge. While blockchain excels at recording individual consent updates with cryptographic guarantees, it cannot efficiently handle cross-patient aggregation queries essential for research cohort compilation. A pharmaceutical company designing a clinical trial needs to query "retrieve all patients who consented to cardiovascular research and pulmonary data sharing", a request that requires scanning millions of blockchain records and filtering by multiple consent categories. This fundamental tension between trust (cryptographic guarantees) and performance (query speed) represents what we term the *blockchain trust-performance dilemma*. Existing blockchain-based healthcare systems either sacrifice performance by querying blockchain directly (resulting in minutes-long delays), or sacrifice cryptographic guarantees by using blockchain merely as an append-only log with no verification mechanism for off-chain query results.

To address this fundamental limitation while preserving blockchain's security guarantees, this paper presents a system that resolves this dilemma through a novel protocol. Our approach maintains blockchain as the authoritative source of truth while delegating high-performance queries to purpose-based lookup tables. The key innovation lies not in simply combining blockchain with traditional databases but in BRIDGE (Bidirectional Reconciliation for Integrity-Driven ledGer-databasE sync), our protocol that maintains cryptographic trust during delegation. We store patient consent in hierarchical trees with Merkle-like hash computation, recording consent changes on the blockchain with category-level hashes. A sync manager listens to blockchain events and updates the lookup tables in real-time, with reconciliation mechanisms using blockchain as the source of truth to resolve discrepancies without full rescans.

Our system also addresses compliance verification through

automated cryptographic proofs. We encode GDPR compliance requirements as machine-readable smart contract terms (k-anonymity thresholds, data minimization, processing constraints), which processors then cryptographically prove adherence to through zero-knowledge proofs. These proofs are verified on-chain, providing real-time compliance assurance that eliminates manual audit overhead.

This paper presents a comprehensive system that addresses the above-mentioned challenges through the following contributions:

- Three-tier data structure architecture with novel BRIDGE protocol that enables practical real-time queries while maintaining cryptographic integrity
- Zero-knowledge proof system for automated GDPR compliance verification, complementing periodic manual audits with automated cryptographic verification for quantifiable compliance requirements

The remainder of this paper is organized as follows. Section II provides background on the key technologies employed in our system. Section III examines prior work in blockchain-based healthcare systems, hybrid architectures, and compliance verification, positioning our contributions. Section IV presents our complete system architecture, detailing the dual data structure design, blockchain integration, and synchronization mechanisms. Section V describes our automated GDPR compliance verification approach. Section VI presents our implementation and evaluation, demonstrating scalability, performance, and security properties. Section VII concludes with future research directions.

II. Background

This section introduces the four key foundations underlying our system: blockchain, hierarchical tree structures, zero-knowledge proofs, and purpose-based access control.

A. Blockchain

A blockchain consists of a chain of blocks, each containing a set of transactions and a cryptographic hash of the previous block, creating an immutable sequence. Distributed consensus protocols (such as Practical Byzantine Fault Tolerance or Proof-of-Work) ensure that all network participants agree on the ledger state without requiring a central authority. Smart contracts —programs executed on the blockchain— enable automated enforcement of business logic, such as access control rules or consent validation policies.

Blockchain technology provides an immutable, distributed ledger that ensures tamper-proof audit trails through cryptographic hashing and consensus mechanisms. These properties make blockchain attractive for healthcare consent management where regulatory compliance demands verifiable records of consent decisions. Once written to the blockchain, consent records cannot be retroactively altered, providing strong guarantees for dispute resolution and GDPR Article 7(1), which requires demonstrable consent.

B. Hierarchical Tree Structures

Tree data structures naturally represent hierarchical consent relationships at multiple granularity levels. In healthcare, patients consent to purposes (research, treatment), categories within purposes (cardiology, oncology), and individual data elements. Tree structures support efficient traversal of these hierarchies and inheritance semantics where parent-level consent applies to children unless explicitly overridden. Graph databases like Neo4j provide native support for storing and querying tree structures through the Cypher query language.

C. Zero-Knowledge Proofs

Zero-knowledge proofs (ZKPs) are cryptographic protocols enabling a prover to demonstrate knowledge or correctness of a computation to a verifier without revealing underlying data. Modern ZKP systems include zk-SNARKs (Succinct Non-Interactive Arguments of Knowledge) and zk-STARKs (Scalable Transparent Arguments of Knowledge), offering different trade-offs in proof size, verification time, and trust assumptions. For healthcare, ZKPs enable processors to prove regulatory compliance —such as achieving k-anonymity thresholds or maintaining encryption— without exposing patient data or proprietary algorithms.

D. Purpose-Based Access Control

Purpose-based access control restricts data usage based on intended purpose, directly implementing GDPR's purpose limitation principle (Article 5(1)(b)). Unlike role-based access control (RBAC) which grants permissions by job function, purpose-based systems evaluate whether data access aligns with the consent purpose. This enables patients to specify that cardiac data may be used for research but not marketing.

III. Related Work

Given the blockchain trust-performance dilemma described in Section I, prior work has explored various architectural approaches to healthcare consent management and compliance verification. We examine these approaches across four categories, analyzing how each addresses the fundamental tension between cryptographic guarantees and query performance requirements.

A. Blockchain-Based Healthcare Consent Systems

Azaria et al. [5] demonstrated blockchain's application to healthcare with MedRec, a system using Ethereum smart contracts for EHR access control. MedRec provides immutable audit trails and patient-controlled access logs through on-chain permission structures. However, storing consent metadata directly on the blockchain creates severe performance bottlenecks: cross-patient aggregation queries require scanning the entire ledger, resulting in query times exceeding 70 seconds even for moderately-sized patient populations. Blockchain's consensus-based processing limits transaction throughput to a few hundred TPS in production deployments, making real-time cohort identification infeasible

at scale.
Shrivastava and Srikanth [6] embedded consent structures in tree representations of EHRs with a tree traversal algorithm for conflict resolution. Their hierarchical approach enables fine-grained consent preferences at multiple granularity levels. While effective for representing hierarchical relationships and supporting patient-specific queries, their architecture lacks cryptographic integrity mechanisms. Without hash-based verification, detecting which consent trees have changed requires examining each tree individually—an O(n) operation across millions of patients. This makes efficient change detection infeasible at scale and provides no tamper-proof guarantees for regulatory compliance.

B. Hybrid Storage Architectures

Recent approaches have explored combining blockchain with traditional databases to improve query performance. Systems like hOCBS [7], EHRGuard [8], and Achealthchain [9] store patient data in conventional databases while recording access permissions or pointers on the blockchain. These architectures achieve better read performance by querying databases directly rather than scanning blockchain records.
However, these hybrid approaches differ fundamentally from rigorous synchronization protocols in three aspects. First, most use blockchain as a pointer store or access log rather than an authoritative source for consent state. The blockchain records that access occurred but does not maintain cryptographically verifiable consent structures. Second, existing architectures lack formal reconciliation mechanisms to detect and resolve synchronization failures between blockchain and off-chain storage. When events are missed or systems restart, there is no systematic way to rebuild correct state. Third, prior systems do not address incremental synchronization at scale—they either perform full database scans periodically or accept eventual consistency without verification.
These limitations mean that while hybrid architectures improve query speed, they sacrifice the cryptographic guarantees that make blockchain attractive for regulatory compliance. Query results from off-chain databases cannot be cryptographically verified against the blockchain, and synchronization failures can result in consent state diverging from the authoritative ledger.

C. Zero-Knowledge Proofs for Healthcare Compliance

Zero-knowledge proofs have been explored for privacy-preserving healthcare verification, primarily in identity authentication and access control. A systematic review [10] identified ZKPs as a key privacy protection technique in healthcare blockchain systems, noting applications in anonymous authentication and unlinkable transactions. Irfan et al. [11] specifically addressed GDPR compliance in dynamic consent systems, proposing zero-knowledge proofs to meet Article 28 processor obligations.
However, existing ZKP applications focus primarily on authentication —proving identity or authorization to access data— rather than processing compliance verification. Proving that a user is authorized to access patient records differs fundamentally from proving that data processing operations adhere to contractual constraints. To our knowledge, no prior work demonstrates ZKP generation for proving compliance with actual data processing requirements. Such requirements include verifying k-anonymity thresholds during anonymization, ensuring only permitted data fields were accessed (data minimization), and confirming that processing occurred within mandated time windows. Additionally, existing work does not integrate machine-readable DPA terms encoded in smart contracts with ZKP-based verification systems, leaving the compliance verification process manual and audit-dependent.

D. Purpose-Based Access Control Systems

Purpose-based access control has been recognized as essential for implementing GDPR's purpose limitation principle in healthcare systems. Kim et al. [12]'s Hyperledger Fabric implementation demonstrates how patients can specify purposes for each data element, with smart contracts enforcing that data access requests match consented purposes. This provides transparency and auditability for purpose compliance.
However, purpose-based access control alone does not address the performance requirements of large-scale medical research. Systems that store all purpose-consent mappings on blockchain face the same aggregation query limitations as other blockchain-only approaches. Finding all patients who consented to a specific research purpose requires scanning millions of on-chain records, making real-time cohort identification impractical for clinical trial design and epidemiological studies.

IV. System Design and Architecture

This section presents our system architecture, describing how the design overcomes the challenges identified in the Introduction.

A. System Overview

Our architecture consists of three primary data structures, each optimized for specific operations, unified through the BRIDGE protocol that maintains consistency and cryptographic integrity across all components. Figure 1 illustrates the complete system architecture.

The system comprises three components:

Hierarchical Consent Trees store patient-level consent with fine-grained control over purposes, categories, and individual data elements. These trees support efficient traversal and inherit Merkle tree properties through bottom-up hash computation.

Purpose-Based Lookup Tables provide high-performance aggregation queries across millions of patients. Organized by purpose and category, these tables enable fast lookups to answer queries like "all patients consenting to cardiovascular research" without scanning individual consent records.

Distributed Ledger maintains an immutable audit trail of all consent changes, storing cryptographic hashes (root and category-level) rather than full consent data. This provides

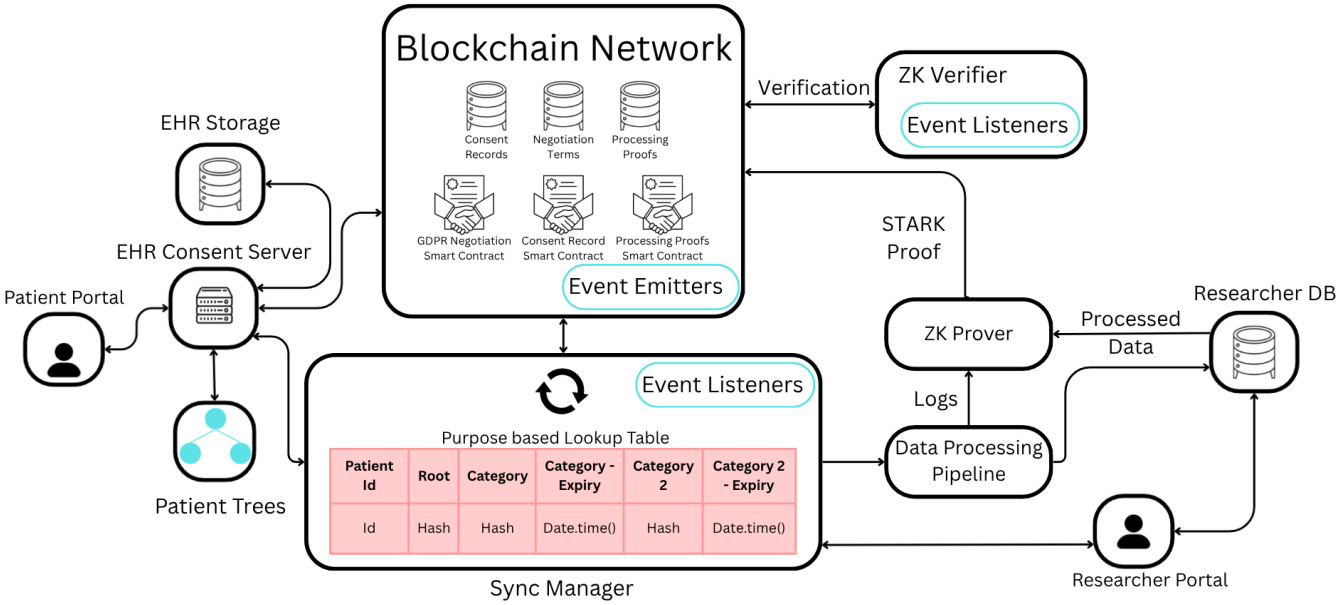

Fig. 1: System Architecture

tamper-proof verification that lookup table contents accurately reflect patient consent.

The key insight is separation of concerns: the distributed ledger guarantees immutability but delegates query operations to the lookup tables, while hierarchical trees manage granular patient consent. The BRIDGE protocol maintains cryptographic links between all three components, ensuring that fast queries from lookup tables can be verified against the immutable ledger through hash comparison.

B. Hierarchical Consent Representation

We represent patient consent as a four-level hierarchical tree structure. The root node represents the patient, with three descending levels:

- **Purpose level**: Broad data usage categories (e.g., Research, Consensus, Marketing)
- **Category level**: Specific domains within purposes (e.g., Cardiology, Pulmonary, Oncology under Research)
- **Document level**: Individual data elements (e.g., specific laboratory results, imaging records, medication lists)

This hierarchy naturally mirrors how healthcare organizations structure data usage and how patients conceptualize consent decisions [13]. Patients can grant consent at any level. Consenting to "Research" implicitly grants access to all research subcategories unless explicitly overridden. Trees implement a sparse representation, storing only nodes where consent is granted. The absence of a node indicates lack of consent. This design provides two critical benefits: first, scalability through storage efficiency, as storing only consented items reduces space requirements by orders of magnitude across thousands of potential data points per patient. Second, dynamic consent management, as patients can grant or withdraw consent by adding or removing nodes, with hash recomputation automatically propagating changes bottom-up through the hierarchy.

Each non-leaf node maintains a SHA-256 hash computed from its children's content through bottom-up traversal. Document nodes (leaves) have no hash and represent references to actual data. Category node hash is calculated by taking the SHA-256 hash of the concatenation of all child document identifiers. The category node also contains a `valid_until` timestamp that specifies the temporal validity of the consent provided by that patient for that category. Purpose node hash is calculated by taking the SHA-256 hash of the concatenation of all category hashes under that purpose. Finally, the root hash is calculated by taking the SHA-256 hash of the concatenation of all purpose hashes.

Hash recalculation occurs automatically after any tree modification. When a patient grants consent to a new document, the system recomputes hashes for: (1) the category containing that document, (2) the purpose containing that category, and (3) the root node. This propagation enables efficient change detection. Comparing two root hashes instantly reveals whether any consent modifications occurred.

C. Dual Data Structure with Distributed Ledger

The system writes all operations to the blockchain for audit immutability, then synchronizes consent hierarchies to the lookup tables for fast retrieval:

1) Hierarchical Trees for Granular Consent: Hierarchical trees excel at patient-specific operations, including retrieving all categories a patient consented to, modifying individual document permissions, and answering queries like "which cardiovascular documents did patient X authorize?". The tree structure supports efficient traversal and naturally represents consent hierarchies. Moreover, this structure allows us to map one-to-many relationships efficiently, allowing us to map the documents consented by a patient as leaf nodes in their tree. In this case, fetching all consented documents of a patient for a category becomes a very efficient operation.

However, trees are unsuitable for cross-patient aggregation. Finding "all patients who consented to cardiovascular research" requires iterating through millions of individual patient trees, an operation that becomes prohibitively expensive at scale.

2) Purpose-Based Lookup Tables for Aggregation: Lookup tables invert the query perspective. Instead of organizing by patient (trees), they organize by purpose and category. Each purpose (Research, Consensus) has a dedicated table with columns for every category in that purpose. Each row represents one patient, with columns storing the patient identifier, the category hash for each category (matching the hash from the patient's hierarchical tree), and the temporal validity from the category node.

For example, a Research lookup table might have columns: `patient_id`, `cardio_hash`, `cardio_valid_until`, `pulmonary_hash`, `pulmonary_valid_until`, etc. A non-empty hash indicates consent; an empty hash indicates no consent. This schema enables fast category queries through database indexing: "all patients consenting to cardiovascular research" becomes a simple indexed lookup on the `cardio_hash` column, returning results in milliseconds. Critically, lookup tables store hashes, not consent data. The hash values provide cryptographic proof that the lookup table accurately reflects the patient's hierarchical tree.

3) Distributed Ledger for Immutable Audit Trail: The distributed ledger records every consent modification as an immutable transaction, creating a tamper-proof audit trail as mandated by GDPR Article 7(1). Each consent record on the ledger contains the patient identifier, the purpose, the root and category hashes, and the `valid_until` timestamp along with the timestamp specifying the moment of consent modification. The ledger stores separate records per patient-purpose combination rather than a single record per patient. This design optimizes for consent update granularity.

D. BRIDGE protocol

The BRIDGE protocol maintains consistency between hierarchical trees, lookup tables, and the distributed ledger despite network failures, service restarts, and missed updates. The algorithm operates in two modes: event-driven synchronization for normal operations and reconciliation for ensuring consistency before dataset compilation queries or fault recovery.

1) Event-Driven Synchronization: Event-driven synchronization provides real-time updates while avoiding full database scans. When a patient modifies their consent preferences, the hierarchical tree is updated and hashes are recomputed bottom-up from the affected leaf nodes to the root. The consent server then writes a transaction to the distributed ledger containing the patient ID, purpose identifier, root hash, and all category hashes for that purpose. Upon successful ledger commit, the ledger emits an event containing the patient ID, purpose, and affected category. Notably, the event does not include hash values themselves, which must be fetched separately.

The sync manager continuously listens for these ledger events. Upon receiving an event, it immediately queries the ledger for that specific patient-purpose-category combination to retrieve the current hash values. Finally, the sync manager updates the corresponding row in the PostgreSQL lookup table with the new category hashes and timestamps, ensuring the lookup table reflects the latest blockchain state within seconds. Both the consent server and sync manager maintain local tracking of changed patients. When the consent server successfully writes to both the hierarchical tree and the ledger, it records that patient's identifier and current root hash in a local "changed patients" list. Similarly, when the sync manager receives an event and updates lookup tables, it records that patient's identifier and root hash in its own "changed patients" list. These lists enable reconciliation by providing not just which patients changed, but their cryptographic state for verification.

2) Reconciliation Mechanism: Reconciliation ensures consistency despite various failure scenarios including missed events, network partitions, service restarts, or transaction failures. The system triggers reconciliation automatically before every dataset compilation request.

During reconciliation, the sync manager sends its "changed patients" list to the consent server, which compares this list against its own tracking. If the lists match, both systems clear their tracking lists. However, if the lists differ, the system identifies the specific discrepancies and uses the distributed ledger as the authoritative source of truth to resolve them.

For patients appearing in the consent server's list but absent from the sync manager's list, the consent server queries the distributed ledger. If the patient exists on the ledger, this indicates that the sync manager missed an event, and the consent server instructs the sync manager to fetch that patient's data directly from the ledger and update the lookup tables accordingly.

Conversely, if the patient is missing from the ledger entirely, this reveals that the original ledger write operation failed, and the consent server retries writing the transaction to the ledger using the current hierarchical tree state. This bidirectional verification ensures that neither transient network failures nor partial system failures can result in permanent inconsistencies. In normal operation with rare failures, discrepancies are minimal. Even with 1% event loss affecting 100 patients out of 10,000 daily changes, reconciliation processes only 100 records.

E. System Workflows

This section presents two key workflows demonstrating how components interact.

1) Workflow 1 Patient Consent Update: A patient grants or revokes consent for specific categories and documents through the patient portal. The consent server modifies the hierarchical tree accordingly, adding or removing nodes as needed. This triggers automatic hash recalculation from the modified category upward through the purpose level to the root node. Once hash propagation completes, the consent server creates a blockchain transaction containing the patient identifier, purpose, and updated category hashes. It then records the patient in its local "changed patients" tracking list. The blockchain

network validates and commits the transaction, then broadcasts an event notification. The sync manager receives this event through its active blockchain subscription. It then queries the ledger to retrieve current hash values and applies these updates to the appropriate lookup table entry, storing both the category hash and temporal validity. Finally, it records the patient in its own "changed patients" tracking list.

2) Workflow 2 Research Dataset Compilation: When a researcher requests a dataset for specific categories, the sync manager first performs reconciliation with the consent server to resolve discrepancies. After reconciliation, the sync manager executes an indexed query on the lookup table to retrieve consenting patients, then requests document references from the consent server, which traverses hierarchical trees and compiles the dataset from the EHR database. Figure 2 shows this flow.

F. Design Properties

Our architecture achieves four key properties essential for large-scale healthcare consent management:

1) Cryptographic Integrity: Hash-based verification ensures lookup table contents can be proven correct against the immutable ledger. Any tampering or corruption of lookup tables is detectable through hash comparison with ledger records.

2) Fault Tolerance: The reconciliation mechanism provides resilient fault tolerance by treating the distributed ledger as the single source of truth. System components can crash, restart, or miss updates without data loss. Reconciliation rebuilds correct state from the ledger.

3) Scalability: Event-driven synchronization operates in $O(k)$ time where k is the number of recent changes, rather than $O(n)$ where n is all patients. Lookup table queries achieve very fast throughputs through database indexing. Reconciliation processes only discrepancies, not entire databases.

4) Flexibility: Sparse tree representation supports dynamic category addition without schema modifications or data migrations. When a new research study introduces a new category, patient trees grow organically as patients grant consent, and lookup tables add columns dynamically.

V. Automated GDPR Compliance Verification

Section IV addressed scalable consent management enabling patients to grant fine-grained permissions and researchers to efficiently compile datasets of consenting patients. However, as established in Section I, consent alone is insufficient for GDPR compliance. This section presents our automated compliance verification approach using zero-knowledge proofs to transform compliance from periodic manual audit to continuous cryptographic assurance.

A. Machine-Readable Compliance Requirements

Our system encodes GDPR compliance requirements as machine-readable smart contract terms on the distributed ledger. While not all GDPR requirements are amenable to mathematical verification (organizational policies, staff training, incident response), a significant subset can be expressed as quantifiable constraints. We identify five mathematically verifiable requirements:

- **K-Anonymity Thresholds** (GDPR Article 32): Dataset must satisfy k-anonymity where each quasi-identifier combination appears for at least k individuals
- **l-Diversity Guarantees** (GDPR Article 32): Each quasi-identifier group contains at least l distinct values for sensitive attributes
- **Data Minimization** (GDPR Article 5(1)(c)): Processors access only explicitly permitted data fields
- **Temporal Processing Constraints** (GDPR Article 5(1)(e)): Processing occurs within specified time windows
- **Encryption Requirements** (GDPR Article 32): Data remains encrypted throughout processing with specified algorithms

Controllers and processors negotiate these specific parameters (e.g., $k = 10$, permitted fields: age and diagnosis only, processing window: 30 days) through smart contract proposals recorded on the blockchain. Finalized agreements create transparent, auditable DPA terms stored immutably on the ledger as authoritative references.

B. Zero-Knowledge Proof System for Processing Compliance

Once DPA terms are established in smart contracts, processors must prove they adhere to the negotiated requirements during data processing. Traditional verification requires processors to expose data, processing algorithms, or intermediate computation states to auditors creating privacy risks and revealing proprietary methods. Zero-knowledge proofs resolve this: processors generate cryptographic proofs demonstrating compliance without revealing any private information.

Our system employs zkSTARK (Zero-Knowledge Scalable Transparent Argument of Knowledge) proofs, which provide post-quantum security and require no trusted setup ceremony. The architecture involves three parties:

- **Prover (Data Processor)**: Executes data processing and generates compliance proofs with access to the full patient dataset, processing algorithms, and DPA terms from the ledger
- **Verifier (Distributed Ledger + Verification Service)**: Validates submitted proofs knowing only the DPA terms, claimed compliance values, and cryptographic proof
- **Data Controller (Hospital)**: Monitors compliance through a dashboard querying ledger records with cryptographic guarantees

1) Proof Generation Workflow: Zero-knowledge proof generation occurs during the processing phase:

- **Dataset Request & Pre-Processing**: Processor requests a dataset, receives raw data, and applies required transformations (removing personally identifiable information (PII), anonymizing quasi-identifiers per DPA)
- **zkSTARK Prover Preparation**: Processor prepares private inputs (anonymized dataset and hash commitment) and public inputs (DPA terms from smart contract and processing metadata)

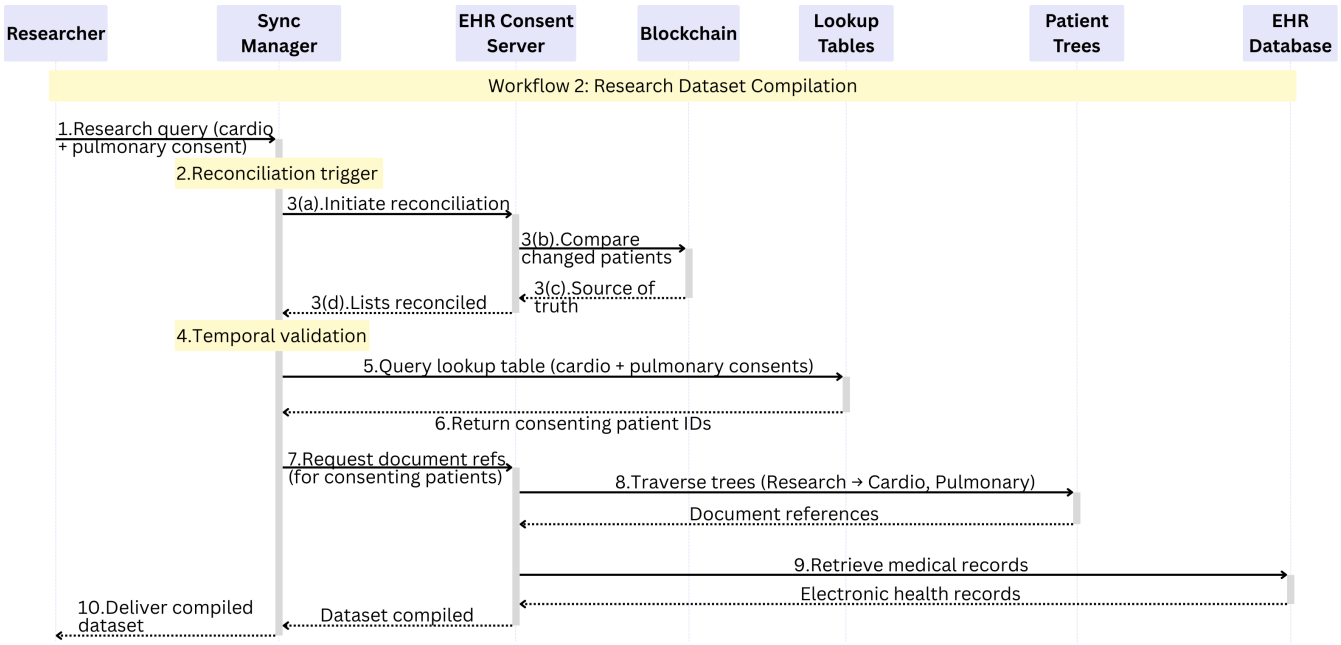

Fig. 2: Workflow 2: Research Dataset Compilation

- **Computation & Proof Generation**: The zkSTARK virtual machine executes compliance verification computations using the private dataset, calculating all DPA-required metrics (k-anonymity, l-diversity, field validation, temporal checks, encryption verification). The zkVM produces public outputs (achieved compliance values and boolean results) and cryptographic proof π attesting to correct execution

Proof verification time remains logarithmic relative to computation. Critically, the proof reveals nothing about private inputs while proving compliance calculations were executed correctly.

2) Compliance Verification Workflow: After generating the zkSTARK proof, the processor submits a compliance record to the distributed ledger containing the agreement identifier, processor details, public outputs, compliance boolean, and proof π. The ledger emits a `ComplianceProofSubmitted` event upon successful storage.

A verification service listens for these events and automatically verifies the zkSTARK proof against public inputs and outputs, confirming cryptographic validity and that the compliance boolean correctly reflects DPA requirement satisfaction. The service updates the ledger record with verification status and timestamp. If verification fails or compliance is false, an alert event notifies the data controller.

The system provides transparency through a compliance dashboard showing all processors who requested datasets, governing DPA terms, and public compliance values for each operation. The dashboard indicates verification status, compares DPA requirements versus achieved values, and surfaces non-compliance incidents. Because all records are stored on the distributed ledger, regulators can independently verify the complete audit trail: DPA establishment before processing, proof submission for every access, cryptographic verification of all proofs, and permanent recording of non-compliance incidents.

This compliance verification system integrates seamlessly with consent management. When processors request datasets, the system performs comprehensive verification checks. It confirms that a valid DPA exists with established compliance terms, data categories match the DPA subject matter, and patients granted consent for the stated purpose. Each compliance record references consent hashes, creating an immutable link between "which patients consented" and "how their data was processed", providing complete lifecycle traceability.

VI. Implementation & Evaluation

This section presents our proof-of-concept implementation and experimental evaluation, demonstrating that our architecture achieves 37 to 64 times query performance improvement over pure blockchain approaches (with speedup inversely related to patient population scale) while maintaining cryptographic integrity.

A. Implementation Details

We implemented the complete system architecture as a proof-of-concept demonstrating feasibility and measuring performance characteristics.

1) Hierarchical Consent Trees: Implemented using Neo4j graph database with the Cypher query language for tree traversal and node manipulation. SHA-256 hash computation follows bottom-up traversal, with category hashes computed from document identifiers and root hashes computed from purpose hashes.

2) Distributed Ledger: Deployed on Hyperledger Fabric (version 2.5) using a Kubernetes operator for container orchestration. The blockchain network consists of multiple peer nodes and orderer nodes implementing the Raft consensus protocol. Smart contracts (chaincode) written in Go implement consent storage functions and DPA term storage. Event emission occurs automatically upon successful transaction commit.

3) Purpose-Based Lookup Tables: Implemented in PostgreSQL (version 15) with dedicated tables for each purpose. B-tree indexes on `patient_id` and category hash columns enable fast lookups. The sync manager, implemented in Node.js, listens to blockchain events via the Fabric SDK and updates lookup tables in real-time.

4) EHR Consent Server & Sync Manager: Built with Node.js and Express framework, these servers orchestrate interactions between Neo4j trees, the blockchain network, and the EHR data service. They maintain SQLite-based tracking of changed patients for reconciliation and provide RESTful API endpoints.

5) Zero-Knowledge Proof System: Implemented using RISC0 zkVM (zero knowledge virtual machine) with the circuits written in Rust. We developed five separate circuits corresponding to the compliance requirements: k-anonymity calculation, l-diversity measurement, data minimization verification, temporal constraint checking, and encryption validation. Each circuit executes inside the zkVM with the anonymized dataset as private input and DPA terms as public input. A separate verification service monitors blockchain events for proof submissions and validates proofs.

6) Deployment Architecture: The Hyperledger Fabric network runs on a Kubernetes cluster deployed on a local development machine (8 CPU cores, 8GB RAM) for proof-of-concept demonstration. All other components run as separate services on the same machine.

B. Experimental Methodology

We designed experiments to evaluate three key questions: (1) Does the hybrid architecture provide significant query performance improvements over pure blockchain approaches? (2) How does performance scale with patient population size? (3) Are zero-knowledge proofs feasible for real-time compliance verification?

We generated synthetic test datasets at scales ranging from 500,000 to 10,000,000 patients. For each patient, we generated hierarchical consent trees with the Research purpose containing four categories and the Consensus purpose containing three categories.

To isolate the performance benefit of our dual data structure architecture, we implemented two query methods:

- **Blockchain Method**: Queries scan the distributed ledger directly, representing the pure blockchain approach used by existing systems like MedRec [5] and Kim et al. [12]
- **Hybrid Method**: Queries leverage PostgreSQL lookup tables synchronized with the blockchain via our BRIDGE protocol, maintaining cryptographic verification through hash comparison.

We tested two query patterns: single-category queries, which retrieve all patients consenting to one specific category (e.g., cardiology), and multi-category queries, which retrieve patients consenting to multiple categories simultaneously (e.g., both cardiology and pulmonary). Each query was executed 5 times with the median reported.

For compliance verification, we generated proofs for the five GDPR requirements using generated test datasets. We measured proof size in bytes and verification time. While RISC0 provides GPU acceleration capabilities, our evaluation used CPU-only execution.

C. Performance Evaluation

This section presents performance results from our hybrid blockchain-database architecture.

1) Query Performance Results: Table I presents performance results for single-category and multi-category queries at the 4,000,000 patient scale.

TABLE I: Query Performance Comparison at 4M Patients: Blockchain vs. Hybrid Approach (times in seconds)

Query Type	Blockchain	Hybrid	Speedup
Single: Cardiology	139.5 s	3.8 s	36.8×
Single: Pulmonary	124.3 s	3.2 s	38.8×
Single: Neurology	134.8 s	3.7 s	36.4×
Multi: Cardio+Neuro	98.3 s	2.2 s	44.7×
Multi: Cardio+Pulm+Onco	84.5 s	1.7 s	49.7×

The hybrid architecture achieves approximately 37–50× speedup across all query patterns at the 4M patient scale. Single-category queries complete in 3.2–3.8 seconds compared to 124–140 seconds for blockchain scanning—representing practical response times versus prohibitively slow blockchain operations. Multi-category queries demonstrate slightly better speedup compared to single-category queries due to the intersection logic reducing the result set size. At the 4M patient scale, approximately 2M patients consent to cardiology research alone, whereas only 300K patients consent to all three categories (cardiology, pulmonary, and oncology) simultaneously. This smaller cohort requires traversing fewer patient trees to retrieve document identifiers, resulting in faster execution times despite the additional filtering complexity.

2) Scalability Analysis: Table II compares query performance across population scales.

TABLE II: Scalability Across Patient Population Sizes (Selected Scales)

Scale	Consenting	Blockchain	Hybrid	Speedup
Single-Category: Cardiology				
500K	322,923	21.4 s	0.35 s	61.1×
2M	1,073,616	75.4 s	1.35 s	55.8×
4M	2,074,511	139.5 s	3.78 s	36.9×
6M	3,106,342	-	6.03 s	-
8M	4,109,628	-	8.37 s	-
10M	5,141,052	-	14.23 s	-
Multi-Category: Cardio+Pulmonary+Oncology				
500K	91,022	13.2 s	0.27 s	48.9×
2M	181,958	40.5 s	0.83 s	48.8×
4M	302,871	84.5 s	1.73 s	48.9×
10M	765,384	-	12.62 s	-

When total patient population increases 8 times (from 500K to 4M), blockchain query times for cardiology increase 6.5

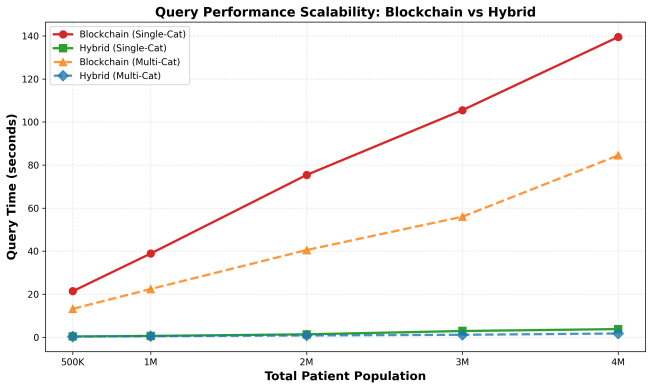

Fig. 3: Query Performance Scalability: Blockchain vs. Hybrid Approach

times (from 21.4s to 139.5s), demonstrating sub-linear yet substantial scaling degradation. Blockchain query times at 4M patients (140 seconds) already approach impractical levels. Extrapolating this trend, a 10M patient deployment would require 300+ second query times ($\geq$5 minutes).

The hybrid method exhibits controlled scaling characteristics across the full range. Single-category cardiology queries scale from 0.35s at 500K patients to 14.23s at 10M patients, maintaining sub-15-second response times even at the largest scale. Multi-category queries exhibit steeper scaling due to larger result set intersections, reaching 12.62s at 10M patients, yet maintain practical response times under 13 seconds. These results validate production viability: even a 10M patient healthcare system can execute complex cohort identification queries in under 15 seconds—orders of magnitude faster than the 300+ seconds that would be required by pure blockchain approaches. Figure 3 visualizes these scaling characteristics up to 4M patients.

3) Zero-Knowledge Proof Performance: Our compliance verification system uses RISC0's zkSTARK implementation. Generating zkSTARK proofs for compliance verification is computationally intensive, with proof generation ranging from minutes to hours depending on dataset scale. However, RISC0 provides GPU acceleration capabilities that can reduce generation time by orders of magnitude. This generation overhead is acceptable for compliance scenarios where proof generation occurs asynchronously.

Verification times remain sub-second for all tested scales, consistent with zkSTARK's polylogarithmic verification property. This enables data controllers to instantly verify compliance proofs upon blockchain submission.

Generated proofs range from 150–500 KB depending on compliance check complexity. While zkSTARK proofs can be 300–700$\times$ larger than equivalent SNARK proofs, this size tradeoff provides critical security advantages. zkSTARKs eliminate the trusted setup requirement, removing a potential security vulnerability. For healthcare compliance, this transparency is essential. The generation overhead represents a substantial improvement over traditional compliance verification, which requires manual audits spanning weeks or months.

D. Comparative Analysis

To contextualize our results, we compare against blockchain-based cohort selection frameworks and position our system architecturally. Table III synthesizes capabilities across systems discussed in Section III.

1) Query Performance Comparison: While our hybrid architecture achieves query latency of approximately 14 seconds for 10 million records, privacy-preserving architectures utilizing homomorphic encryption or secure multi-party computation struggle to maintain interactivity at this scale. The Privacy-Preserving Cohort Discovery (PPCD) framework [14], evaluated on 7.1 million records, reported query execution times ranging from 165 to 262 seconds depending on cohort complexity. Architectures that rely on direct traversal of off-chain pointers exhibit severe linear degradation; the HealthChain framework [15], utilizing IPFS for storage, recorded latency of 97.5 seconds for batch retrieval of only 1,000 records. Foundational smart contract-based matching systems like the re-engineered MedRec prototype [16] required 1.39 seconds to match a cohort from just 6,000 patients. These results confirm that while cryptographic privacy and decentralized storage are essential, their direct application to aggregation logic introduces latency penalties (ranging from 60$\times$ to 100$\times$ slower than our approach) that render them unsuitable for real-time feasibility analysis in clinical research. Our system also demonstrates advantages over standard permissioned blockchain implementations. Benchmarks of Hyperledger Fabric utilizing CouchDB for "Rich Queries" indicate that complex queries on datasets exceeding 100,000 records frequently trigger transaction timeouts or incur latencies upwards of 120 seconds due to state database indexing overhead [17].

2) Architectural Feature Comparison: Table III compares architectural capabilities across consent management and compliance systems. Our system uniquely combines fine-grained hierarchical consent with scalable aggregation performance demonstrated up to 10 million patients. The integration of smart contract-based DPA terms with ZKP verification enables automated compliance checking that addresses gaps in existing architectures. This comparative analysis validates that our BRIDGE protocol resolves the trust-performance dilemma: achieving high-performance cross-patient queries (37–64$\times$ faster than pure blockchain) while maintaining cryptographic guarantees through hash-based verification. The feature synthesis confirms our system addresses the complete lifecycle from consent representation through dataset compilation to compliance verification.

VII. Conclusion and Future Work

This paper presented a comprehensive system for healthcare consent management that resolves the fundamental tension between blockchain's trust guarantees and the performance requirements of large-scale medical research. Our three-tier architecture —hierarchical consent trees, purpose-based lookup tables, and blockchain audit trails— achieves 37–64$\times$ query speedups while maintaining cryptographic integrity through the BRIDGE protocol with reconciliation mechanisms. Evaluation demonstrates practical query performance up to 10

TABLE III: Comparison of Healthcare Consent and Compliance Systems

System	Consent Granularity	Query Performance	Cryptographic Integrity	DPA Encoding	Automated Compliance
MedRec [5]	Limited (document-level)	Poor (70+ sec for cross-patient)	✓ (Blockchain)	×	×
Shrivastava et al. [6]	Fine (purpose → category → field)	Good (graph queries $\leq$2s)	× (No integrity)	×	×
Kim et al. [12]	Medium (purpose-based)	Poor (on-chain storage)	✓ (Hyperledger)	×	×
ConsentChain [18]	Fine (granular permissions)	Good (off-chain)	✓ (Blockchain)	×	×
Patient-Provider Agreements [19]	Medium (role-based)	Medium	✓ (Smart contracts)	×	Partial (access only)
Our System	**Fine (4-level hierarchy)**	**Excellent ($\leq$15s up to 10M)**	**✓ (Hash-sync)**	**✓ (Smart contracts)**	**✓ (ZKP verification)**

million patients with response times under 15 seconds. Our zero-knowledge proof system transforms compliance verification from periodic manual audits to continuous cryptographic assurance.

Future work should address several key areas. First, multi-jurisdictional compliance should be achieved by encoding requirements from multiple regulatory frameworks (HIPAA, CCPA) to enable international data sharing. Second, cross-organization consent federation protocols are needed for consent propagation across hospital networks. Third, secure multi-party computation and federated learning techniques should be investigated to respect fine-grained consent for collaborative machine learning. Finally, performance optimization should focus on automated schema migration, improved PostgreSQL query plans, caching layers, and column-store databases for analytical workloads.

REFERENCES

[1] F. von Scherenberg, M. Hellmeier, and B. Otto, "Data sovereignty in information systems," *Electronic Markets*, vol. 34, 02 2024.

[2] Y. Shen, J. Yu, J. Zhou, and G. Hu, "Twenty-five years of evolution and hurdles in electronic health records and interoperability in medical research: Comprehensive review," *J Med Internet Res*, vol. 27, p. e59024, Jan 2025.

[3] O. A. Cejas, M. I. Azeem, S. Abualhaija, and L. C. Briand, "Nlp-based automated compliance checking of data processing agreements against gdpr," *IEEE Transactions on Software Engineering*, vol. 49, no. 9, pp. 4282–4303, 2023.

[4] S. Gupta, F. Lang, U. Ozeer, and G. Salaün, "Blockchain as a service for electronic health records," in *2025 IEEE International Conference on Digital Health (ICDH)*, pp. 91–101, 2025.

[5] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "Medrec: Using blockchain for medical data access and permission management," in *2016 2nd International Conference on Open and Big Data (OBD)*, pp. 25–30, 2016.

[6] S. Shrivastava and T. K. Srikanth, "A comprehensive consent management system for electronic health records in the healthcare ecosystem," in *Information Security and Privacy in Smart Devices: Tools, Methods, and Applications* (C. Rabadão *et al.*, eds.), pp. 194–233, IGI Global, 2023.

[7] K. Miyachi and T. Mackey, "hocbs: A privacy-preserving blockchain framework for healthcare data leveraging an on-chain and off-chain system design," *Information Processing & Management*, vol. 58, p. 102535, 02 2021.

[8] S. B. Othman and M. Getahun, "Leveraging blockchain and iomt for secure and interoperable electronic health records," *Scientific Reports*, vol. 15, no. 12358, 2025.

[9] A. M. Tawfik, A. Al-Ahwal, A. S. T. Eldien, *et al.*, "Achealthchain blockchain framework for access control and privacy preservation in healthcare," *Scientific Reports*, vol. 15, no. 16696, 2025.

[10] K. Sabiri, F. Sousa, and T. Rocha, "A systematic review of privacy-preserving blockchain applications in healthcare," *Multimedia Tools and Applications*, vol. 84, pp. 39925–39980, 2025.

[11] M. I. Khalid, M. Ahmed, and J. Kim, "Enhancing data protection in dynamic consent management systems: Formalizing privacy and security definitions with differential privacy, decentralization, and zero-knowledge proofs," *Sensors*, vol. 23, no. 17, 2023.

[12] D. Tith, J. S. Lee, H. Suzuki, W. M. A. B. Wijesundara, N. Taira, T. Obi, and N. Ohyama, "Patient consent management by a purpose-based consent model for electronic health record based on blockchain technology," *Healthcare Informatics Research*, vol. 26, pp. 265–273, Oct. 2020.

[13] I. Kassam, D. Ilkina, J. Kemp, H. Roble, A. Carter-Langford, and N. Shen, "Patient perspectives and preferences for consent in the digital health context: State-of-the-art literature review," *J Med Internet Res*, vol. 25, p. e42507, Feb 2023.

[14] J. Yuan, B. Malin, F. Modave, Y. Guo, W. R. Hogan, E. Shenkman, and J. Bian, "Towards a privacy preserving cohort discovery framework for clinical research networks," *Journal of Biomedical Informatics*, vol. 66, pp. 42–51, 2017.

[15] J. Xu, K. Xue, S. Li, H. Tian, J. Hong, P. Hong, and N. Yu, "Healthchain: A blockchain-based privacy preserving scheme for large-scale health data," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8770–8781, 2019.

[16] Y. Zhuang, L. Zhang, X. Gao, Z.-Y. Shae, J. J. P. Tsai, P. Li, and C.-R. Shyu, "Re-engineering a clinical trial management system using blockchain technology: System design, development, and case studies," *J Med Internet Res*, vol. 24, p. e36774, Jun 2022.

[17] K. Azbeg, O. Ouchetto, and S. Jai Andaloussi, "Blockmedcare: A healthcare system based on iot, blockchain and ipfs for data management security," *Egyptian Informatics Journal*, vol. 23, no. 2, pp. 329–343, 2022.

[18] F. Albalwy, A. Brass, and A. Davies, "A blockchain-based dynamic consent architecture to support clinical genomic data sharing (consentchain): Proof-of-concept study," *JMIR Med Inform*, vol. 9, p. e27816, Nov 2021.

[19] M. Al Amin, H. Tummala, S. Mohan, and I. Ray, "Healthcare policy compliance: A blockchain smart contract-based approach," *arXiv preprint arXiv:2312.10214*, 2023.