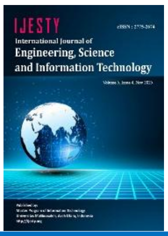
International Journal of Engineering, Science and Information Technology
Volume 5, No. 4 (2025) pp. 132-143
eISSN: 2775-2674
Website: http://ijesty.org/index.php/ijesty
DOI: https://doi.org/10.52088/ijesty.v5i4.928
Research Paper, Short Communication, Review, Technical Paper



Empirical Performance Analysis of Hyperledger Fabric Blockchain Network for Healthcare

Shampa Rani Das[1], NZ Jhanjhi[1]*, Farzeen Ashfaq[1], Husham M. Ahmed[2], Azeem Khan[3]

[1]School of Computer Science, Taylor's University, Subang Jaya, Malaysia
[2]College of Engineering, University of Technology Bahrain, Bahrain
[3]Faculty of Islamic Technology, Universiti Islam Sultan Sharif Ali, Brunei Darussalam

**Corresponding author Email: noorzaman.jhanjhi@taylors.edu.my*

The manuscript was received on 27 February 2025, revised on 15 May 2025, and accepted on 24 August 2025, date of publication 5 November 2025

Abstract

The most prevalent blockchain-enabled systems have several apparent advantages, but scaling remains a technical difficulty that results in performance deficiencies in latency and throughput. The foremost concern is that a thorough performance analysis is required to determine their viability and efficiency. The prospective impact of transaction delay on blockchain networks is an acute issue for e-healthcare-related services since it can jeopardize the patient's life safety. A benchmarking tool Hyperledger Caliper is utilized to measure performance parameters in the Hyperledger Fabric network. The effects of workload fluctuation in 6 rounds with up to 3000 Transactions Per Second (TPS) are demonstrated when four organizations are put up in the network. Significant findings include a noteworthy decrease of 27.11% in open latency and 26.27% in query latency, and an increase of 3.13% in query throughput and 3.44% in open throughput demonstrating enhancements in adaptability and operational efficiency over the recent existing approaches proposed. It demonstrates an ongoing increase in CPU and memory consumption, peaking at 5.49% and 528.23 MB for 3000 TPS, respectively. Inbound and outbound traffic indicate relatively even utilization, with variations falling within a moderate range.

Keywords: *Hyperledger Fabric Blockchain, Hyperledger Caliper, Network Performance, Throughput, Latency*.

1. Introduction

A blockchain database contains information in precise chunks and distributes it across a network node, preventing unwanted access to the data or files from ever happening [1]. Generally, each recorded fact would be split into blocks that are chronologically organized and time stamped. The immutability of data is yet another pertinent blockchain feature. A P2P [2] network enables direct communication between nodes, allowing them to share files, data, services, or any information without intermediaries, thereby increasing resilience. Hyperledger is a cross-industry open-source collaborative initiative intended to enhance distributed ledger technologies [3]. Hyperledger Fabric [4] is one of the most widely adopted frameworks within the Hyperledger ecosystem. It provides modular and flexible architecture for building enterprise-grade blockchain networks as depicted in Figure 1.

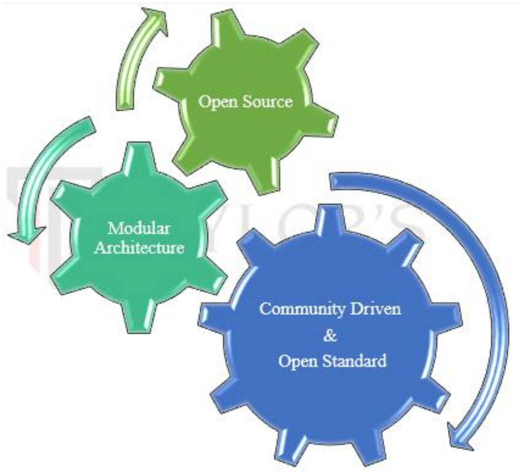


Fig 1. Aspects of Hyperledger Fabric

Healthcare 4.0 aspires to provide digital tools that can facilitate communication between diverse stakeholders and smooth information flow along the patient's path to wellness [5]. Services related to EHRs are very delay-sensitive [6]. Transaction latency is a crucial parameter for blockchain networks because it has the potential to have substantial impacts on task quality of service.
This research provides an empirical performance analysis of Hyperledger Fabric v2.5 in the context of healthcare, focusing on scalability, latency, and throughput under varying transaction loads. The improved performance metrics observed can be attributed to several key features. Enhanced chain code deployment and upgrade processes are reducing downtime and improving transaction throughput. Optimized resource allocation and workload distribution are leading to lower latency and higher throughput. Support for parallel execution of transactions is improving scalability under high transaction loads.
This research evaluates the network's performance using a realistic workload derived from the MIMIC-III dataset [27], which simulates real-world EHR transactions. We utilize Hyperledger Caliper to benchmark the network's performance, incorporating a realistic chaincode implementation that simulates EHR transactions, such as patient record updates, access control checks, and data sharing between organizations. Research contributions are as follows:

1. It led to a corresponding decrease in query latency of 26.27% TPS and an open latency of 27.11% TPS.
2. The enhancement in the query throughput is estimated to be 3.13%, while the open throughput is 3.44% transaction load is relatively stable, with no extreme spikes in resource consumption observed. System settings or specifications affect how effectively the Hyperledger Fabric blockchain network performs.

The relevant literature and the research methods are presented in Sections 2 and 3. The research findings are outlined in Section 4. Section 5 concludes the research outcome, including future direction.

Table 1. Notation Table

Abbreviation	Description
CA	Certificate Authority
ECert	Enrollment Certificate
EHR	Electronic Health Record
HF	Hyperledger Fabric
MSP	Membership Service Provider
PBFT	Practical Byzantine Fault Tolerant
TLS	Transport Layer Security
TPS	Transactions Per Second

2. Literature Review

Most recent comprehensive research appears to indicate that the main factor in the broader implementation of blockchain technology is its performance and scalability, specifically concerning latency and throughput [7]. The majority of blockchain storage expansion initiatives are designed [8] for public blockchain, yet publicly accessible blockchain scaling difficulties with catering to networks of immense scale with high transaction throughput, low latency, and security concerns. Many of the proposed blockchain-driven systems [9] have not been put to the test and are unregulated, posing a challenge for them to offer scalable services at a big scale with a high success rate. It is imperative to either improve the existing systems or offer workable alternatives to them [10].
Fabric is restricted to the approved users only within the network. It is particularly suited for the healthcare ecosystem where strict access control and privacy are essential. It facilitates the seamless upgrading of individual components without affecting the entire network, ensuring a high degree of flexibility, security [11, 12, 13], and maintainability throughout the lifecycle of deployment. It has private channels, which are predetermined messaging paths, to protect privacy. Chaincode [14] functions as the executable distributed code that participants on the network invoke to read from or write to the ledger. It offers a resilient framework that allows organizations to adapt endorsement policies following their specific business requirements and regulatory compliance needs. It enables the distribution of workloads across multiple nodes and facilitates parallel transaction processing. The network's support for channels and private data collection further enhances scalability by allowing selective data sharing among participants. Pluggable consensus algorithms endorsement policies provide flexibility in tailoring the network to specific scalability requirements, making it well-suited for diverse enterprise use cases where scalability is the utmost concern. Table 2 focuses on the theoretical aspects of consensus mechanisms and their practical implications in Hyperledger Fabric.

Table 2. Consensus Properties with Actual Implementation

Consensus Mechanism	Theoretical Properties	Practical Implications in Hyperledger Fabric
Solo	• Effective for testing purposes	• Not suitable for distributed production environments
Kafka	• Scalable, fault-tolerant	• Introduces complexity and dependency on external components
Raft	• Fault-tolerant	• May face scalability challenges in very large networks
Practical Byzantine Fault Tolerant (PBFT)	• High security	• High computational resource demands, complex implementation

Consensus Mechanism	Theoretical Properties	Practical Implications in Hyperledger Fabric
Byzantine Fault Tolerant	• It offers network integrity. • When transactions are validated then these are regarded as final.	• It involves navigating complexities related to node behavior and communication patterns. • It incurs higher computational resource demands.
Crash Fault Tolerant	• Efficient resource usage	• Reduced resistance to malicious behavior

The Membership Service Provider (MSP) manages digital identities within the network, including the creation, issuance, and revocation of cryptographic certificates. Participants are enrolled with an MSP, which validates their identities and assigns the necessary cryptographic material. The MSP, in conjunction with the Certificate Authority (CA), strengthens the entire security and integrity by ensuring that only authorized entities with valid identities can participate in the network. CA issues X509 certificate is portrayed in Figure 2 to authenticated and authorized entities.

.

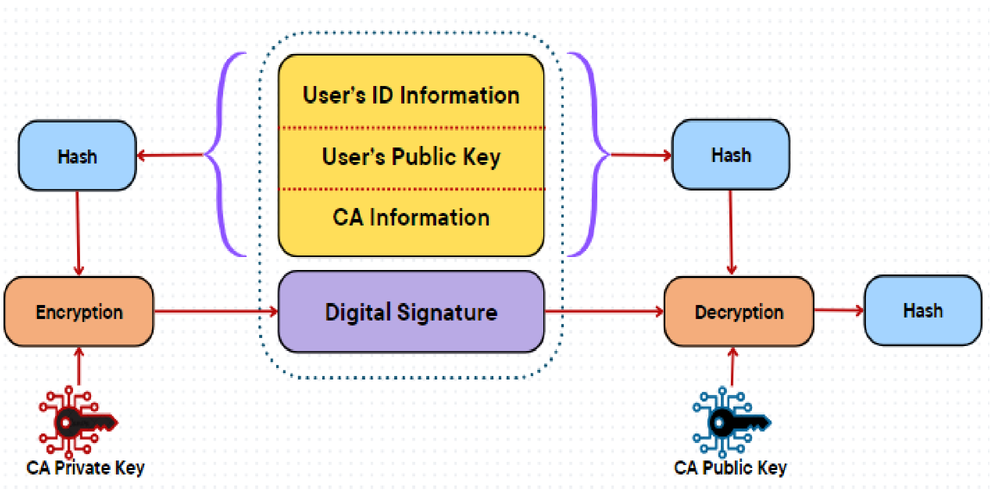


Fig 2. X509 Certificate Format

Figure 3 illustrates the Fabric network's transaction flow. There is no need for a mining procedure. It allows organizations to create permissioned networks with fine-grained access control and scalable transaction processing. There are two distinct types of nodes within the network: peer nodes and ordering nodes. Transaction batching and verification are the responsibilities of peer nodes. Ordering nodes generate and order the network's current history of happenings. Multiple transactions can be processed simultaneously with sufficient efficiency.

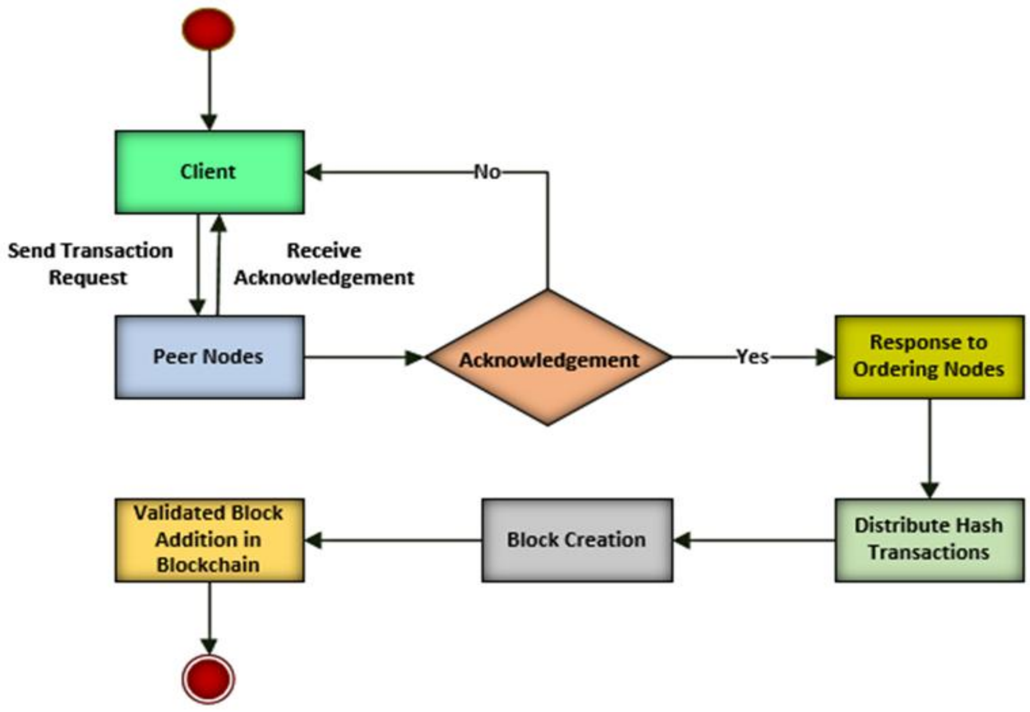


Fig 3. Fabric Network's Transaction Flow

Orderer nodes receive transaction proposals, organize them into blocks, and send the resulting blocks to peer nodes for commitment and validation. It ensures that transactions are consistent and in order throughout the network. It keeps the ledger state consistent by making sure that every transaction is carried out in the same sequence among all peers. Peer nodes are responsible for overseeing ledger copies and smart contracts, taking part in transaction procedures including consensus, validation, and endorsement, and carrying out transactions while updating the ledger and attending to client inquiries. They fall into three categories: anchor peers, committed peers, and endorsing peers. Each of these groups is involved in distinctly executing transactions and network functionality.
Table 2 presents the existing blockchain-powered proposed approaches for electronic health records (EHR) storage systems without considering network performance assessment. IoT applications can benefit from enhanced security using blockchain technology [15-19], which can come up with a few issues with latency and energy efficiency. A comprehensive network performance evaluation of widely accepted blockchain is required [20].

Table 3. Contributions of Research Papers in Healthcare

Authors	Year	Cited	Approached Platform	Contributions for Healthcare
Sonkamble et al. [21]	2023	9	Hyperledger Fabric	• Authors assess the efficacy of the proposed distributed patient-oriented medical records management structure. • It leverages secure password authentication-based key exchange. • It focuses mainly for regulating access.
Rajawat et al. [22]	2022	12	Blockchain and machine learning	• An EHR infrastructure is proposed. • It addresses data confidentiality and integrity issues. • It focuses on substantial clinical data sharing and advances patient-focused treatment through the separation of tailored information.
Pang et al. [23]	2022	25	Blockchain and cloud	• EHR sharing scheme is presented. • It ensures data integrity and restricted accessibility. • It utilizes cryptography and a node-state-checkable PBFT consensus mechanism. • It demonstrates heightened processing capacity and resilience.
Fatokun et al. [24]	2021	53	Ethereum	• A patient-focused EHR system is proposed. • It allows patients to manage their medical records. • It fosters interoperability between healthcare providers. • It articulates efficaciousness in confidentiality and security.
Rajput et al. [25]	2021	61	Hyperledger Fabric and Hyperledger Composer	• The healthcare administration infrastructure is proposed. • It performs better in aspects such as accessibility, privacy, and security.
Wang and Qin [26]	2021	20	Hyperledger Fabric	• A medical records exchange system is put forward. • It includes multilevel authorization in chaincode developed. • It focuses on improved interoperability and privacy protection.
Proposed Research	2024	-	Hyperledger Fabric v2.5 and cloud	• A healthcare network is provided. • It focuses on Hyperledger Fabric network performance varying transaction load. • The evaluated network performance is compared with prior research that reflects high throughput, low latency, and low resource consumption.

3. Methods

Transactions in Hyperledger Fabric are processed in stages: they are performed first, sorted by consensus, and then validated before being recorded in the ledger. The Fabric network architecture is illustrated in Figure 4. Scalability is enhanced and an immense number of inbound transactions can be processed promptly and efficiently. All the Fabric network's components are installed and distributed as Docker containers. Developers can build, deploy, update, manage, and execute their applications with the additional support of the open-source Docker project. Caliper is used to monitor the network's throughput, latency, and resource consumption.
The simulation PC configuration is as follows: Core 4 Threads CPU [Intel Core i5-@ 3.50 GHz - 3.90 GHz (3.90 GHz in overclocked mode) with 6MB shared L3 cache], 16 GB Memory, GPU with 8GB of memory, 256GB NVMe SSD, 1 Gbit/s network, and OS -Ubuntu 22.04 LTS.

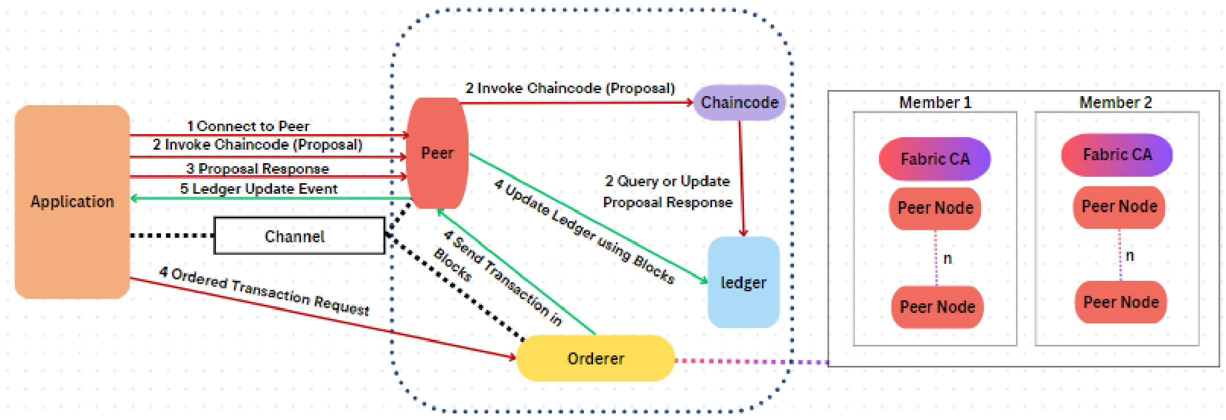

Application
1 Connect to Peer
2 Invoke Chaincode (Proposal)
3 Proposal Response
5 Ledger Update Event
Channel
4 Ordered Transaction Request
Peer
2 Invoke Chaincode (Proposal)
Chaincode
2 Query or Update Proposal Response
4 Update Ledger using Blocks
4 Send Transaction in Blocks
ledger
Orderer
Member 1
Member 2
Fabric CA
Peer Node
n
Peer Node
Fabric CA
Peer Node
n
Peer Node

Fig 4. Hyperledger Fabric Architecture

3.1. Dataset

MIMIC-III [27] encompasses deidentified, full clinical data of patients admitted to the critical care unit at the Beth Israel Deaconess Medical Centre in Boston, Massachusetts. It is readily available to researchers worldwide on PhysioNet under the terms of information usage condition. It contains over 40,000 actual patients' data including personal information, admission, regular follow-up information, prescription, ICUstays, and transfers. Specifically, we integrated the following fields into the ledger:

1. Patient Demographics: Age, gender, and medical history.
2. Admission Records: Admission and discharge dates, diagnosis codes, and treatment plans.
3. ICU Stays: Duration of ICU stays, vital signs, and medication administration.
4. Prescriptions: Drug names, dosages, and administration schedules.

Each transaction payload was designed to simulate typical EHR operations, with an average size of 1 KB per transaction. This workload is representative of real-world healthcare scenarios, ensuring the relevance and applicability of our findings.

3.2. Network Setup

The experiments were conducted in a distributed environment, with each organization's peer and orderer nodes deployed on separate virtual machines. The network was configured to simulate a real-world healthcare ecosystem, with four organizations (Hospital-1, Hospital-2, Hospital-3, and Health Insurer) and one per organization. This setup ensures that the performance metrics are representative of a distributed blockchain network. Infrastructural components are depicted in **Figure 5.** myehrchannel has created, joined the channel, installed the chaincode, approved the chaincode, committed the chain code if it received enough approvals from the participant organizations, invoked the chain code, queried the chain code, and enabled client communication.

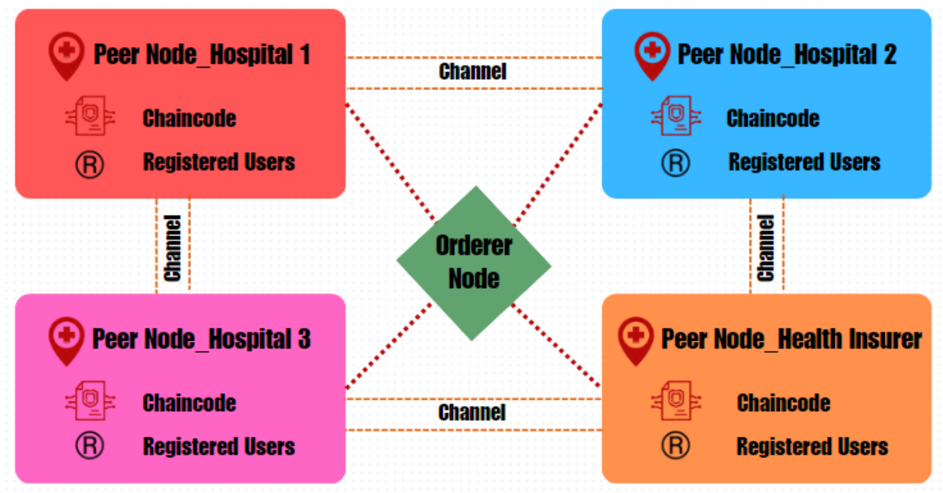


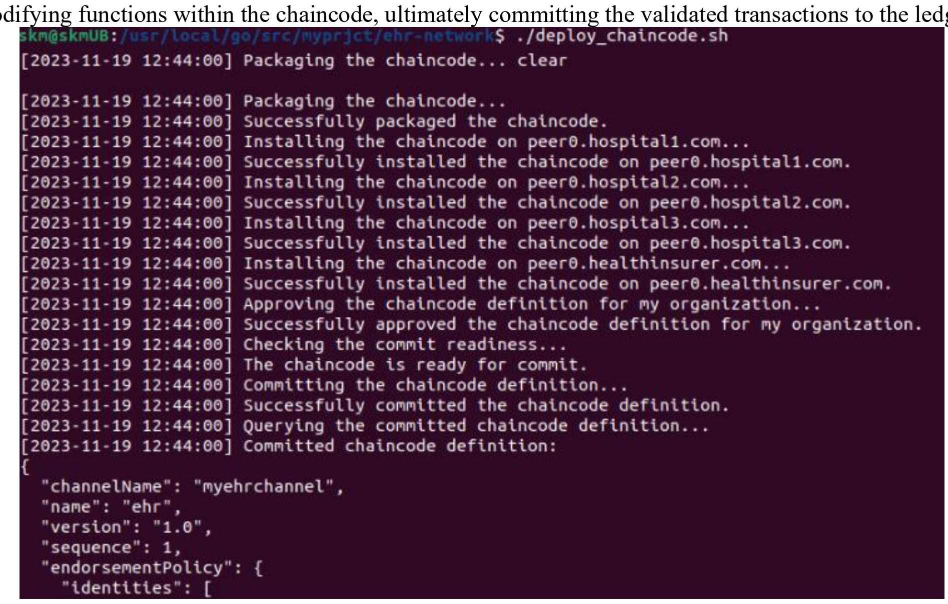
Fig 5. Major Components of Fabric Network

Participants, assets, transactions, and control logic established with Hyperledger Fabric are the fundamental components. These components are formed to carry out certain tasks that are controlled by the EHR chaincode, which is a collection of legislation. The CA is a critical component responsible for managing and issuing digital certificates to network participants. Certificates play a pivotal role in securing communications and establishing the identity of users within the network. This involves establishing a secure connection with the CA server using the provided administrative access. The enrollment process follows registration, providing the user with an Enrollment Certificate (ECert) necessary for participating in the network.

Local and channel MSPs are the two forms of MSPs that are present in fabric networks. The local node on which local MSPs are defined serves as their domain of responsibility. Each node needs its own local MSP. It is essentially just a file system directory containing the authorized certificates on the node. It should become evident how localized permission is implemented through inspection of the local MSP file layout. Specifically, the node itself knows which nodes to fully trust since *cacerts* and *intermediatecerts* include a set of certificates from the node's very own organization. The *keystore* has a unique private key for each node. A certificate in *signcerts* attests to the node's organization signing this node's private key. Subsequently, an accumulation of additional network nodes' certificates that the node with this MSP considers is stored by *tlscacerts* and *tlsintermediatecerts*. The present node is going to disregard the Transport Layer Security (TLS) certificate of another organization and decline all attempts at interaction regardless of whether it is mentioned here or not. An organization's eligibility for participation and membership in a channel, including organizations granted administrative access to change channel parameters, is specified by channel MSP. The following MSP is specified in the channel's specification directly on the ledger since it is crucial for everyone who is involved in the same channel to have a single, semantically consolidated perspective. There is thus no misunderstanding as to whoever is and is not permitted to take part. Four organizations' channel artifacts are configured via the execution of *configtx.yaml*, *create-artifacts.sh*, and *crypto-config.yaml*. It retrieves the MSP for all organizations with the intent to create the genesis block and channel transaction records. *Configtx.yaml* contains the relevant MSP directories for the orderer and peers. A need for agreement from a minimum of three organizations, and policies are tailored to be read, written, and endorsed by three of the four organizations[28-30].

3.3. Deploy Chaincode

Chaincode deployment is a comprehensive process that encompasses the installation, instantiation, and execution of smart contracts on the blockchain network. Figure 6 presents the chaincode deployment in the network. The installation phase involves packaging and distributing the chaincode to endorsing peers, ensuring its availability for execution. Subsequently, the *InstantiateChaincode* function initializes the chaincode on a specific channel, setting initial parameters. During transaction endorsement, custom functions within the chaincode dictate the business logic, guiding the simulation of proposed transactions by endorsing peers. The consensus step involves submitting endorsed transactions to the ordering service, which orders them into blocks. Peers then validate these transactions by re-executing state-modifying functions within the chaincode, ultimately committing the validated transactions to the ledger.



```
skm@skmUB:/usr/local/go/src/myprjct/ehr-network$ ./deploy_chaincode.sh
[2023-11-19 12:44:00] Packaging the chaincode... clear

[2023-11-19 12:44:00] Packaging the chaincode...
[2023-11-19 12:44:00] Successfully packaged the chaincode.
[2023-11-19 12:44:00] Installing the chaincode on peer0.hospital1.com...
[2023-11-19 12:44:00] Successfully installed the chaincode on peer0.hospital1.com.
[2023-11-19 12:44:00] Installing the chaincode on peer0.hospital2.com...
[2023-11-19 12:44:00] Successfully installed the chaincode on peer0.hospital2.com.
[2023-11-19 12:44:00] Installing the chaincode on peer0.hospital3.com...
[2023-11-19 12:44:00] Successfully installed the chaincode on peer0.hospital3.com.
[2023-11-19 12:44:00] Installing the chaincode on peer0.healthinsurer.com...
[2023-11-19 12:44:00] Successfully installed the chaincode on peer0.healthinsurer.com.
[2023-11-19 12:44:00] Approving the chaincode definition for my organization...
[2023-11-19 12:44:00] Successfully approved the chaincode definition for my organization.
[2023-11-19 12:44:00] Checking the commit readiness...
[2023-11-19 12:44:00] The chaincode is ready for commit.
[2023-11-19 12:44:00] Committing the chaincode definition...
[2023-11-19 12:44:00] Successfully committed the chaincode definition.
[2023-11-19 12:44:00] Querying the committed chaincode definition...
[2023-11-19 12:44:00] Committed chaincode definition:
{
  "channelName": "myehrchannel",
  "name": "ehr",
  "version": "1.0",
  "sequence": 1,
  "endorsementPolicy": {
    "identities": [
```

Fig 6. Deploy Chaincode

The chaincode was designed to simulate realistic EHR transactions, including:

1. Patient Record Updates: Adding or modifying patient demographics, medical history, and treatment plans.
2. Access Control Checks: Verifying user permissions before granting access to sensitive data.
3. Data Sharing between Organizations: Facilitating secure data exchange between hospitals and health insurers.

We conducted additional tests to evaluate the performance impact of these mechanisms, finding that the overhead associated with private data collections and encryption was minimal, with a less than 5% increase in latency and a 3% reduction in throughput.

3.4. Proposed Transactions

The proposed transaction flow is depicted in Figure 7. It ensures fault tolerance by promptly selecting a new leader in the event of a failure, operating on an efficient leader-based model for streamlined decision-making. Raft strategically ensures a high degree of openness for ordering services due to its majority voting endorsement policies, with the Genesis block being generated and executed following the creation of cryptographic materials.

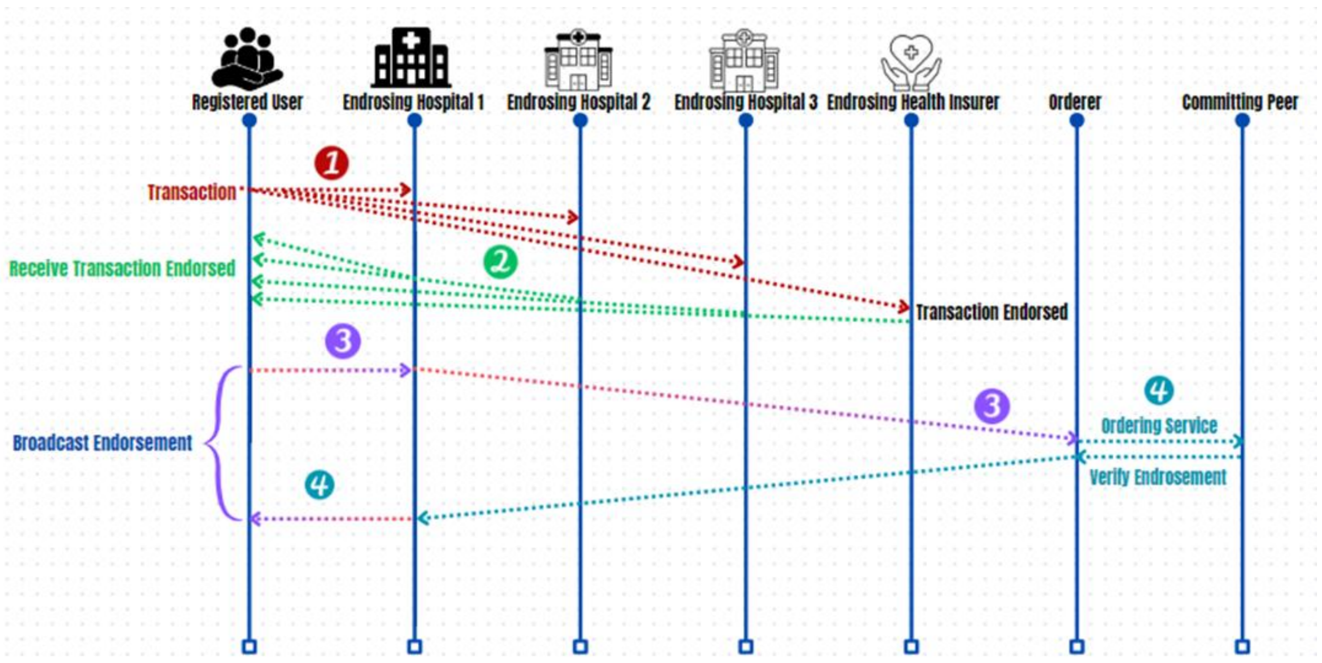


Fig 7. Proposed Transactions in the Fabric Network

Algorithm 1 presents the parameters for batching transactions made on the network. The ordering service configuration includes a BatchTimeout of 2 seconds, specifying the maximum time allowed to aggregate transactions into a block. The BatchSize parameters define constraints: MaxMessageCount limits transactions to 10 per block, AbsoluteMaxBytes caps block size at 256 MB, and PreferredMaxBytes suggests a target block size of 99 MB. These settings are pivotal for optimizing the performance and resource utilization of the Hyperledger Fabric network's ordering service.

Algorithm 1: Transaction batching parameters

```
1. Orderer: &OrdererDefaults
2. OrdererType: etcdraft
3. EtcdRaft:
4.   Consenters:
5.     - Host: hospital-orderer.hospital.com
6.       Port: 7050
7.       ...
8.    Addresses:
9.     - hospital-orderer.hospital.com:7050
10. BatchTimeout: 2s
11. BatchSize:
12. MaxMessageCount: 10
13. AbsoluteMaxBytes: 256 MB
14.  PreferredMaxBytes: 99 MB
```

3.5. Raft Consensus

Raft was chosen as the primary consensus algorithm for our e-health scenario due to its fault tolerance, simplicity, and suitability for permissioned networks. It ensures that the network can still reach a consensus over what is currently happening in the distributed log in the case of a system failure or unreachable node. Here's a high-level explanation:
Terms and Node Definitions, where N: The total node count within the Raft cluster, Ti: Term number of node I, Si: State of node i, where $Si \in \{\text{Follower, Candidate, Leader}\}$.
Election Process: When a node i decides to become a candidate, it increments its term and transitions to the Candidate state: $Ti' = Ti+1$, Si'=Candidate.
Voting Process: When a node j receives a RequestVote message from a candidate i:
If $Tj<Ti$, j votes for i: if $Tj<Ti$ then votes for i
If $Tj \geq Ti$, j does not vote for i: if $Tj \geq Ti$ then does not vote for i
Quorum Requirement: A candidate becomes the leader when it receives votes from most nodes: if the majority of votes are received then Si'=Leader.

Leader Confirmation: Once a node i receives votes from a majority, it transitions to the Leader state.
The fault-tolerance condition can be represented as: If the leader fails (crashes or becomes unreachable), the remaining nodes can opt for a new leader quickly. This condition is ensured by the election process, wherein nodes take part in voting, and a new leader gets selected

through voting from a majority. The concept of "promptly" is inherent in the Raft algorithm's design, where timeouts and randomized intervals are used to trigger elections promptly after the leader failure is detected.

3.6. Performance Analysis

Algorithm 2 enables the analysis of resource utilization and performance data by automating the benchmarking of EHR activities. The *ExecuteEHRBenchmark* function encapsulates the proposed approach, which consists of several critical phases to assess the reliability and efficacy of the network. The executable verifies the integrity of the parameters provided listed first by validating configurations. EHR workloads are dynamically loaded and processed using the blockchain client as part of an iterative benchmarking process, and transaction outcomes and resource utilization statistics are meticulously logged.

Algorithm 2: Hyperledger Caliper Execution
Input:
(1) benchmarkConfig: Transaction rates
(2) networkConfig: Blockchain network configuration
(3) chaincodeConfig: Chaincode configuration
(4) ehrWorkloadModule: Module generating EHR-specific transactions
Output: ehrBenchmarkReport: Report with performance metrics

```
1. Function ExecuteEHRBenchmark(benchmarkConfig, networkConfig, chaincodeConfig, ehrWorkloadModule):
2. If not benchmarkConfig.isValid():
3.  - Return "Invalid EHR benchmark configuration."
3. blockchainClient = Initialize using networkConfig.
4. If chaincodeConfig.isNotDeployed():
5.  - Deploy EHR chaincode using chaincodeConfig.
6. testResults = []
7.  For each round in benchmarkConfig.rounds:
8.   - workload = ehrWorkloadModule.load(round)
9.    - For each transaction in workload:
10.  - result = blockchainClient.sendEHRTransaction(transaction)
11.  - testResults.append(result)
12.   - If round.shouldMonitorResourceUsage():
13.   - resourceUsage = Monitor and record EHR system resource usage
14.   - testResults.append(resourceUsage)
15.  ehrBenchmarkReport = Analyze testResults to compile EHR performance metrics.
16.  CleanupEHRTestingEnvironment()
17. Return ehrBenchmarkReport
```

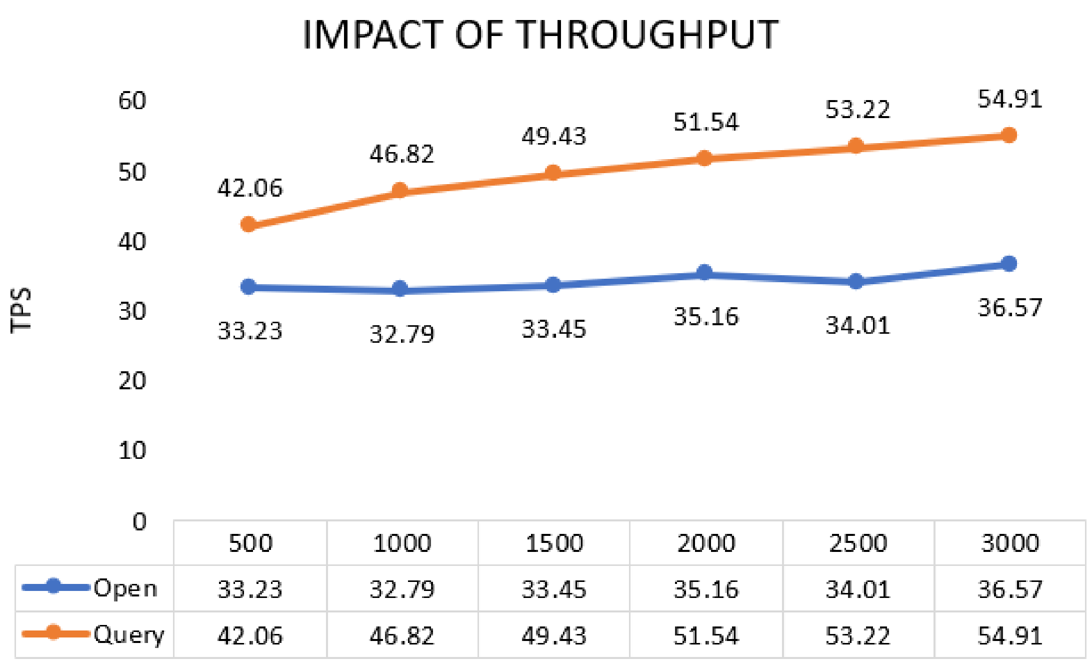
The methodology's flexibility in gauging resource utilization is one of its key features as it enables a detailed assessment of the system's functionality under different parameters. Ultimately, the algorithm evaluates the aggregated test results to provide an extensive EHR benchmark report that provides information on transaction throughput, latency, and other relevant performance indicators. The testing environment is kept immaculate after execution attributable to the cleanup phase's inclusion. The parameter configuration is mentioned in Table 4.

Table 4. Measurement Metrics of Fabric Network

Parameter	**Configuration**
Rounds	6
Network Size	4 Organizations, 1 Peer per Organization
Mode	Read and Write
Transactions Per Second (TPS)	500, 1000, 1500, 2000, 2500, 3000

4. Results and Disc

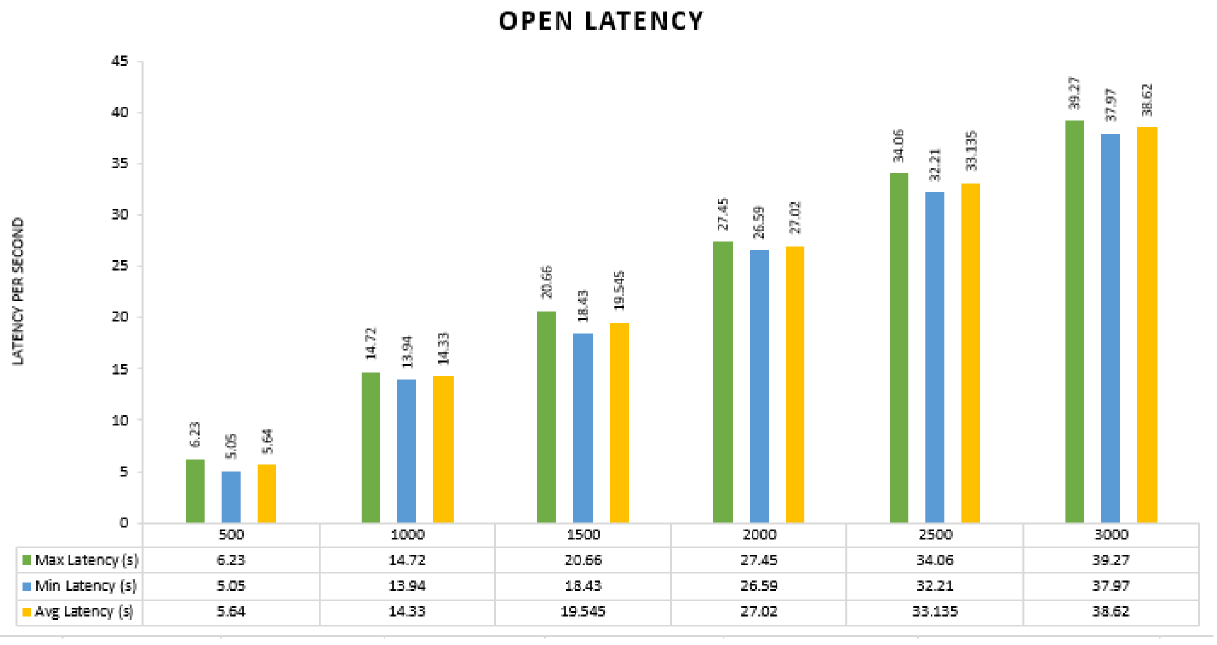
The transaction throughput metrics presented herein in Figure 8 encapsulate a comprehensive evaluation of the network's real-time performance across a spectrum of TPS rates, specifically delineating open and query operations. In the domain of Open transaction throughput, where the network's efficiency in managing write operations is observed, the recorded values ranging from 33.23 to 36.57 seconds illuminate the system's adeptness in processing and committing new transactions to the blockchain ledger. Concurrently, the query transaction throughput metrics, spanning from 42.06 to 54.91 seconds, provide a meticulous assessment of the network's responsiveness in executing read operations. It exhibits that the system can withstand growing loads up to the maximum measured point without seeing a decrease in transaction processing rate.



	500	1000	1500	2000	2500	3000
Open	33.23	32.79	33.45	35.16	34.01	36.57
Query	42.06	46.82	49.43	51.54	53.22	54.91

Fig 8. Transaction Throughput Performance

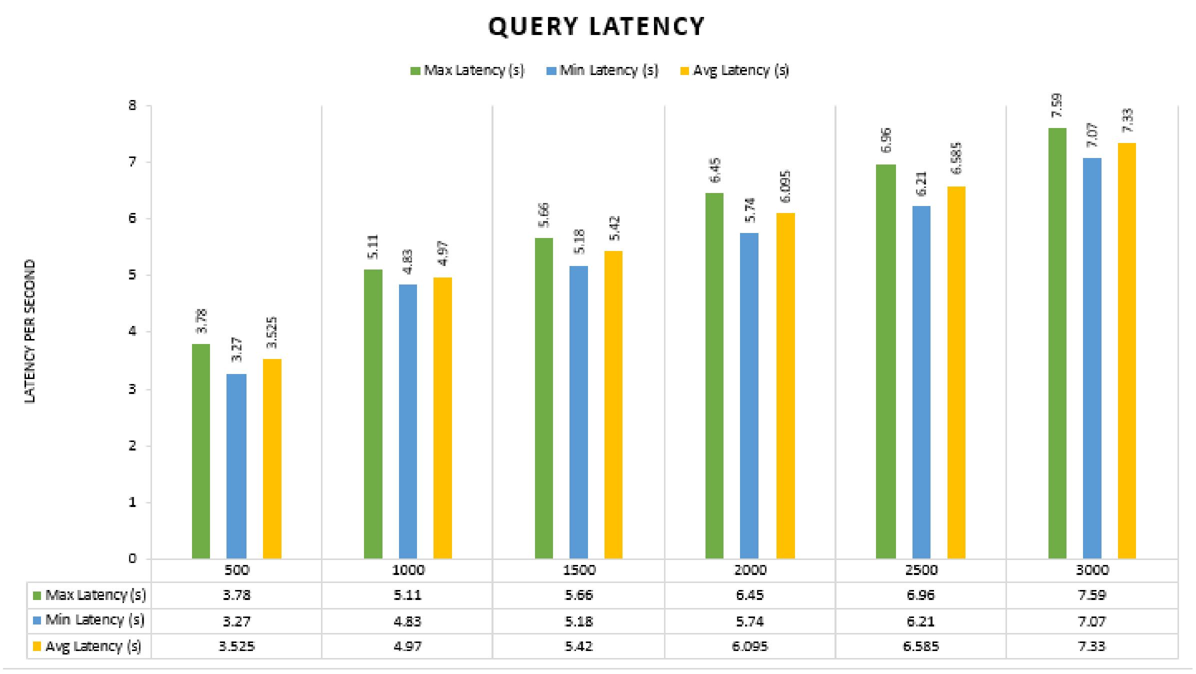
The outcomes of Figure 9 and Figure 10 stemming from the empirical examination of open and query latency offer intricate insights into the temporal intricacies of the network's responsiveness across varying transaction rates. It exhibited an average open latency of 5.64 seconds at 500 TPS and escalated up to 38.62 seconds. It underscores the importance of meticulous calibration and resource allocation to sustain optimal operational efficiency across diverse transactional loads.



	500	1000	1500	2000	2500	3000
🟩 Max Latency (s)	6.23	14.72	20.66	27.45	34.06	39.27
🟦 Min Latency (s)	5.05	13.94	18.43	26.59	32.21	37.97
🟨 Avg Latency (s)	5.64	14.33	19.545	27.02	33.135	38.62

Fig 9. Transaction Open Latency

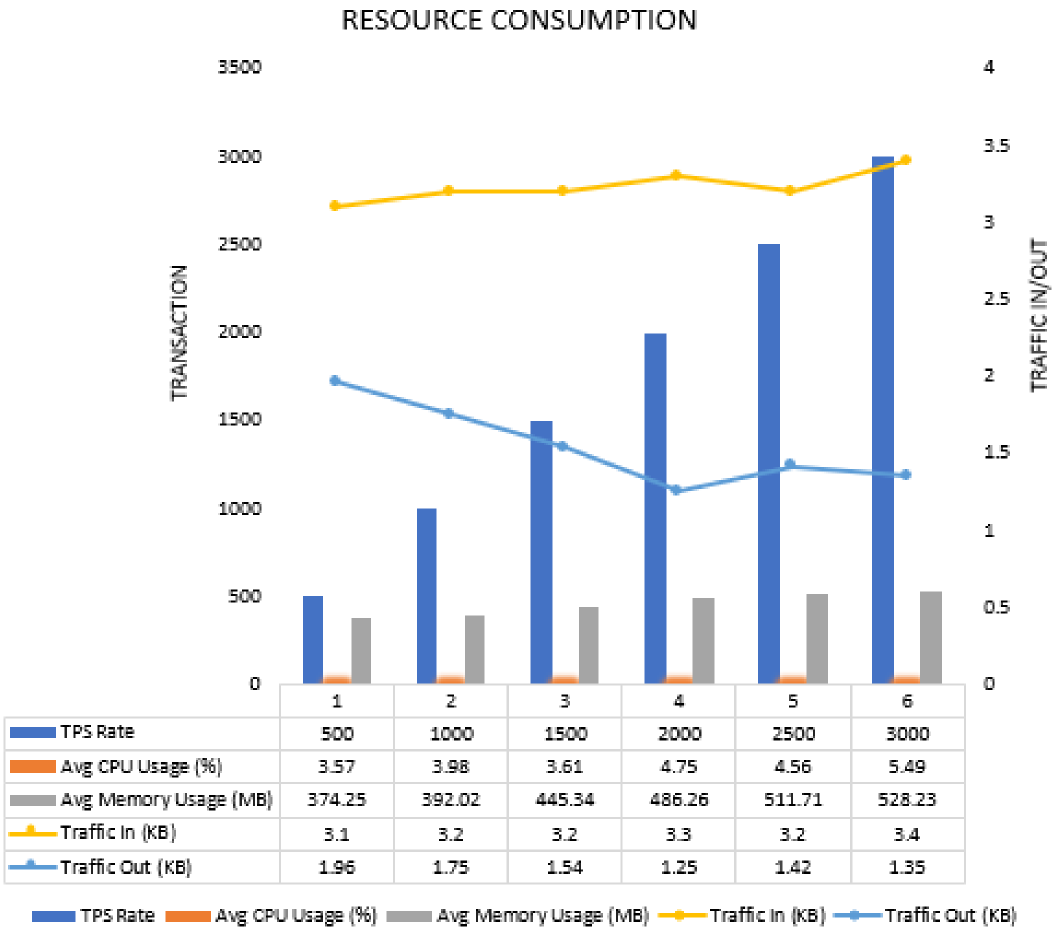
The query latency exhibited commendably low latencies, featuring an average of 3.525 seconds at 500 TPS. However, with a progressive increase in the transaction rate to 3000 TPS, a consequential escalation in query latencies becomes apparent. It emphasizes the inherent challenges associated with maintaining optimal responsiveness amidst heightened transactional workloads.



	500	1000	1500	2000	2500	3000
🟩 Max Latency (s)	3.78	5.11	5.66	6.45	6.96	7.59
🟦 Min Latency (s)	3.27	4.83	5.18	5.74	6.21	7.07
🟨 Avg Latency (s)	3.525	4.97	5.42	6.095	6.585	7.33

Fig 10. Transaction Query Latency

The resource metrics in Figure 11 were collected using a distributed monitoring tool that tracks CPU, memory, and network utilization across all nodes in the network. The results indicate that the system's resource utilization remained relatively stable, even under high transaction loads. This suggests that the network was not fully utilizing its available resources, highlighting the need for further optimization of the network configuration and workload distribution.



	1	2	3	4	5	6
TPS Rate	500	1000	1500	2000	2500	3000
Avg CPU Usage (%)	3.57	3.98	3.61	4.75	4.56	5.49
Avg Memory Usage (MB)	374.25	392.02	445.34	486.26	511.71	528.23
Traffic In (KB)	3.1	3.2	3.2	3.3	3.2	3.4
Traffic Out (KB)	1.96	1.75	1.54	1.25	1.42	1.35

Fig 11. Resource Consumption

Several studies in Table 3 do not report latency and throughput metrics under similar experimental conditions, making direct comparisons difficult. Khan et al. [28] evaluated Hyperledger Fabric v2.2 in a healthcare context, with a focus on latency and throughput under varying transaction loads. This aligns closely with our experimental setup, making their results directly comparable to ours.

Table 5 provides a comprehensive summary of distinct performance metrics, encompassing throughput and latency, across different evaluation settings and systems. The comparison involved the execution of 1000 TPS.

Table 5. Performance Metrics for Diverse Evaluation Settings

Authors	Year	Hyperledger Fabric (HF) Version	Organizations and Peers	Consen-sus	Latency		Throughput		Test Environment
					Query	Open	Query	Open	
Khan et al. [31]	2022	HF 2.2	2 Organizations (1 Peer each)	Raft	6.74	19.66	45.4	31.7	Intel ® Xeon®, 2.6 GHz with 12 core CPU, 16 GB RAM, 500 GB disk space, and Ubuntu 18.04 LTS
This Research	2023	HF 2.5	4 Organizations (3 Peer each)	Raft	4.97	14.33	46.82	32.79	Core 4 Threads CPU [Intel Core i5-@ 3.50 GHz - 3.90 GHz (3.90 GHz in overclocked mode) with 6MB shared L3 cache], 16 GB Memory, GPU with 8GB of memory, 256GB NVMe SSD, 1 Gbit/s network, and OS -Ubuntu 22.04 LTS

The results show that the network was operating near its capacity, leading to near-minute latencies. To mitigate this issue, we recommend optimizing the network configuration, including:
1. Increasing the number of peer nodes per organization to distribute the workload more evenly.
2. Adjusting the batch size and timeout parameters to reduce block commit times.
3. Implementing parallel transaction processing to improve throughput and reduce latency.

In addition to client-side latency and throughput, we evaluated the following metrics to provide a more comprehensive assessment of the network's performance:
1. Transaction Success Rates: The percentage of transactions successfully committed to the ledger.
2. Block Commit Times: The time taken to commit a block of transactions to the ledger.
3. Network Latency: The time taken for transactions to propagate across the network.
4. Error Rates: The frequency of transaction failures due to network congestion or resource limitations.

These metrics provide a more nuanced understanding of the network's performance under varying transaction loads, highlighting areas for improvement in scalability and resource allocation.
To assess the system's ability to handle real-world fluctuations in healthcare data traffic, we conducted additional stress testing with varying transaction rates (500 to 3000 TPS). The results indicate that the network can maintain stable performance under typical workloads, with error rates below 1% for transaction loads up to 2500 TPS. However, at 3000 TPS, the error rate increased to 3%, suggesting that the network was operating near its capacity. These findings highlight the need for further optimization to handle peak loads in real-world healthcare scenarios.

5. Conclusion

The research findings highlight the potential of Hyperledger Fabric v2.5 to support scalable, secure, and efficient EHR systems, while also identifying areas for future improvement. An empirical investigation of the Hyperledger Fabric version's functionality as a permissioned blockchain platform is presented here. The analysis has emphasized varying the transactions and requests workload. The impact on workload variation up to 3000 TPS is examined. Notable results show improvements in openness and operational productivity within the experimental network, with substantial decreases of 27.11% in open latency and 26.27% in query latency and increases of 3.13% in query throughput and 3.44% in open throughput compared to Khan et al. with the recent existing literature. Resource consumption analysis indicates a consistent rise in both CPU and memory use. Both outbound and incoming traffic show rather uniform use, with moderate variability. The specific configurations, versions, or settings of the Hyperledger Fabric network employed have an impact on the findings so results may vary depending on versions or setups. Scalability alternatives should be investigated and put into place to cater to an increasing number of network users and transactions. This can entail investigating parallel transaction processing, sharding strategies, or network communication protocol optimization.

References

[1] Das, S. R., Jhanjhi, N. Z., Asirvatham, D., Ashfaq, F., & Abdulhussain, Z. N. (2023, February). Proposing a Model to Enhance the IoMT-Based EHR Storage System Security. In *International Conference on Mathematical Modeling and Computational Science* (pp. 503-512). Singapore: Springer Nature Singapore.
[2] Kumar, M. S., Vimal, S., Jhanjhi, N. Z., Dhanabalan, S. S., & Alhumyani, H. A. (2021). Blockchain based peer to peer communication in autonomous drone operation. *Energy Reports, 7,* 7925-7939.
[3] Hyperledger Foundation by the Linux foundation. Available Online: https://www.hyperledger.org/ [Retrieved in July 2022].

[4] George, J. T. (2022). Introducing Blockchain Applications: Understand and Develop Blockchain Applications Through Distributed Systems. Apress.
[5] Li, J., & Carayon, P. (2021). Health Care 4.0: A vision for smart and connected health care. IISE transactions on healthcare systems engineering, 11(3), 171-180.
[6] Xu, X., Sun, G., Luo, L., Cao, H., Yu, H., & Vasilakos, A. V. (2021). Latency performance modeling and analysis for hyperledger fabric blockchain network. Information Processing & Management, 58(1), 102436.
[7] Junejo, A. Z., Hashmani, M. A., & Memon, M. M. (2021). Empirical evaluation of privacy efficiency in blockchain networks: Review and open challenges. Applied Sciences, 11(15), 7013.
[8] Liu, W., Zhang, D., Mu, C., Zhao, X., & Zhao, J. (2022). Ring-Overlap: A Storage Scaling Mechanism for Hyperledger Fabric. Applied Sciences, 12(19), 9568.
[9] Rahman, M. S., Al Omar, A., Bhuiyan, M. Z. A., Basu, A., Kiyomoto, S., & Wang, G. (2020). Accountable cross-border data sharing using blockchain under relaxed trust assumption. IEEE Transactions on Engineering Management, 67(4), 1476-1486.
[10] Dabbagh, M., Kakavand, M., Tahir, M., & Amphawan, A. (2020, September). Performance analysis of blockchain platforms: Empirical evaluation of hyperledger fabric and ethereum. In 2020 IEEE 2nd International conference on artificial intelligence in engineering and technology (IICAIET) (pp. 1-6). IEEE.
[11] Ashraf, H., Hanif, M., Ihsan, U., Al-Quayed, F., Humayun, M., & Jhanjhi, N. Z. (2023, March). A Secure and Reliable Supply chain management approach integrated with IoT and Blockchain. In 2023 International Conference on Business Analytics for Technology and Security (ICBATS) (pp. 1-9). IEEE.
[12] Alferidah, D. K., & Jhanjhi, N. Z. (2020, October). Cybersecurity impact over bigdata and iot growth. In 2020 International Conference on Computational Intelligence (ICCI) (pp. 103-108). IEEE.
[13] Humayun, M., Jhanjhi, N. Z., Hamid, B., & Ahmed, G. (2020). Emerging smart logistics and transportation using IoT and blockchain. IEEE Internet of Things Magazine, 3(2), 58-62.
[14] Amir Latif, R. M., Hussain, K., Jhanjhi, N. Z., Nayyar, A., & Rizwan, O. (2020). A remix IDE: smart contract-based framework for the healthcare sector by using Blockchain technology. Multimedia tools and applications, 1-24.
[15] Menon, S., Anand, D., Kavita, Verma, S., Kaur, M., Jhanjhi, N. Z., ... & Ray, S. K. (2023). Blockchain and Machine Learning Inspired Secure Smart Home Communication Network. Sensors, 23(13), 6132.
[16] Humayun, M., Jhanjhi, N. Z., Alsayat, A., & Ponnusamy, V. (2021). Internet of things and ransomware: Evolution, mitigation and prevention. Egyptian Informatics Journal, 22(1), 105-117.
[17] Almusaylim, Z. A., Alhumam, A., & Jhanjhi, N. Z. (2020). Proposing a secure RPL based internet of things routing protocol: A review. Ad Hoc Networks, 101, 102096.
[18] Kok, S., Abdullah, A., Jhanjhi, N., & Supramaniam, M. (2019). Ransomware, threat and detection techniques: A review. Int. J. Comput. Sci. Netw. Secur, 19(2), 136.
[19] Alamri, M., Jhanjhi, N. Z., & Humayun, M. (2019). Blockchain for Internet of Things (IoT) research issues challenges & future directions: A review. Int. J. Comput. Sci. Netw. Secur, 19(1), 244-258.
[20] Platt, M., & McBurney, P. (2023). Sybil in the haystack: A comprehensive review of blockchain consensus mechanisms in search of strong Sybil attack resistance. Algorithms, 16(1), 34.m
[21] Sonkamble, R. G., Bongale, A. M., Phansalkar, S., Sharma, A., & Rajput, S. (2023). Secure Data Transmission of Electronic Health Records Using Blockchain Technology. Electronics, 12(4), 1015.
[22] Rajawat, A. S., Goyal, S. B., Bedi, P., Simoff, S., Jan, T., & Prasad, M. (2022). Smart Scalable ML-Blockchain Framework for Large-Scale Clinical Information Sharing. Applied Sciences, 12(21), 10795.
[23] Pang, Z., Yao, Y., Li, Q., Zhang, X., & Zhang, J. (2022). Electronic health records sharing model based on blockchain with checkable state PBFT consensus algorithm. IEEE Access, 10, 87803-87815.
[24] Fatokun, T., Nag, A., & Sharma, S. (2021). Towards a blockchain assisted patient owned system for electronic health records. Electronics, 10(5), 580.
[25] Rajput, A. R., Li, Q., & Ahvanooey, M. T. (2021, February). A blockchain-based secret-data sharing framework for personal health records in emergency condition. In Healthcare (Vol. 9, No. 2, p. 206). MDPI.
[26] Wang, Q., & Qin, S. (2021). A hyperledger fabric-based system framework for healthcare data management. Applied Sciences, 11(24), 11693.
[27] The MIMIC-III Clinical Database. Available Online: PhysioNet (doi: 10.13026/C2XW26). For more information on the collection of the data, see the MIMIC-III Clinical Database page.
[28] Unsriana, L., Dhaniar, A., Tamarina, P. M., & Utari, N. (2025). The Impacts of Collaborative Online International Learning, Web-Based Language Learning, and Onsite Learning on Japanese Language Ability. *International Journal of Engineering, Science and Information Technology*, 5(3), [page numbers if available]. https://doi.org/10.52088/ijesty.v5i3.903
[29] Halim, A., Safwandi, S., & Fajriana, F. (2024). *Application of Data Mining with the Least Square Method to Predict Web-Based Drug Inventory*. International Journal of Engineering, Science and Technology (IJESTY), 5(3). https://doi.org/10.52088/ijesty.v5i3.897
[30] Berutu, I. F., Dinata, R. K., & Afrillia, Y. (2025). *Public Facility Recommendation System in Subulussalam City Using Fuzzy C-Means Algorithm*. International Journal of Engineering, Science and Information Technology (IJESTY), 5(3). https://doi.org/10.52088/ijesty.v5i3.873
[31] Khan, D., Jung, L. T., Hashmani, M. A., & Cheong, M. K. (2022). Empirical performance analysis of hyperledger LTS for small and medium enterprises. Sensors, 22(3), 915.