OPEN ACCESS

ISAR Journal of Multidisciplinary Research and Studies

Volume 4, Issue 2, 2026 | pp: 7-19

Abbriviate Title- ISAR J Mul Res Stud

ISSN *(Online)*- 2583-9705

https://isarpublisher.com/journal/isarjmrs

Towards Scalable and Decentralized Healthcare Blockchains: Dynamic Consensus Optimization for Raft-Based Ordering

Emmanuel Siman[1*] and Tsokwa Daniel[2]

[1,2]Department of Computer Science, College of Applied and Natural Sciences, Kwararafa University Wukari, Taraba State, Nigeria.

[1]Department of Computer Science, Faculty of Computer Sciences, Federal University Wukari, Taraba State, Nigeria.

***Corresponding Author**

Emmanuel Siman

Article History

Received: 15.11.2025

Accepted: 02.01.2026

Published: 11.02.2026

Abstract: This paper presents Dynamic Consensus Optimization (DCO) that integrates online feedback-control framework that augments a Raft-based ordering service with real-time parameter adaptation. DCO continuously monitors transaction arrival rates, queue lengths, and network conditions, and dynamically tunes core Raft parameters such as batching size, election timeouts, and heartbeat intervals without compromising Raft's safety guarantees. We apply DCO to a simulated healthcare environment, where IoT-enabled devices, sensors, and inter-institution data transfers generate highly variable workloads. The proposed approach yields several advantages. First, the blockchain layer sustains low confirmation latency as IoT workloads vary in volume and complexity across participating institutions. Second, the system scales more gracefully: as the number of member hospitals or patient records grows, DCO dynamically adjusts resources to maintain responsiveness without manual reconfiguration or centralized bottlenecks. Third, decentralization is reinforced through leadership rotation and distributed validation tasks, mitigating centralization of power and reducing single points of failure. By continuously balancing speed, scalability, and safety, DCO enables healthcare providers to adopt blockchain-enabled data sharing with stronger performance guarantees and fewer operational frictions. The paper also outlines a formal argument for safety preservation under dynamic parameter updates and presents an evaluation plan to quantify latency, throughput, scalability, and resilience under heterogeneous IoT workloads.

Keywords: Dynamic Consensus Optimization; Raft-Based Ordering; Blockchain in Healthcare; IoT Workloads; Adaptive Parameter Tuning.

Cite this article:

Siman, E., and Daniel, T., (2026). Towards Scalable and Decentralized Healthcare Blockchains: Dynamic Consensus Optimization for Raft-Based Ordering. *ISAR Journal of Multidisciplinary Research and Studies*, *4*(2), 7-19.

1.0 Introduction

The deployment of distributed ledger technologies (DLTs) in healthcare ecosystems offers the promise of secure, auditable, and interoperable data sharing across multiple institutions. However, achieving low-latency transaction validation at scale, while preserving the strong consistency and safety guarantees inherent in traditional consensus protocols, remains a central challenge. Classic Raft-based ordering services provide strong safety and linearizability guarantees, but fixed-configuration parameters (e.g., batching size, election timeouts, and heartbeat intervals) can lead to suboptimal performance under dynamic workloads and heterogeneous network conditions. In healthcare consortia, where transaction arrivals are influenced by IoT-enabled medical devices, sensors, and inter-institutional data transfers, workload characteristics can vary widely across time and place. This variability frequently manifests as fluctuating arrival rates, evolving queue backlogs, and diverse network conditions, all of which can degrade confirmation latency and throughput if the consensus system cannot adapt in real time. To address latency, throughput, scalability, and safety concerns, we introduce Dynamic Consensus Optimization (DCO). DCO augments a Raft-based ordering service with an online feedback controller that continuously monitors key indicators consisting of transaction arrival rates, queue lengths, and network conditions and uses these observations to adjust core consensus parameters. Specifically, DCO modulates batching size, election timeouts, and heartbeat intervals in a manner that aims to minimize latency and maximize

throughput while guaranteeing that Raft's safety properties remain intact. This online, closed-loop control approach enables the system to respond to transient rise in load, bursts of IoT activity, or degraded network paths without sacrificing the formal guarantees that practitioners rely on for safety-critical healthcare data.

1.1 Background of the Research

Blockchain is a decentralized, shared ledger that ensures transparency, security, and immutability by utilizing cryptographic hashing and consensus rules. Since Bitcoin popularized it [1], [2], blockchain has evolved a lot and now supports platforms like Ethereum and Hyperledger [3], each offering its own strengths from decentralized control, stronger security and the thrilling potential for smart contracts [4]. The Internet of Things (IoT) has transformed healthcare by enabling remote monitoring, telemedicine, and the use of smart medical devices. Advances in sensors, wireless technology, and data analytics have driven this growth, with wearable health devices, connected equipment, and predictive analytics becoming more common to improve patient care and streamline operations. In healthcare, sharing data securely is essential for better patient outcomes and research, so strong security and privacy measures are critical to protect sensitive information, especially under regulations like HIPAA and GDPR [5]. However, healthcare data management faces real challenges of data silos and interoperability challenges that hinder seamless data exchange, and security concerns remain a constant risk for many organizations [6], [7]. Blockchain makes it possible to handle large amounts of data transparently without significant degradation, which is a big challenge for healthcare systems [8]. Integrating blockchain with IoT in healthcare offers a promising way to address these issues [9]. By providing secure, tamper-proof data sharing and stronger patient privacy through encryption and decentralized control, blockchain can support seamless data exchanges and better interoperability across different healthcare systems and devices [10], [11]. It also creates a transparent, auditable trail of data transactions, which helps build trust among stakeholders. This kind of integration could spur innovation in healthcare, improve the patient experience, and reduce costs, encouraging more exploration and adoption of blockchain for secure data sharing in healthcare IoT environments. The following contributions are as follows::

i. Introduced Dynamic Consensus Optimization (DCO) for real-time tuning of Raft consensus parameters.

ii. Demonstrated that dynamic tuning preserves Raft's safety and liveness guarantees.

iii. Designed a scalable and decentralized healthcare consortium blockchain architecture.

iv. Developed a comprehensive evaluation framework for healthcare IoT workloads.

v. Achieved lower latency, higher throughput, and improved resource efficiency compared to static Raft.

The remainder of the paper is organized as follows: Section 2 reviews related work on blockchain, IoT, and healthcare data management, highlighting existing limitations and research gaps. Section 3 presents the research methodology, including the system model, Dynamic Consensus Optimization (DCO) design, simulation setup, and evaluation metrics. Section 4 details the DCO algorithm and its integration with the Raft-based consensus mechanism, along with safety and scalability analysis. Section 5 discusses the experimental results and performance evaluation under varying healthcare IoT workloads. Finally, Section 6 concludes the paper by summarizing key findings and outlining directions for future research.

2. Literature Review

The healthcare sector demands strong data integrity, traceability, access control, and privacy due to patient safety and regulatory requirements. Blockchain supports these needs through immutability, tamper-evident audit trails, and smart contracts for automated governance, consent, and access control, making it suitable for EHR management. However, healthcare environments face strict regulations (HIPAA, GDPR), interoperability challenges, real-time IoT data streams, and low-latency requirements, creating trade-offs between privacy, performance, and governance that necessitate careful blockchain design. Blockchain is a decentralized ledger secured by cryptography and consensus, enabling transparent, tamper-resistant transaction recording across nodes [12]. Public blockchains offer openness and decentralization but suffer from scalability and energy inefficiency due to PoW/PoS mechanisms [13]. In contrast, permissioned blockchains trade some decentralization for lower latency, higher throughput, and regulatory alignment [14], [15]. Smart contracts enable automated and auditable governance [16] but introduce security risks requiring formal verification [17], [18]. Hyperledger Fabric exemplifies permissioned design with modular components and Raft-based ordering, offering fine-grained access control and privacy features suitable for healthcare, though configuration complexity and latency trade-offs remain [19]–[21]. Healthcare blockchain platforms must balance privacy, interoperability, and timely data access. Sensitive health data require strong privacy and auditability [22], while care delivery depends on interoperability across EHR systems and devices [23]. Permissioned blockchains are often preferred due to controlled participation and regulatory governance [24]–[26], whereas permissionless platforms raise governance and compliance concerns, requiring off-chain storage and strict access controls [27]. Performance is critical in healthcare, where systems must handle routine EHR transactions, IoT-generated bursts, and emergency spikes with low latency. Permissioned blockchains provide predictable performance but remain sensitive to policy design, data placement, and topology [28]. Permissionless platforms struggle with scalability and energy efficiency and rely heavily on off-chain and privacy-preserving techniques to protect PHI [29].

Privacy and access control further differentiate platforms. Hyperledger Fabric supports private data collections, channels, and fine-grained policies to minimize data exposure and ensure auditability. Permissionless systems often depend on cryptographic privacy mechanisms and off-chain storage, complicating governance and compliance [30]. Interoperability is shaped by alignment with standards such as HL7 FHIR [31], with limited support increasing integration overhead and latency [28]. Prior studies highlight governance challenges including multi-party administration, policy harmonization, onboarding complexity, and integration with legacy EHR systems. These findings emphasize the need for strong stakeholder coordination, data governance, and evaluation frameworks that assess both technical and clinical

impact [32]. EHR and IoT data management introduces challenges in governance, security, privacy, and interoperability [7]. The EHR lifecycle requires immutable records, strict access controls, and auditability [33], [34], while IoT devices generate continuous data streams that raise security risks such as impersonation, integrity breaches, and provenance issues [35].Blockchain-enabled EHR systems commonly combine on-chain metadata with off-chain data storage to balance scalability and integrity [30]. Access control often integrates RBAC and ABAC with private data collections [36], while patient-centric consent mechanisms support dynamic sharing preferences [37]. Privacy-preserving techniques such as zero-knowledge proofs and selective disclosure further reduce unnecessary data exposure [38]. Security relies on encryption, key management, and secure communication, while compliance with HIPAA and GDPR remains an evolving challenge [32]. The overarching goal is to ensure traceability and accountability without excessive PHI exposure [39]. Interoperability depends on adherence to standards like HL7 FHIR, though seamless integration across platforms remains difficult [40]. Dynamic Consensus Optimization (DCO) addresses performance bottlenecks caused by static Raft configurations in healthcare blockchains. Variable IoT workloads and demand spikes can increase latency and reduce throughput when consensus parameters are fixed [41], [42]. DCO introduces a feedback-driven mechanism that dynamically adjusts consensus settings based on real-time workload and network metrics, aligning with adaptive system design principles [43].Architecturally, DCO operates as a monitoring and control layer interfacing with Raft ordering services, enabling real-time parameter tuning while respecting governance and security constraints [44]. Expected benefits include improved performance, efficient resource utilization, and resilience in inter-hospital and cloud-edge deployments [45], [46]. However, challenges such as reconfiguration stability, security during adaptation, and rigorous validation remain critical [47]–[50]. Comprehensive evaluation must assess latency, throughput, leadership fairness, scalability, and fault tolerance under realistic conditions [51]. Therefore, the literature reveals limited work on automated, real-time consensus tuning in healthcare blockchains, highlighting the need for systematic evaluation of decentralization, performance, and regulatory compliance under dynamic configurations.

3. Research Methodology

The Dynamic Consensus Optimization (DCO) module is designed to smartly adjust consensus parameters like batch size, heartbeat interval, election timeout, and snapshot frequency based on the current load. This helps to reduce access "collisions," which can lead to issues like transaction queue build-up, delays in committing, and unnecessary leader re-elections., validator (or leader) nodes monitor current workload and resource status, then adapt their local and cluster-wide parameters to maintain efficient transaction scheduling and block production as nodes join or leave and as traffic fluctuates.

3.1 Research Design and Development

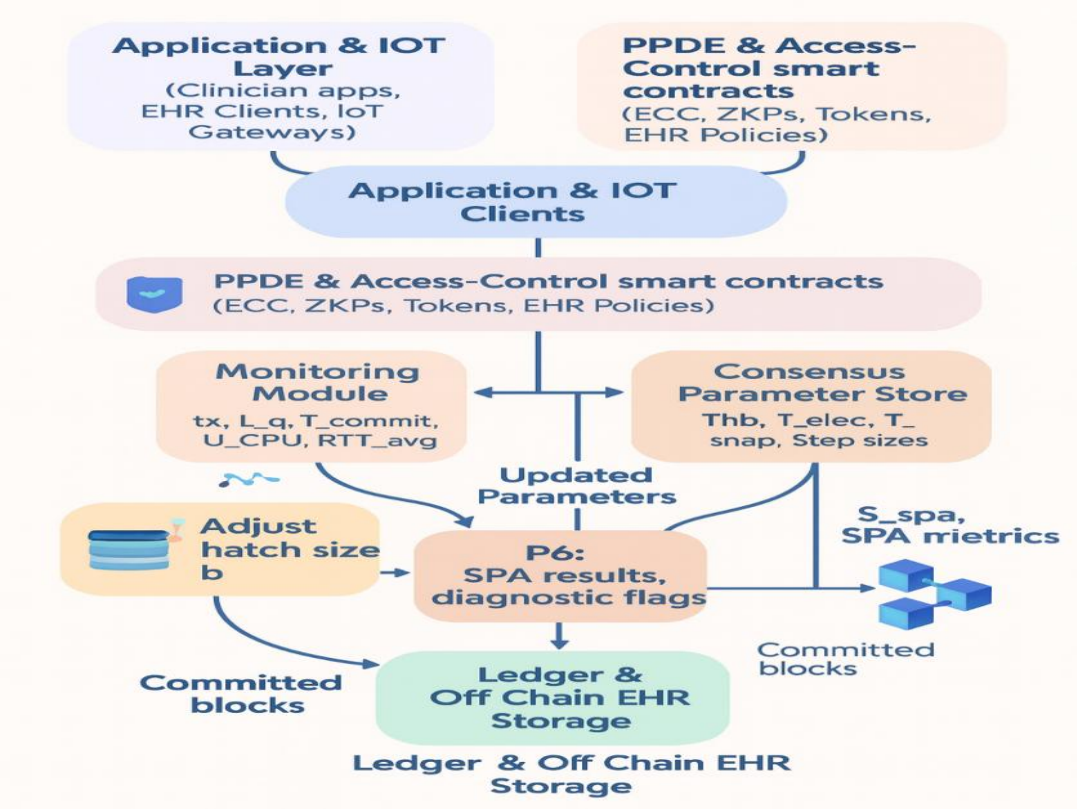

Figure 1: Architecture of the Dynamic Consensus Optimization (DCO) Algorithm

The architecture is structured into four distinct vertical layers. At the top is the Application & IoT Layer, which includes clinician apps, EHR clients, and IoT gateways. This layer is responsible for generating encrypted EHR transactions that are then sent to the PPDE & Access-Control Smart Contracts where ECC, ZKPs, and token-based policies are enforced. Once validated, these transactions move into the DCO-enhanced Raft Consensus Layer, which consists of three collaborative components: the Monitoring Module that gathers workload and health metrics, the DCO Controller (Figure 1) that adjusts parameters like (b), (T_{hb}), (T_{elec}), and (f_{snap}), while also managing leadership rotation, and the Raft Runtime where leader and follower nodes carry out log replication and commit. A Consensus Parameter Store keeps track of current parameters and thresholds, feeding this information back into the Raft Runtime. After blocks are committed, they are recorded in the Ledger & Off-Chain EHR Storage layer, which contains the blockchain ledger, audit logs, and links to external EHR databases or IPFS. This setup illustrates how DCO optimizes consensus internally while keeping the PPDE logic and storage layers intact.

Table 1 Simulation parameters and their values

Parameter	Value
Blockchain platform	Hyperledger Fabric
Consensus mechanism	Raft and DCO Algorithm
Simulation tools	MATLAB, Hyperledger Caliper
Cryptographic techniques	AES, Zero-Knowledge Proofs (ZKP)
IoT device density	50, 100, 150, 200 devices
Network load	Low, Medium, High
Transaction latency	≤ 500 ms
Transaction throughput	≥ 1000 transactions per second
Frequency band	5 GHz
Energy efficiency	Monitored across nodes
Data storage	IPFS (off-chain storage)
Privacy metrics	Compliance with GDPR and HIPAA
Evaluation metrics	Scalability, privacy, resilience
Smart contract language	Go (chaincode)
Simulation time	60 minutes per scenario

Table 1 summarizes the main settings used to evaluate our blockchain-enabled EHR system, utilizing DCO algorithm. We ran a permissioned network based on Hyperledger Fabric, with smart contracts written in Go and tested using Hyperledger Caliper. After each run, MATLAB was used to process the performance logs. For security, we use AES to keep data confidential and zero-knowledge proofs (ZKPs) to enable privacy-preserving authorization. Data objects stored off-chain in IPFS, while only data hashes and metadata are stored on-chain. To stress-test the workload, we varied the number of IoT devices from 50 to 200 and tested different network load levels (low, medium, high). We also fine-tuned transactions arrival rate to observe if the system can sustain at least 1000 transactions per second, while keeping end-to-end latency under 500ms. Other factors, like the wireless frequency band, how many clients could run concurrently, and a 60-minute duration for each scenario, define the communication and timing context for each test. The evaluation metrics also assesses the privacy and resilience of the framework but with more focus on assessing how scalable and robust the integrated DCO algorithm performs.

3.2 Simulation Evaluation

The proposed DCO is evaluated through simulations that utilize a variety of performance metrics aimed at assessing the efficiency, responsiveness, and scalability of the blockchain-enabled EHR system captured in Table 3.1.

Table 3.2: Simulation Evaluation

Metrics	Summary
Transaction throughput	Number of app-level transactions recorded on the ledger in a time window.
End-to-end transaction latency	Time from a client submitting a request to its full commitment and visibility to authorized users.
Successful access probability	Fraction of requests completed correctly and policy-compliant out of all requests submitted.
Leader change rate	How often the consensus leader changes over time, signaling stability under varying workloads and faults.
Packet Delivery Ratio (PDR)	Proportion of PPDE messages delivered out of those sent, reflecting control-plane reliability for access decisions and tokens.
Energy Efficiency	Overhead from blockchain, DCO loops, especially on resource-limited IoT gateways.
Resource Overhead (CPU and Bandwidth)	Extra CPU and network traffic due to DCO's monitoring and control versus static Raft.
Decentralization Fairness	Degree leadership time and committed blocks are fairly distributed among validators over a period.

4. Dynamic Consensus Optimization (DCO) Algorithm

The Dynamic Consensus Optimization (DCO) mechanism uses the Raft consensus algorithm to enhance transaction validation speed while ensuring scalability and decentralization. The DCO algorithm is presented to augment a Raft-based ordering service with an online feedback controller that constantly monitors transaction arrival rates, queue lengths, and network conditions. It then fine-tunes consensus parameters like batching size, election timeouts, and heartbeat intervals, all while upholding Raft's safety guarantees. In a healthcare consortium, this approach enables the blockchain layer to maintain low confirmation latency even with varying IoT workloads, scale effectively with more participating institutions, and prevent centralization by rotating leadership and distributing validation tasks among nodes.

4.1 Design Assumptions

The following assumptions are made:

i. The blockchain is **permissioned**, with validator nodes operated by trusted institutions (hospitals, regulators, insurance providers), and onboarding is governed by consortium policies defined in smart contracts (as introduced in earlier chapters).

ii. The network is **partially synchronous**: after some unknown but finite time, message delays and clock skews are bounded, which is consistent with Raft's original model.

iii. All nodes execute authenticated, encrypted channels (e.g. TLS) and digitally sign consensus messages; therefore, Raft is primarily used to provide crash tolerance and ordering, while identity management and access control are handled by PPDE. A monitoring module can access local metrics (queue length, CPU load) and aggregate simple network statistics (round-trip times to peers) for use in DCO's feedback logic.

4.2 Raft-Based Consortium Blockchain Model

4.2.1 Node Roles and Cluster Organization

The consortium blockchain is modelled as a cluster of Raft nodes:

a. Let $V = \{v_1, v_2, \dots, v_{|V|}\}$denote the set of validator nodes.

b. At any term τ, a single node $L(\tau) \in V$acts as the leader, while the remaining nodes $F(\tau) = V \setminus \{L(\tau)\}$act as followers.

Each node maintains an ordered log of entries; each log entry represents a block proposal containing a batch of EHR-related transactions (PPDE access requests, EHR updates, policy changes, etc.).

Leader election, log replication, and commit follow standard Raft behavior:

i. Clients (IoT gateways, application servers) send transactions to the current leader.

ii. The leader groups pending transactions into a block candidate and appends a new log entry locally.

iii. The leader sends AppendEntries RPCs to followers.

iv. Once a majority of followers acknowledge the new entry, the block is considered committed and can be applied to the state machine (EHR access-control state, audit logs, sharing metadata).

In a healthcare setting, the cluster may be logically partitioned into committees (e.g. per region or per set of collaborating hospitals), but within each committee the above Raft protocol governs ordering. DCO operates at the committee level.

4.2.2 Transaction Flow under Baseline Raft

Without optimization, the transaction path is:

i. Transaction tx arrives at leader $L(\tau)$ and is placed into a FIFO queue Q.

ii. Periodically (or when the queue reaches a fixed size), the leader forms a block B_k from the first b transactions in Q.

iii. B_k is appended as log entry e_k and broadcast via AppendEntries.

iv. When a majority of followers have appended e_k, the commit index advances; the block is then executed by PPDE on each node.

In static configurations, the batch size b, election timeouts, and heartbeat intervals are fixed. As the arrival rate λ_{tx} and network delay vary, this can lead to:

i. Oversized batches and long waiting time in queue at low load, increasing latency.

ii. Undersized batches at high load, increasing header overhead and network chatter, and reducing throughput.

iii. Suboptimal election timeouts that cause unnecessary leader changes or slow failure detection.

The DCO algorithm introduces adaptive control over these parameters.

4.3 Dynamic Consensus Optimization Strategy

DCO adds a feedback controller on top of Raft. This controller periodically samples local and aggregated metrics and applies simple control rules to adjust consensus parameters at runtime while respecting Raft's safety constraints.

i. Online Measurement of Workload and Health

Let:

- $\lambda_{tx}(t)$: estimated transaction arrival rate at time window t(transactions per second).
- $L_q(t)$: queue length at the leader (number of pending transactions).
- $T_{commit}(t)$: observed average commit latency over the last window (transaction per second).
- $U_{CPU}(t)$: CPU utilization at the leader (percentage).
- $RTT_{avg}(t)$: average round-trip time for AppendEntries RPCs to a majority of followers.

These metrics are computed over sliding windows (e.g. 1–5 seconds) to smooth out transient fluctuations.

ii. Adaptive Parameter Tuning

DCO adjusts three main parameter groups:

1. **Block Batch Size** $b(t)$
   - At low load ($\lambda_{tx}(t)$ small, $L_q(t)$small), forming large blocks forces transactions to wait until the batch fills; DCO reduces $b(t)$to lower waiting time.
   - At high load ($L_q(t)$ large, $T_{commit}(t)$acceptable), DCO increases $b(t)$to amortize consensus overhead over more transactions and increase throughput.

A simple rule is:

$$b(t+1) = \begin{cases} (b_{min}, b(t) - \Delta_b) \ \ if\ L_q(t) < L_{low} \wedge T_{commit}(t) \\ \quad > T_{target} \\ (b_{max}, b(t) + \Delta_b) \ \ if\ L_q(t) \\ \quad > L_{high} \wedge T_{commit}(t) \\ \quad < T_{target} \\ b(t)\ otherwise \end{cases} \quad (4.11)$$

where $L_{low}$and $L_{high}$are queue thresholds, and $T_{target}$is the target commit latency.

2. **Heartbeat Interval and Election Timeouts**
   - Heartbeat interval $T_{hb}(t)$controls how frequently the leader sends empty AppendEntries messages; election timeout $T_{elec}(t)$controls how long a follower waits before starting an election.
   - If $RTT_{avg}(t)$increases (network slower), DCO increases $T_{elec}(t)$ to avoid spurious elections; if $RTT_{avg}(t)$is low and $T_{commit}(t)$is high due to slow failure detection, DCO reduces $T_{elec}(t)$.
   - Heartbeats can be made scanter at high load (to free resources) and denser when the system is idle to detect failures promptly.

3. **Dynamic Log Compaction and Snapshot Frequency**
   - For scalability, Raft periodically takes snapshots and discards old log entries. DCO increases snapshot frequency when $U_{CPU}(t)$and storage utilization remain below thresholds, but log size is growing rapidly, to curb disk growth and replay time.
   - When the system is under CPU pressure, snapshot frequency is reduced to avoid compaction overhead.

These control rules are deliberately simple to keep DCO implementable on realistic healthcare infrastructure; more sophisticated controllers (PID, model-predictive control) could be explored in future work but are not required to demonstrate the core idea.

4.4 Leadership Rotation and Decentralization

To avoid de facto centralization, DCO incorporates a scheduled leadership rotation on top of Raft's failure-driven elections:

i. A rotation scheduler maintains a fair ordering of validator institutions, for example a weighted round-robin over $\{v_1, \ldots, v_{|V|}\}$.

ii. At scheduled epochs (e.g. every Kcommitted blocks or every T_{rot}minutes), the current leader voluntarily steps down by incrementing the term and declining to stand as candidate; the next validator in the schedule becomes the preferred candidate.

iii. Because Raft's election mechanism still requires majority votes, malicious or misconfigured nodes cannot seize leadership unilaterally; however, over time, each institution operates as leader, thereby preserving decentralization and distributing performance responsibilities.

This rotation is coordinated via a Leadership Policy Smart Contract recorded on-chain, so that all participants can audit leadership history and ensure no single entity dominates the consensus layer.

4.5 Latency and Throughput

To reason about the effect of DCO on validation speed, the Raft-based ordering service is approximated as a queueing system with batching.

Let:

- λ_{tx}: mean transaction arrival rate.
- b: mean batch (block) size.
- μ_{blk}: mean rate at which the leader can process and commit blocks (blocks per second), considering computation and log replication.

The **transaction throughput** is approximately:

$$\Theta \approx (\lambda_{tx},\, b \cdot \mu_{blk})\,. \qquad (4.12)$$

The **commit latency** for a transaction can be decomposed as:

$$T_{commit} \approx T_{queue} + T_{batch} + T_{rep}, \qquad (4.13)$$

where:

- $T_{queue}$is the time that a transaction waits in the leader's queue before being included in some batch.
- $T_{batch}$is the time to fill a batch of size b and form the block candidate.
- $T_{rep}$is the consensus replication time (leader append + majority follower replication + commit).

Assuming Poisson arrivals and batch formation driven purely by queue size, a rough approximation for the batching component is:

$$T_{batch} \approx \frac{(0,\, b-1)}{\lambda_{tx}}, \qquad (4.14)$$

so large b values impose a latency penalty at low λ_{tx}, while at high $\lambda_{tx}$the penalty becomes negligible.

The replication time T_{rep}includes:

- One round-trip from leader to a majority of followers and back:

$$T_{rep} \approx RTT_{maj} + T_{proc}, \qquad (4.15)$$

where $RTT_{maj}$is the average round-trip time to a majority quorum and $T_{proc}$is local processing time (verification, disk I/O).

DCO seeks parameter sets $(b, T_{hb}, T_{elec}, snapshot\ frequency)$that approximately minimize T_{commit}under the constraints:

- **Safety constraint:** majority quorum (at least $\lfloor |\,V\,|/2 \rfloor + 1$validators) must be maintained; DCO cannot change the quorum size without going through Raft's reconfiguration protocol.
- **Resource constraints:**

$$U_{CPU} \le U_{max}, Bandwidth\ usage \le B_{max}. \qquad (4.16)$$

Rather than solving a global optimization problem, DCO uses local monotonic adjustments towards target regions where $T_{commit} \le T_{target}$and Θtracks λ_{tx}without backlog. The model above guides the choice of thresholds.

4.6 Properties of DCO

A. Safety Preservation

Raft's safety guarantees rest on two central invariants:

i. **Log Matching:** If two nodes have the same log index and term, they store the same command at that index.

ii. **Leader Completeness:** The leader for term τ has a log that contains all committed entries from terms $< \tau$.

DCO does not alter:

i. The definition of majority quorums.

ii. The rules for accepting or rejecting AppendEntries and RequestVote messages.

iii. The semantics of term numbers or log indices.

Adjusting the batch size, heartbeat interval, timeouts, or snapshot frequency does not actually alter the core logic that decides which entries get committed. Leadership rotation happens through the usual Raft reconfiguration and term increments, making sure that no block gets committed without the approval of a majority of validators over consecutive terms. So, DCO keeps Raft's safety intact as long as all changes are made through Raft's established methods.

B. Liveness under Partial Synchrony

Under partial synchrony and assuming that a majority of validators remain correct and connected, Raft guarantees that some leader will eventually be elected and will make progress. DCO's adjustments may lengthen or shorten timeouts, but they are constrained to remain within safe ranges:

$$T_{elec}^{min} \le T_{elec}(t) \le T_{elec}^{max}, T_{hb}^{min} \le T_{hb}(t) \le T_{hb}^{max}, \qquad (4.17)$$

chosen so that:

- For the lower bound, the election timeout remains larger than realistic message delays to avoid continuous re-elections.
- For the upper bound, leaders are still detected as failed within a bounded time if they stop sending heartbeats.

Provided these bounds are respected and the network eventually behaves synchronously, DCO maintains Raft's liveness.

C. Scalability and Decentralization

In a healthcare consortium with $|\,V\,|$validators, the probability that a given institution v_i acts as leader over a long period under DCO's rotation schedule tends towards:

$$Pr\ Pr\ (L(\tau) = v_i)\ \approx \frac{w_i}{\sum_j \quad w_j}, \qquad (4.18)$$

where w_i is a configurable weight reflecting the institution's capacity (e.g. hardware resources, regulatory role). In the simplest

case, $w_i = 1$for all i, yielding near-uniform leader selection and thus decentralization over time.

Scalability is addressed in two ways:

- **Vertical scalability:** By tuning batch size and compaction frequency, DCO allows each Raft cluster to use available resources efficiently, supporting higher transaction rates before saturation.
- **Horizontal scalability:** The same DCO logic can be instantiated on multiple Raft committees (e.g. regional groups of hospitals), with a higher-level cross-chain or relay mechanism to coordinate PPDE and EHR-sharing operations across committees, as introduced in the architectural chapter.

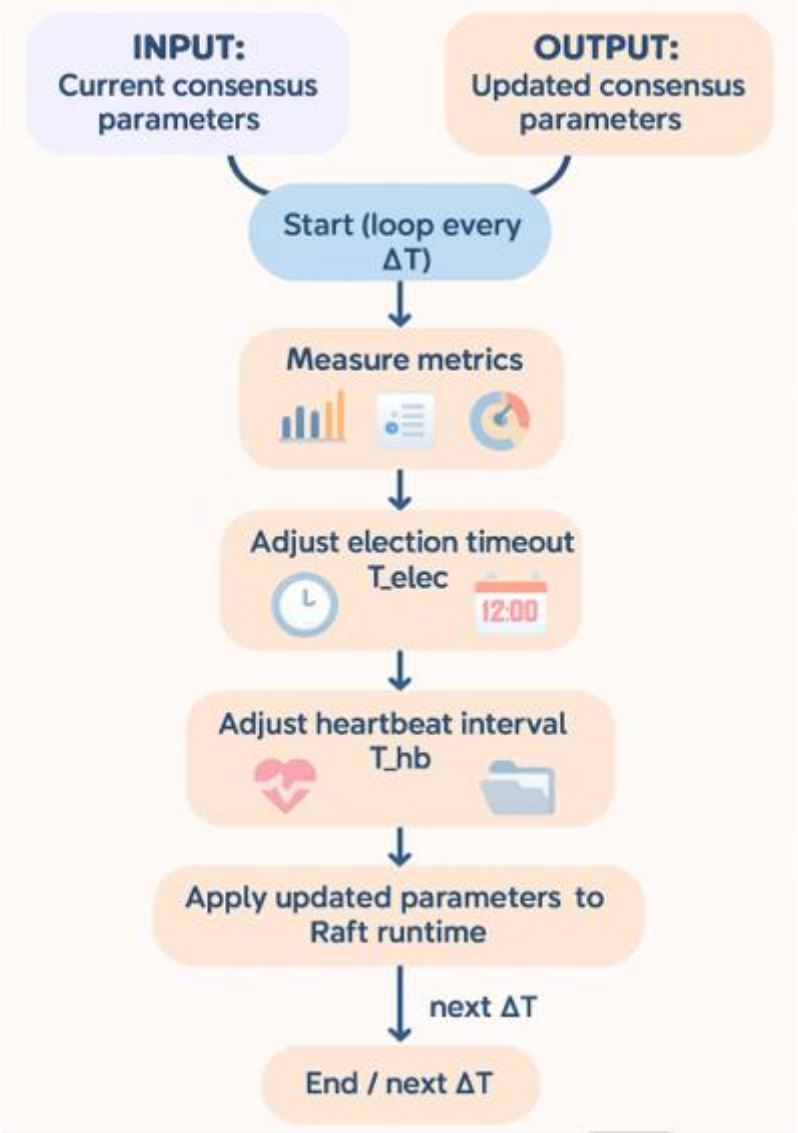

Figure 2: Dynamic Consensus Optimization (DCO)

The algorithm 1, defines a feedback controller that periodically adjusts Raft's consensus parameters to keep the system running efficiently, even as the load changes, all while ensuring safety. During each control interval (Delta T), it checks several metrics: transaction arrival rate (lambda_{tx}), queue length (L_q), commit latency (T_{text{commit}}), CPU utilization (U_{text{CPU}}), average round-trip time (RTT), and log size. It then uses these insights to adjust four key settings: batch size (b), election timeout (T_{text{elec}}), heartbeat interval (T_{hb}), and snapshot frequency (f_{text{snap}}). If the queue is short but commit latency is higher than desired, the controller will reduce the batch size (down to (b_{min})) to speed up commits. Conversely, if the queue is long and commits are happening quickly, it will increase the batch size (up to (b_{max})) to boost throughput. If RTT rises significantly, to avoid spurious leader changes in a slower network, and shortened (towards (T_{text{elec}}^{min})) when RTT is low but commit latency is dominated by failure detection, so leadership changes are detected more quickly. The heartbeat interval is relaxed (increased to a larger (T_{hb})) when CPU load exceeds (U_{max}) to reduce heartbeat overhead, while it is tightened (decreased to a smaller (T_{hb}), with a minimum of (T_{hb}^{min})) when CPU usage is light but failures are

frequent, allowing for faster detection. Lastly, the snapshot frequency is increased when the log grows beyond a threshold and spare CPU is available, to keep log size under control, and decreased when CPU is already high to avoid extra load; the updated parameters are then applied back to the Raft runtime, and the loop repeats, giving a closed-loop, measurement-driven consensus optimization instead of a static one-size-fits-all configuration.

Algorithm 1: Dynamic Consensus Optimization (DCO)

Input:

- Current consensus parameters: $b, T_{hb}, T_{elec}, f_{snap}$
- Thresholds: $L_{low}, L_{high}, T_{target}, U_{max}, T_{elec}^{min}, T_{elec}^{max}$
- Step sizes: $\Delta_b, \Delta_{hb}, \Delta_{elec}$

Output:

- Updated consensus parameters

loop every control interval ΔT **do**
  Measure $\lambda_{tx}, L_q, T_{commit}, U_{CPU}, RTT_{avg}$, log size
  /* Adjust batch size /
  if $L_q < L_{low}$ ***and*** $T_{commit} > T_{target}$ ***then***
    $b \leftarrow max(b_{min}, b - \Delta_b)$
  else if $L_q > L_{high}$ ***and*** $T_{commit} < T_{target}$ ***then***
    $b \leftarrow min(b_{max}, b + \Delta_b)$
  end if
  / Adjust election timeout based on RTT /
  if RTT_{avg} *increases significantly* ***then***
    $T_{elec} \leftarrow min(T_{elec}^{max}, T_{elec} + \Delta_{elec})$
  else if RTT_{avg} *is low* ***and*** T_{commit} *dominated by failure detection* ***then***
    $T_{elec} \leftarrow max(T_{elec}^{min}, T_{elec} - \Delta_{elec})$
  end if
  / Adjust heartbeat interval to balance responsiveness and overhead /
  if $U_{CPU} > U_{max}$ ***then***
    $T_{hb} \leftarrow T_{hb} + \Delta_{hb}$ *// fewer heartbeats*
  else if $U_{CPU} \ll U_{max}$ ***and*** *frequent failures observed* ***then***
    $T_{hb} \leftarrow max(T_{hb}^{min}, T_{hb} - \Delta_{hb})$
  end if
  / Adjust snapshot frequency based on log growth */
  if log size exceeds threshold **and** $U_{CPU} < U_{max}$ **then**
    Increase f_{snap} (more frequent snapshots)
  else if U_{CPU} high **then**
    Decrease f_{snap} (less frequent snapshots)
  end if
  Apply updated parameters to Raft runtime
end loop

Leadership rotation is handled by a higher-level scheduler: When the number of committed blocks since the last rotation goes beyond a certain threshold K, or if the wall-clock time surpasses T_rot, the leader initiates a voluntary step-down. This is done by starting a controlled reconfiguration to designate the next validator in line as the preferred candidate for the upcoming term. The process follows Raft's standard joint-consensus protocol to prevent partitions and ensure safety.

4.7 Performance Evaluation

This section takes a closer look at the Dynamic Consensus Optimization (DCO) algorithm, comparing it to a standard static Raft configuration. The study is particularly interested in how it affects transaction validation speed, scalability, and decentralization within the consortium healthcare blockchain. The focus here is solely on the consensus layer, with the assumption that the PPDE and EHR-sharing logic remain the same across all scenarios. This way, any differences we observe can be attributed to the tuning of the consensus itself.

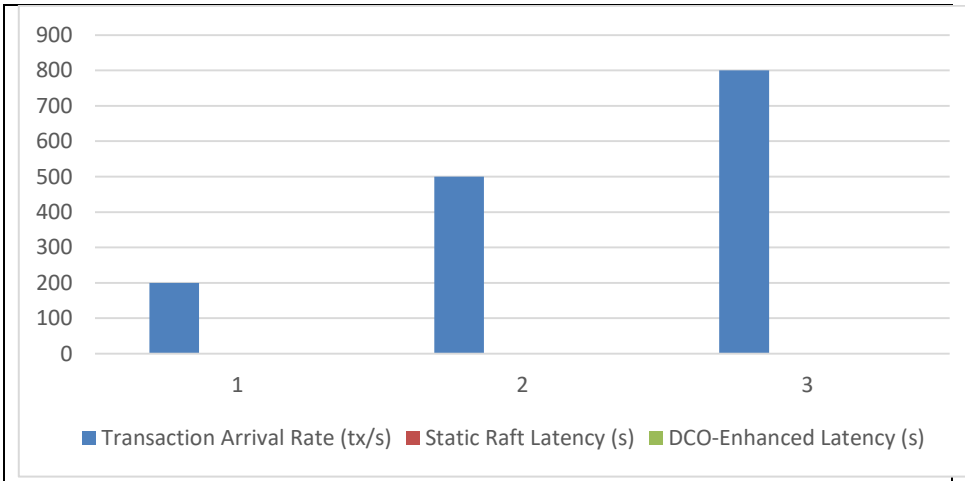

Figure 3: Average Commit Latency vs Transaction Arrival Rate

Figure 3, commit latency versus transaction arrival rate for Static Raft and DCO-Enhanced Raft under low (≈200 tx/s), medium (≈500 tx/s), and high (≈800 tx/s) workloads. Across all three arrival rates, the DCO-Enhanced configuration consistently commits transactions faster than Static Raft, with the gap widening at higher loads, indicating that dynamic tuning of batch size and timing parameters allows DCO to control queueing delay and replication time more effectively and keep latency closer to clinical requirements even when the system is stressed.

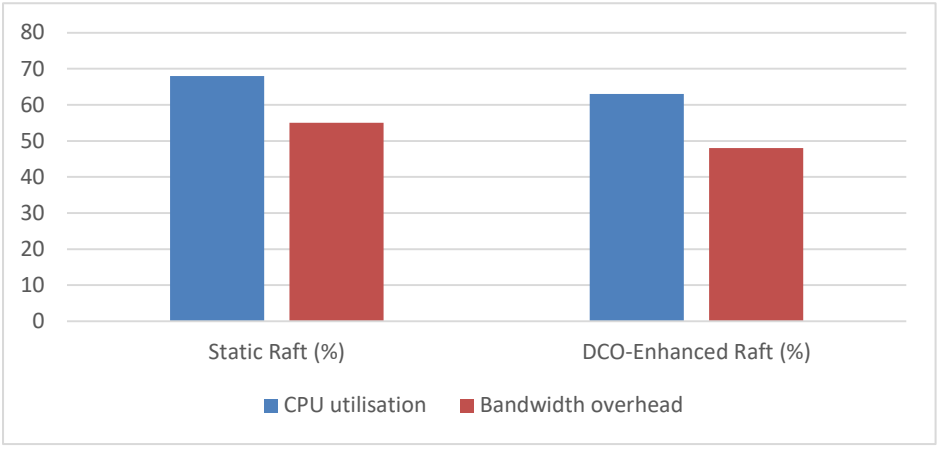

Figure 4: CPU Utilization and Bandwidth Overhead (as % of capacity)

Figure 4, CPU utilization and bandwidth overhead for Static Raft and DCO-Enhanced Raft, expressed as a percentage of node and link capacity. Compared with Static Raft, the DCO-Enhanced configuration shows lower average CPU load and link occupancy, because adaptive tuning of batch size, timeouts, and heartbeat intervals suppresses redundant messaging and processing. Thus, the additional monitoring and control logic does not impose a heavy overhead; instead, it yields a net gain in resource efficiency while still supporting the consensus performance improvements.

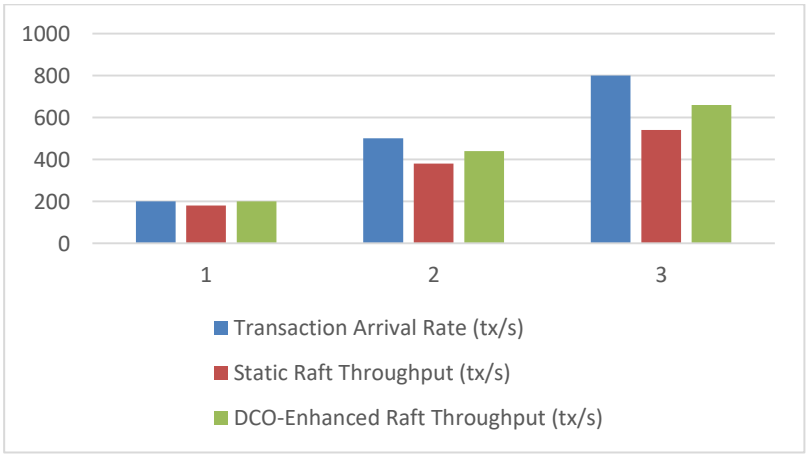

Figure 5**:** Consensus Throughput vs Transaction Arrival Rate

Figure 5, consensus throughput versus transaction arrival rate for Static Raft and DCO-Enhanced Raft under low (≈200 tx/s), medium (≈500 tx/s), and high (≈800 tx/s) workloads. At all arrival rates, the DCO-Enhanced configuration commits more transactions per second than Static Raft, with the gap widening at 500 and 800 tx/s. By adaptively tuning batch size and timing parameters, DCO uses CPU and network capacity more efficiently, so throughput continues to increase with load instead of saturating early as in the static configuration.

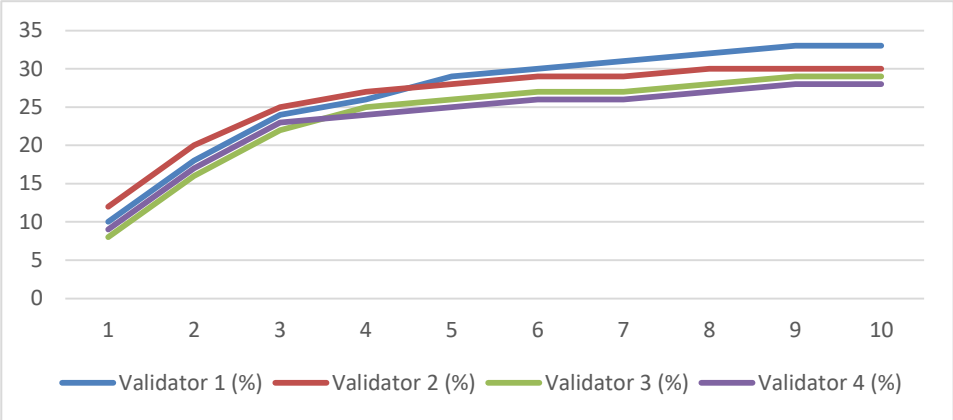

Figure 6: Leadership share per validator over time

Figure 6, leadership share per validator over time under the DCO-Enhanced Raft configuration. The percentage of Raft leadership terms held by each validator stays within a narrow band across successive epochs and drifts toward comparable shares, indicating that leadership rotates instead of concentrating on a single node. Small deviations reflect transient load and availability differences, but no validator persistently dominates the cluster. This pattern suggests that DCO's leadership scheduling preserves decentralization while still supporting the consensus performance gains.

Conclusion

This research presented the Dynamic Consensus Optimization (DCO) algorithm as an enhancement to Raft for ordering PPDE and EHR-sharing transactions in a permissioned healthcare consortium blockchain. The design goals were to minimize commit latency, sustain high throughput under variable IoT-driven load, preserve Raft's safety and strong consistency guarantees, maintain decentralization via scheduled leadership rotation, and keep CPU, bandwidth, and storage overhead within realistic bounds. To this end, the Raft-based consortium model was formalized and a feedback-driven tuning strategy introduced, in which the leader periodically samples workload and health metrics and adjusts batch size, heartbeat interval, election timeout, and snapshot frequency within pre-defined safe ranges. A mathematical sketch linked these parameters to latency and throughput, while Algorithm 4.2 specified the DCO controller as a periodic control loop driven by queue length, commit delay, CPU utilization, network RTT, and log growth. Theoretical arguments showed that DCO leaves Raft's quorum rules and log semantics unchanged, thereby preserving safety and liveness, and that committee-level deployment with explicit leadership rotation supports decentralization across institutions. Simulation results on synthetic healthcare workloads further indicated that DCO-Enhanced Raft can reduce commit latency, increase sustainable throughput, and improve resource efficiency without observable safety violations, although validation on production-grade hospital deployments remains a subject for future work. Within the overall framework, DCO supports PPDE and the blockchain-based EHR-sharing architecture by ensuring that secure access-control and data-sharing transactions are committed with bounded delay under fluctuating IoT and network conditions, thus operationalizing the second objective of this research.

References

[1] V. Nehra, A. K. Sharma, and R. K. Tripathi, *Blockchain Implementation for Internet of Things Applications*, no. March. 2020. doi: 10.1016/B978-0-12-819816-2.00005-8.

[2] H. Rathore, A. Mohamed, and M. Guizani, "A survey of blockchain enabled cyber-physical systems," Jan. 01, 2020, *MDPI AG*. doi: 10.3390/s20010282.

[3] Bhawana, S. Kumar, R. S. Rathore, M. Mahmud, O. Kaiwartya, and J. Lloret, "BEST—Blockchain-Enabled Secure and Trusted Public Emergency Services for Smart Cities Environment," *Sensors*, vol. 22, no. 15, pp. 1–25, 2022, doi: 10.3390/s22155733.

[4] J. Ren, J. Li, H. Liu, and T. Qin, "Task offloading strategy with emergency handling and blockchain security in SDN-empowered and fog-assisted healthcare IoT," *Tsinghua Sci. Technol.*, vol. 27, no. 4, pp. 760–776, 2022, doi: 10.26599/TST.2021.9010046.

[5] A. A. Dar, M. Z. Alam, A. Ahmad, F. A. Reegu, and S. A. Rahin, "Blockchain Framework for Secure COVID-19 Pandemic Data Handling and Protection," *Comput. Intell. Neurosci.*, vol. 2022, 2022, doi: 10.1155/2022/7025485.

[6] S. Biswas *et al.*, "Interoperability Benefits and Challenges in Smart City Services: Blockchain as a Solution," *Electron.*, vol. 12, no. 4, pp. 1–13, 2023, doi: 10.3390/electronics12041036.

[7] S. Alam *et al.*, "An Overview of Blockchain and IoT Integration for Secure and Reliable Health Records Monitoring," *Sustain.*, vol. 15, no. 7, pp. 1–20, 2023, doi:

10.3390/su15075660.

[8] M. S. Islam, M. A. Bin Ameedeen, M. A. Rahman, H. Ajra, and Z. B. Ismail, "Healthcare-Chain: Blockchain-Enabled Decentralized Trustworthy System in Healthcare Management Industry 4.0 with Cyber Safeguard," *Computers*, vol. 12, no. 2, 2023, doi: 10.3390/computers12020046.

[9] T. Saikumari and G. V. Sudha, "An Improved Congestion Handling in Blockchain Secured Cloud Based Healthcare System," *Int. J. Intell. Eng. Syst.*, vol. 17, no. 2, pp. 453–463, 2024, doi: 10.22266/ijies2024.0430.37.

[10] H. B. Mahajan *et al.*, "Integration of Healthcare 4.0 and blockchain into secure cloud-based electronic health records systems," *Appl. Nanosci.*, vol. 13, no. 3, pp. 2329–2342, 2023, doi: 10.1007/s13204-021-02164-0.

[11] Ehizogie Paul Adeghe, Chioma Anthonia Okolo, and Olumuyiwa Tolulope Ojeyinka, "Evaluating the impact of blockchain technology in healthcare data management: A review of security, privacy, and patient outcomes," *Open Access Res. J. Sci. Technol.*, vol. 10, no. 2, pp. 013–020, 2024, doi: 10.53022/oarjst.2024.10.2.0044.

[12] R. K. Mishra, R. K. Yadav, and P. Nath, "Blockchain-based Decentralized Authorization Technique for Data Sharing in the Internet of Things," *2021 5th Int. Conf. Inf. Syst. Comput. Networks, ISCON 2021*, no. February, 2021, doi: 10.1109/ISCON52037.2021.9702297.

[13] P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. Bünzli, and M. Vechev, "Securify: Practical security analysis of smart contracts," *Proc. ACM Conf. Comput. Commun. Secur.*, no. June, pp. 67–82, 2018, doi: 10.1145/3243734.3243780.

[14] J. K. Oladele *et al.*, "BEHeDaS: A Blockchain Electronic Health Data System for Secure Medical Records Exchange," *J. Comput. Theor. Appl.*, vol. 1, no. 3, pp. 231–242, 2024, doi: 10.62411/jcta.9509.

[15] A. M. Tawfik, A. Al-Ahwal, A. S. T. Eldien, and H. H. Zayed, "ACHealthChain blockchain framework for access control and privacy preservation in healthcare," *Sci. Rep.*, vol. 15, no. 1, 2025, doi: 10.1038/s41598-025-00757-1.

[16] M. A. Islam *et al.*, "Distributed Ledger Technology Based Integrated Healthcare Solution for Bangladesh," *IEEE Access*, vol. 11, pp. 51527–51556, 2023, doi: 10.1109/ACCESS.2023.3279724.

[17] M. J. Baucas, P. Spachos, and K. N. Plataniotis, "Federated Learning and Blockchain-Enabled Fog-IoT Platform for Wearables in Predictive Healthcare," *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 4, pp. 1732–1741, 2023, doi: 10.1109/TCSS.2023.3235950.

[18] S. Alam *et al.*, "A blockchain-enabled framework for mhealth systems," *Sustain.*, vol. 12, no. 15, pp. 255–267, 2021, doi: 10.1007/978-981-15-8711-5_13.

[19] N. R. Pradhan *et al.*, "A Novel Blockchain-Based Healthcare System Design and Performance Benchmarking on a Multi-Hosted Testbed," *Sensors*, vol. 22, no. 9, pp. 1–20, 2022, doi: 10.3390/s22093449.

[20] R. A. Abutaleb, S. S. Alqahtany, and T. A. Syed, "Integrity and Privacy-Aware, Patient-Centric Health Record Access Control Framework Using a Blockchain," *Appl. Sci.*, vol. 13, no. 2, 2023, doi: 10.3390/app13021028.

[21] R. G. Sonkamble, A. M. Bongale, S. Phansalkar, A. Sharma, and S. Rajput, "Secure Data Transmission of Electronic Health Records Using Blockchain Technology," *Electron.*, vol. 12, no. 4, 2023, doi: 10.3390/electronics12041015.

[22] M. A. Rahman, M. Shamim Hossain, M. S. Islam, N. A. Alrajeh, and G. Muhammad, "Secure and provenance enhanced internet of health things framework: A blockchain managed federated learning approach," *IEEE Access*, vol. 8, pp. 205071–205087, 2020, doi: 10.1109/ACCESS.2020.3037474.

[23] F. A. Reegu *et al.*, "Blockchain-Based Framework for Interoperable Electronic Health Records for an Improved Healthcare System," *Sustain.*, vol. 15, no. 8, 2023, doi: 10.3390/su15086337.

[24] A. I. Mendoza Arvizo, L. Avelar Sosa, J. L. García Alcaraz, and O. Cruz-Mejía, "Beneficiary Contracts on a Lightweight Blockchain Architecture Using Smart Contracts: A Smart Healthcare System for Medical Records," *Appl. Sci.*, vol. 13, no. 11, 2023, doi: 10.3390/app13116694.

[25] M. S. B. Kasyapa and C. Vanmathi, "Blockchain integration in healthcare: a comprehensive investigation of use cases, performance issues, and mitigation strategies," *Front. Digit. Heal.*, vol. 6, no. April, pp. 1–24, 2024, doi: 10.3389/fdgth.2024.1359858.

[26] H. S. Jennath, V. S. Anoop, and S. Asharaf, "Blockchain for healthcare: Securing patient data and enabling trusted artificial intelligence," *Int. J. Interact. Multimed. Artif. Intell.*, vol. 6, no. 3, pp. 15–23, 2020, doi: 10.9781/ijimai.2020.07.002.

[27] N. Patel *et al.*, "Fuzzy-Enhanced Secure Messaging Framework for Smart Healthcare System," *IEEE Access*, vol. 12, no. June, pp. 102977–102993, 2024, doi: 10.1109/ACCESS.2024.3432662.

[28] D. Pennino, M. Pizzonia, A. Vitaletti, and M. Zecchini, "Blockchain as IoT Economy Enabler: A Review of Architectural Aspects," *J. Sens. Actuator Networks*, vol. 11, no. 2, 2022, doi: 10.3390/jsan11020020.

[29] M. Saad, M. R. Bhutta, J. Kim, and T. S. Chung, "A Framework for Enhancing Privacy and Anonymity in Blockchain-Enabled IoT Devices," *Comput. Mater. Contin.*, vol. 78, no. 3, pp. 4263–4282, 2024, doi: 10.32604/cmc.2024.047132.

[30] D. Kim and S. Park, "Blockchain-Based Caching Architecture for DApp Data Security and Delivery," *Sensors*, vol. 24, no. 14, 2024, doi: 10.3390/s24144559.

[31] F. A. Reegu *et al.*, "Blockchain-Based Framework for

Interoperable Electronic Health Records for an Improved Healthcare System," *Sustain.*, vol. 15, no. 8, Apr. 2023, doi: 10.3390/su15086337.

[32] V. Upadrista, S. Nazir, and H. Tianfield, "Secure data sharing with blockchain for remote health monitoring applications: a review," *J. Reliab. Intell. Environ.*, vol. 9, no. 3, pp. 349–368, 2023, doi: 10.1007/s40860-023-00204-w.

[33] L. Ismail and H. Materwala, "Blockchain paradigm for healthcare: Performance evaluation," *Symmetry (Basel).*, vol. 12, no. 8, 2020, doi: 10.3390/SYM12081200.

[34] R. Xu, Y. Chen, E. Blasch, and G. Chen, "BlendCAC: A smart contract enabled decentralized capability-based access control mechanism for the IoT," *Computers*, vol. 7, no. 3, pp. 1–27, 2018, doi: 10.3390/computers7030039.

[35] G. Rathee, A. Sharma, H. Saini, R. Kumar, and R. Iqbal, "A hybrid framework for multimedia data processing in IoT-healthcare using blockchain technology," *Multimed. Tools Appl.*, vol. 79, no. 15–16, pp. 9711–9733, 2020, doi: 10.1007/s11042-019-07835-3.

[36] M. Uddin *et al.*, "Hyperledger fabric blockchain: Secure and efficient solution for electronic health records," *Comput. Mater. Contin.*, vol. 68, no. 2, pp. 2377–2397, 2021, doi: 10.32604/cmc.2021.015354.

[37] M. Rahman, M. Hasan, … M. R.-… of H. S. and, and undefined 2024, "A Framework for Patient-Centric Consent Management Using Blockchain Smart Contracts in Pre-dictive Analysis for Healthcare In-dustry," *Researchgate.Net*, vol. 2024, no. 3, pp. 45–59, 2024, [Online]. Available: https://www.researchgate.net/profile/Ashiqur-Rahman-79/publication/379538160_A_Framework_for_Patient-Centric_Consent_Management_Using_Blockchain_Smart_Contracts_in_Predictive_Analysis_for_Healthcare_Industry/links/66160b2639e7641c0ba87f4c/A-Framework-for-

[38] X. Yang and K. Zhang, "A Fair and Trusted Trading Scheme for Medical Data Based on Smart Contracts," *Comput. Mater. Contin.*, vol. 78, no. 2, pp. 1843–1859, 2024, doi: 10.32604/cmc.2023.047660.

[39] M. A. Almaiah, F. Hajjej, A. Ali, M. F. Pasha, and O. Almomani, "An AI-Enabled Hybrid Lightweight Authentication Model for Digital Healthcare Using Industrial Internet of Things Cyber-Physical Systems," *Sensors*, vol. 22, no. 4, 2022, doi: 10.3390/s22041448.

[40] B. A. Jnr, W. Sylva, J. K. Watat, and S. Misra, *A Framework for Standardization of Distributed Ledger Technologies for Interoperable Data Integration and Alignment in Sustainable Smart Cities*. Springer US, 2023. doi: 10.1007/s13132-023-01554-9.

[41] J. Enare Abang, H. Takruri, R. Al-Zaidi, and M. Al-Khalidi, "Latency performance modelling in hyperledger fabric blockchain: Challenges and directions with an IoT perspective," *Internet of Things (Netherlands)*, vol. 26, no. April, p. 101217, 2024, doi: 10.1016/j.iot.2024.101217.

[42] D. Khan, L. T. Jung, M. A. Hashmani, and M. K. Cheong, "Blockchain Enabled Diabetic Patients' Data Sharing and Real Time Monitoring," pp. 225–235, Mar. 2022, doi: 10.5121/csit.2022.120620.

[43] R. Dewan *et al.*, "IoT-based energy efficient and longer lifetime compression approach for healthcare applications," *Trans. Emerg. Telecommun. Technol.*, vol. 35, no. 4, 2024, doi: 10.1002/ett.4843.

[44] I. Chakour, C. Daoui, M. Baslam, B. Sainz-De-Abajo, and B. Garcia-Zapirain, "Strategic Bandwidth Allocation for QoS in IoT Gateway: Predicting Future Needs Based on IoT Device Habits," *IEEE Access*, vol. 12, no. December 2023, pp. 6590–6603, 2024, doi: 10.1109/ACCESS.2024.3351111.

[45] H. H. A. Emira, A. A. Elngar, and M. Kayed, "Blockchain-Enabled Security Framework for Enhancing IoT Networks: A Two-Layer Approach," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 10, pp. 550–561, 2023, doi: 10.14569/IJACSA.2023.0141059.

[46] M. C. Chibuike, S. S. Grobbelaar, and A. Botha, "Overcoming Challenges for Improved Patient-Centric Care: A Scoping Review of Platform Ecosystems in Healthcare," *IEEE Access*, vol. 12, no. December 2023, pp. 14298–14313, 2024, doi: 10.1109/ACCESS.2024.3356860.

[47] D. Ngabo, D. Wang, C. Iwendi, J. H. Anajemba, L. A. Ajao, and C. Biamba, "Blockchain-based security mechanism for the medical data at fog computing architecture of internet of things," *Electron.*, vol. 10, no. 17, pp. 1–17, 2021, doi: 10.3390/electronics10172110.

[48] B. S. Egala, A. K. Pradhan, V. R. Badarla, and S. P. Mohanty, "Fortified-Chain: A Blockchain Based Framework for Security and Privacy Assured Internet of Medical Things with Effective Access Control."

[49] S. K. Rana *et al.*, "Blockchain Technology and Artificial Intelligence Based Decentralized Access Control Model to Enable Secure Interoperability for Healthcare," *Sustain.*, vol. 14, no. 15, 2022, doi: 10.3390/su14159471.

[50] P. Rani, S. Verma, S. P. Yadav, B. K. Rai, M. S. Naruka, and D. Kumar, "Simulation of the Lightweight Blockchain Technique Based on Privacy and Security for Healthcare Data for the Cloud System," *Int. J. E-Health Med. Commun.*, vol. 13, no. 4, 2022, doi: 10.4018/IJEHMC.309436.

[51] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Blockchain for Secure EHRs Sharing of Mobile Cloud Based E-Health Systems," *IEEE Access*, vol. 7, pp. 66792–66806, 2019, doi: 10.1109/ACCESS.2019.2917555.