https://doi.org/10.31436/iiumej.v26i3.3610

PERFORMANCE BENCHMARKING OF HYPERLEDGER FABRIC ON HETEROGENEOUS HARDWARE FOR IOT APPLICATIONS

MUHAMMAD MUAZ ZULKARNAIN[1], NABILAH RAMLI[1*], ANIS NURASHIKIN NORDIN[2]

[1]*Department of Mechanical and Aerospace Engineering, Kulliyyah of Engineering, International Islamic University Malaysia, Kuala Lumpur, Malaysia*
[2]*Department of Electrical and Computer Engineering, Kulliyyah of Engineering, International Islamic University Malaysia, Kuala Lumpur, Malaysia*

**Corresponding author: nabilah@iium.edu.my*

(Received: 13 February 2025; Accepted: 22 August 2025; Published online: 9 September 2025)

ABSTRACT: Hyperledger Fabric (HLF), a widely used open-source private blockchain, has garnered attention for its ability to enable blockchain in Internet of Things (IoT) applications. Embedding HLF nodes within IoT devices enables smart contract integration for secure and automated communications, thereby reducing reliance on intermediaries. However, resource-constrained IoT devices often face challenges with complex operations due to their limited processing power. While HLF deployment on single-board computers (SBCs) like Raspberry Pi has been explored, comprehensive performance evaluations across diverse hardware setups in a heterogeneous blockchain network are limited. This study benchmarks HLF performance on a network comprising SBCs with ARM architectures (Cortex-A72 and Cortex-A76) and laptops with Intel Core i7 and Intel Celeron processors. Using Hyperledger Caliper, key performance metrics, including throughput, latency, CPU usage, and memory consumption, were measured. Results show that high-throughput applications are best supported by high-end processors capable of handling multiple clients, achieving up to 1,148.3 TPS. In contrast, SBCs efficiently handle moderate transaction loads from single clients with minimal latency increases. These findings demonstrate the adaptability of HLF across varied hardware configurations, provided a proper network architecture setup, supporting its deployment in diverse IoT environments.

ABSTRAK: Fabrik hiperlejer (Hyperledger Fabric, HLF), iaitu rantaian blok peribadi sumber terbuka yang digunakan secara meluas, telah mendapat perhatian kerana keupayaannya dalam memacu blok rantaian dalam aplikasi Internet of Things (IoT). Penyepaduan nod HLF ke dalam peranti IoT membolehkan penggunaan kontrak pintar bagi komunikasi selamat dan automatik, seterusnya mengurangkan kebergantungan kepada pihak ketiga. Namun, peranti IoT yang memiliki sumber terhad sering menghadapi cabaran dalam melaksanakan operasi kompleks disebabkan oleh kuasa pemprosesannya yang terhad. Walaupun penerapan nod HLF pada komputer papan tunggal (SBC) seperti Raspberry Pi telah dikaji, penilaian prestasi yang komprehensif merangkumi pelbagai konfigurasi perkakasan dalam blok rantaian heterogen masih terhad. Kajian ini merupakan penanda aras prestasi HLF pada rangkaian yang terdiri daripada SBC berasaskan seni bina ARM (Cortex-A72 dan Cortex-A76) serta komputer riba berprosesor Intel Core i7 dan Intel Celeron. Mengguna pakai Hyperledger Caliper, metrik prestasi utama seperti kadar penghantaran, masa tindak balas, penggunaan CPU, dan penggunaan memori telah diukur. Dapatan kajian menunjukkan bahawa kadar aplikasi penghantaran tertinggi, paling sesuai disokong oleh pemproses berprestasi tinggi yang mampu mengendalikan pelbagai klien, mencapai sehingga 1,148.3 TPS. Sementara itu, SBC berupaya mengendalikan beban transaksi sederhana daripada klien tunggal dengan peningkatan masa tindak balas yang minima. Penemuan ini menunjukkan kebolehsesuaian

HLF merentasi pelbagai konfigurasi perkakasan, dengan syarat rangkaian yang sesuai disediakan, sekaligus menyokong pelaksanaannya dalam persekitaran IoT yang pelbagai.

KEY WORDS: *Hyperledger Fabric, Hyperledger Caliper, Internet of Things (IoT), Performance Analysis, IoT-Enabled Blockchain*

1. INTRODUCTION

The rapid evolution of the IoT has transformed industries by enabling interconnected devices to process and exchange data in real-time, playing a pivotal role in sustainability and environmental management [1]. IoT drives advancements in renewable energy adoption, energy efficiency, and carbon offset tracking, aligning with global efforts to combat environmental challenges and achieve the Sustainable Development Goals (SDGs) [2], [3]. When integrated with blockchain technology, IoT's capabilities are further enhanced through decentralized, secure, and immutable ledgers that enable applications such as peer-to-peer energy trading, real-time energy monitoring, and carbon credit traceability [4], [5]. These innovations strengthen data security, improve traceability, and promote sustainable energy practices, accelerating the shift toward cleaner energy systems [6].

However, traditional centralized IoT systems often suffer from single points of failure, latency issues, and cybersecurity risks [7], [8]. Blockchain technology addresses these problems by decentralizing data management and enhancing security. Integrating blockchain directly into IoT devices as single units with in-situ capability offers the potential for improved efficiency and autonomy. Despite these advantages, this approach introduces new challenges, particularly scalability concerns and the need for efficient network coordination in distributed environments.

A key challenge in integrating blockchain with IoT devices in situ lies in the resource constraints of IoT devices. These devices frequently operate with limited CPU, memory, and storage capacity, which are further strained by blockchain operations such as cryptographic signing, state database updates, and consensus participation [9]. Furthermore, consensus mechanisms such as proof of work (PoW) further exacerbate these issues, introducing high latency and scalability challenges, especially in large-scale deployments [10].

In heterogeneous IoT networks, hardware performance varies significantly — from high-end servers or laptops acting as gateways to low-power single-board computers (SBCs) operating at the edge. Understanding how these different classes of devices perform under blockchain workloads is critical for IoT architects, as it informs role assignment, workload balancing, and cost-performance trade-offs.

This paper aims to benchmark Hyperledger Fabric performance across heterogeneous IoT-representative hardware (high-end, mid-range, and SBC) by investigating a system network architecture using Hyperledger Fabric deployed on heterogeneous hardware, including two personal computers with Intel Core i7-1355U @ 1.70 GHz and Intel Celeron 3205U @ 1.50 GHz processors, and two SBCs with ARM Cortex-A72 @ 1.8 GHz and Cortex-A76 @ 2.4 GHz processors. This combination reflects real-world IoT deployment scenarios where central nodes possess high processing power, while distributed edge devices are cost-efficient and energy-efficient but resource-limited. Performance was benchmarked using Hyperledger Caliper (HLC) [11] to evaluate key metrics such as transaction throughput, latency, and resource utilization to analyze the impact of hardware specifications in distributed blockchain-based IoT ecosystems. By directly linking these performance metrics to IoT operational requirements, the study provides insights into how hardware constraints affect blockchain

https://doi.org/10.31436/iiumej.v26i3.3610

performance in scenarios such as carbon offset reporting, energy metering, and other data-intensive IoT services.

The paper is organized as follows: Section II covers the background knowledge on the Hyperledger system and related works; Section III outlines the methodology approach; Section IV presents the results and discusses the system's performance based on the key metrics. Finally, Section V concludes the paper by summarizing the findings and suggesting directions for future work.

2. BACKGROUND

2.1. Hyperledger Fabric

Hyperledger Fabric is a permissioned blockchain platform for enterprise-grade applications, offering a modular architecture for flexible customization of components such as consensus protocols and membership services [12]. Hyperledger Fabric supports lightweight consensus mechanisms like Raft, making it highly efficient and well-suited for applications requiring high throughput and low resource consumption. As illustrated in Figure 1, the architecture highlights the core component of Hyperledger Fabric, the Certificate Authority (CA), which is responsible for issuing digital certificates to verify and authenticate network participants, ensuring that only authorized entities can access and interact with the network. Another key element is the peer, which maintains a replicated copy of the ledger, endorses transactions, and hosts smart contracts, known as chaincode, that encode the business logic governing transaction execution.

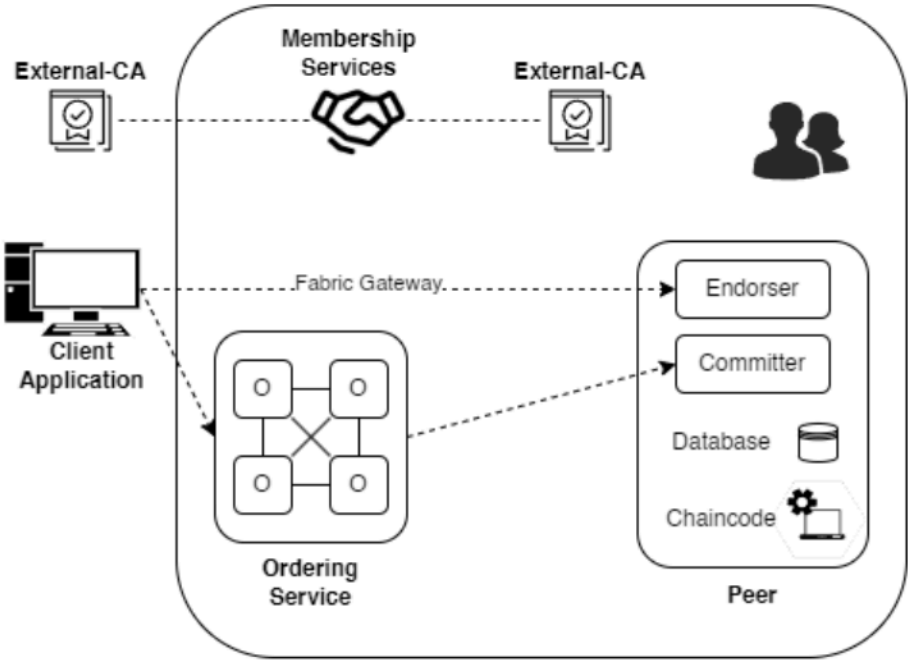

Figure 1. Modular architecture of Hyperledger Fabric, highlighting its key components

The orderer serves as the backbone of transaction management, orchestrating the ordering of transactions and distributing blocks to peers to ensure a consistent and validated transaction sequence across the network. The ledger structure in Hyperledger Fabric is divided into two parts: the world state and the transaction log. The world state represents the current snapshot of all network assets, while the transaction log maintains an immutable record of all

transactions that have shaped the current state. This dual-layer design facilitates efficient asset querying, robust data validation, and comprehensive audit trails for enterprise applications.

2.2. Hyperledger Caliper

Hyperledger Caliper (HLC) is a benchmarking tool designed to evaluate the performance of blockchain networks by generating workload scenarios and collecting metrics that provide performance insights under various conditions [13]. It achieves this by interfacing directly with Fabric networks through the Fabric peer gateway, allowing it to interact with the network like a client application, such as sending transactions to the network and querying the ledger. The architecture of HLC is shown in Figure 2.

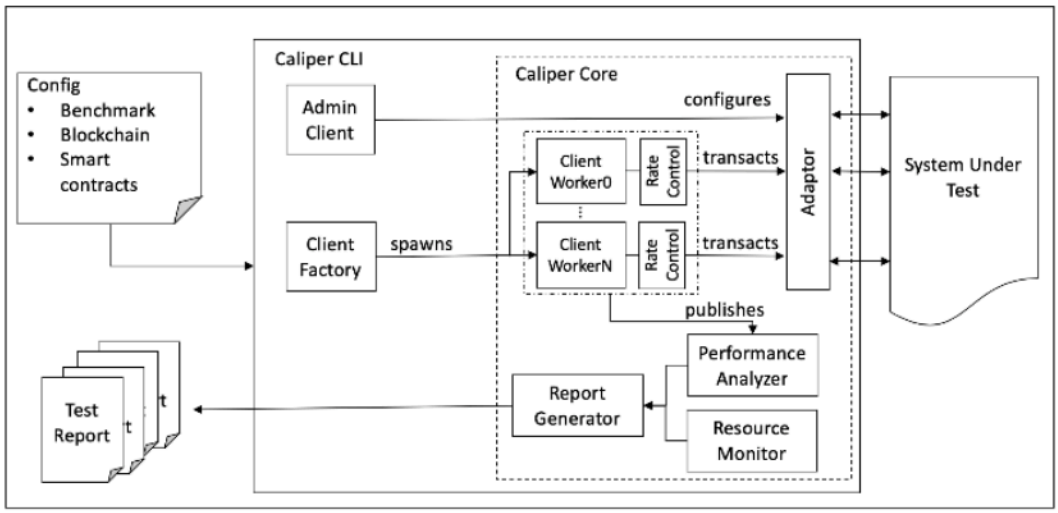

Figure 2. System architecture of Hyperledger Caliper and connection to the System Under Test (SUT) [13]

The benchmarking process begins with configuring the test environment and defining the workload, including a predefined set of transactions and operations to be executed on the blockchain network. Hyperledger Caliper submits transaction proposals to endorsing peers based on the network's defined endorsement policy, using specified network and connection profiles. Peer responses are collected and analyzed to evaluate key performance metrics, including throughput (TPS), which measures the number of transactions processed per second, and latency, which assesses the time required for transaction confirmation. Caliper also monitors memory consumption (usage of peers and orderers), CPU utilization (processing resources consumed), disk read/write (R/W) operations, and network traffic (data exchanged between nodes).

Hyperledger Caliper interacts with Hyperledger Fabric as the system under test (SUT), leveraging the Fabric peer gateway for transaction invocation and response collection. It integrates with tools like Prometheus and Grafana for enhanced monitoring and visualization, offering real-time metrics and dashboard capabilities. Workload definitions and core functionalities are implemented using Node.js and JavaScript, providing a flexible framework for scripting and executing benchmarks.

2.3. Transaction Flow in Hyperledger Fabric

The transaction flow within Hyperledger Fabric is illustrated in Figure 3. The flow follows a structured process to ensure transactions are accurately endorsed, ordered, and committed to the ledger. This process begins with a client application submitting a transaction proposal to

https://doi.org/10.31436/iiumej.v26i3.3610

the endorsing peers (step 1). These peers simulate the transaction by executing the associated chaincode (step 2) and return a proposal response that includes a digital signature and the read/write set of the simulated transaction (step 3). The client application collects and verifies these endorsement responses against the endorsement policy. If the policy is satisfied, the client submits the transaction to the orderer (step 4).

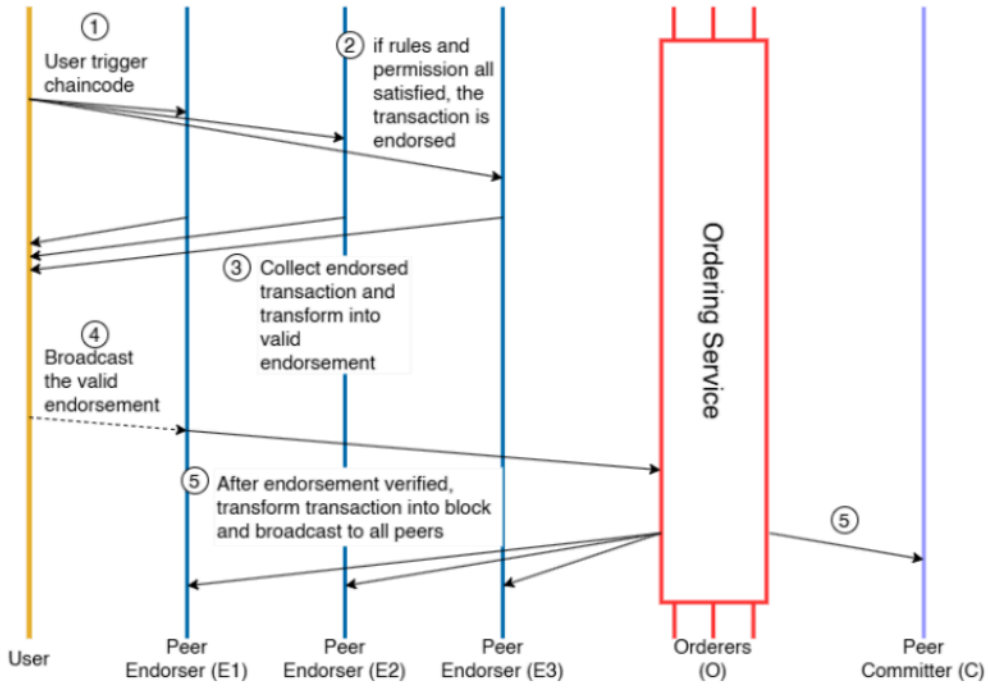

Figure 3. The timing diagram showing the flow of chaincode endorsement and block distribution across different types of nodes in Hyperledger Fabric

The orderer organizes the transaction into a block and disseminates this block to all network peers (step 5). Each peer validates the transaction to ensure it conforms to the endorsement policy and checks for any conflicts in the read/write set. Once validated, the transaction is committed to the ledger, and the world state is updated accordingly.

2.4. Related Works

Research on Hyperledger Fabric's performance and scalability emphasizes its suitability for secure and efficient transaction processing across various applications, including IoT. Comparative studies show that Fabric generally outperforms Ethereum in terms of throughput and latency, though its performance declines under high transaction rates, particularly in IoT healthcare systems [14]. Studies [15], [16] identify key performance factors sensitive to configuration, such as CouchDB utilization and transaction validation speeds, which impact scalability and latency.

Investigations into IoT integration in [17], [18], [19] highlight Fabric's strengths, with implementations in smart cities and enhanced security models improving both throughput and resource efficiency over standard models. In particular, a Raspberry Pi-based access control system demonstrated Fabric's capability in IoT device management. Further studies focusing on data integrity and security underscore Fabric's support for high data security, although scalability challenges remain [20]. Analysis of different node configurations offers developers insights for reducing latency [21].

https://doi.org/10.31436/iiumej.v26i3.3610

Empirical research on Fabric's scalability for small and medium enterprises (SMEs) [22] and validated the latency model [23] further contribute to understanding its limitations and potential. While Fabric performs well under moderate workloads, larger-scale deployments require careful optimization to address scalability and performance bottlenecks effectively.

Moreover, several studies and pilot projects have highlighted the application of Hyperledger Fabric in supply chain management and halal certification, particularly in Southeast Asia. Implementations have demonstrated Fabric's ability to ensure product traceability, authenticity, and compliance within complex supply chains, with the added benefit of supporting halal certification processes through transparent and tamper-proof recordkeeping [24], [25]. These use cases further validate Fabric's flexibility in enabling trust, auditability, and process automation in real-world business environments. They also highlight the need for network optimization to manage increased transaction volumes and data complexity.

3. METHODOLOGY

This work implemented a heterogeneous blockchain network comprising nodes with varying hardware architectures and computational resources to evaluate the impact of heterogeneity on performance and scalability. The network consisted of a high-performance laptop, a medium-performance laptop, and two SBCs equipped with ARM Cortex-A72 and Cortex-A76 processors. Table I outlines the hardware specifications and CPU scores, determined using the PassMark Performance Test [26], which evaluates computational capabilities through tasks such as integer and floating-point calculations, data compression, and encryption. These scores provide a standardized measure of processing power, essential for understanding each node's ability to support blockchain operations in Hyperledger Fabric. The methodology began with the design of the blockchain network topology, followed by optimizing orderer settings and developing chaincode for transaction testing. The network was then deployed, and performance measurements were conducted using Hyperledger Caliper.

Table 1. Summary of Hardware Specifications

Device Number	Device Model	Processor	CPU Score	Disc Type	Nodes in the Device	Idle	Sustained Load
1	Acer Aspire 5	Intel Core i7-1355U @1.70 GHz	14820	SSD	2 peers	6.9W	63.4W
2	Acer Aspire R14	Intel Celeron 3205U @ 1.50 GHz	947	HDD	2 peers	6.6W	32.2W
3	Raspberry Pi 4	ARM Cortex-A72 1.8 GHz	692	HDD	2 peers	2.44W	4.84W
4	Raspberry Pi 5	ARM Cortex-A76 2.4 GHz	2151	HDD	3 orderers	4.7W	11.6W

3.1. Blockchain Network Architecture Design

The blockchain network setup consists of four devices, as listed in Table I, each assigned specific roles. Ubuntu was selected as the operating system for all devices to ensure compatibility with Hyperledger Fabric dependencies, including Docker, Docker Compose, Node.js, and the Go programming language. Docker containers were used to host all nodes and services, with Docker Swarm orchestrating the network to create a unified system.

As illustrated in Figure 4, Devices 1, 2, and 3 host the peer nodes, representing high-performance, mid-range, and low-end hardware tiers, respectively. Despite their hardware differences, all peer nodes were uniformly configured, with each device hosting an endorser

https://doi.org/10.31436/iiumej.v26i3.3610

node, a committer node, a dedicated database, and a chaincode container. This standardized configuration across diverse hardware tiers provided a consistent foundation for evaluating the impact of computational resources on transaction throughput, latency, and resource utilization within the network.

3.2. Orderer Configuration Optimization

Before benchmarking individual peer nodes, the configuration of the orderer node in the Hyperledger Fabric network was optimized. This process, a standard best practice in setting up Hyperledger Fabric networks, focused on fine-tuning key parameters to ensure scalability and performance. First, the maximum pending load was adjusted to identify the tipping point of the orderer node without sacrificing throughput. This experiment involved monitoring the number of pending transactions the orderer could handle before performance degradation occurred.

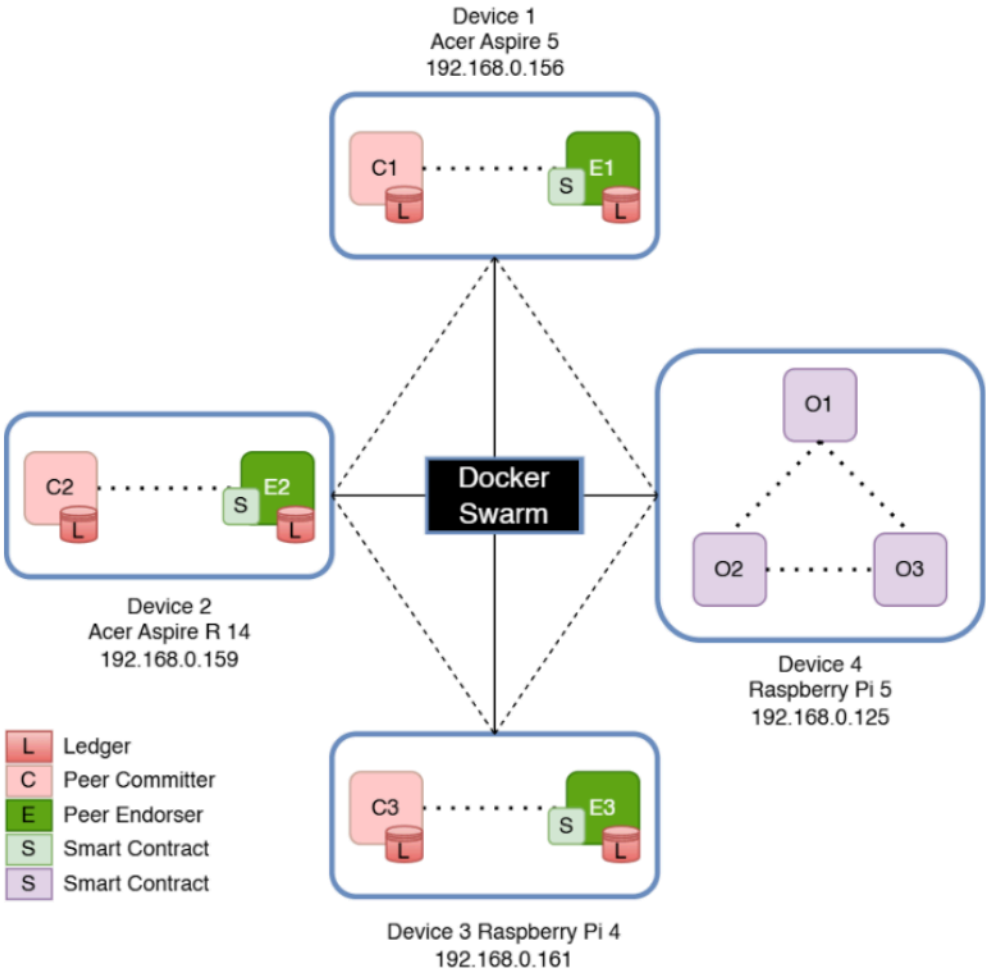

Figure 4. Benchmarked Hyperledger Fabric nodes topology. Devices 1, 2, and 3 hold the same configuration of peer nodes while Device 4 acts as the Orderer organization.

Next, the block size was fine-tuned to manage memory effectively. By adjusting the number of transactions per block, the configuration ensured that each block could accommodate a balanced load without overwhelming the system. Then the batch timeout parameter was fine-tuned by testing various intervals for batching transactions, minimizing delays during low-traffic periods while maintaining efficient block formation under high traffic. These optimized parameters were obtained through iterative experimentation, allowing

for the identification of configurations that optimized the network for a high-performing and consistent baseline.

3.3. Peer Nodes Benchmarking

The benchmarking settings were defined in the caliper workload configuration file (caliper-config.yaml) and network configuration file (network.yaml), specifying details of the blockchain network, chaincodes to invoke, and assigned workload parameters. Three chaincodes were used in this benchmarking exercise to evaluate the network's read, write, and read/write operations capability. These chaincodes include read, tokenize (create a digital asset), and transfer (do a digital transfer), respectively, as illustrated in Figure 5.

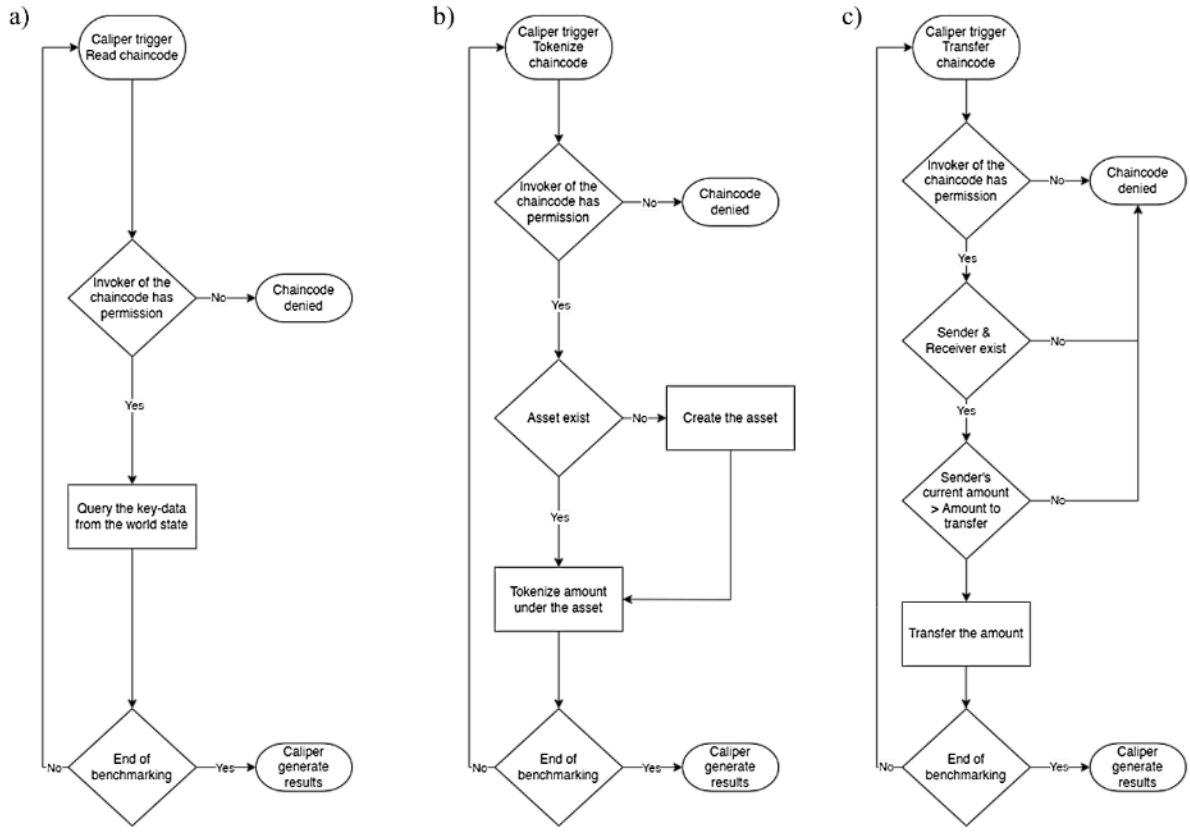

Figure 5. Three Flowcharts of Chaincode Benchmarked. (a) Read chaincode query ledger data from the blockchain world state, (b) Tokenize chaincode, create a new asset if it doesn't exist, or tokenize a new amount to the existing asset, (c) Transfer chaincode check if the sender's amount is higher than the amount to transfer to the sender before doing the transaction

The read chaincode (Figure 5a) retrieves an asset's current state from the state database without generating a new transaction that alters the state database or the blockchain ledger. This chaincode evaluates query performance and provides insights into how it scales with larger datasets and higher concurrency levels.

Figure 5b illustrates the process flow for tokenization, which involves a chaincode that creates a new asset on the blockchain. This consists of writing a new key-value pair to the state database and recording the transaction on the blockchain. The tokenization process tests the network's capacity to handle a large number of asset creation requests in a short time. It analyzes the impact of ledger growth on write-heavy operations and storage performance.

https://doi.org/10.31436/iiumej.v26i3.3610

The third chaincode, transfer, as shown in Figure 5c, updates an existing asset's state to reflect a transfer or change in ownership. This process involves modifying an existing entry in the state database and recording the transaction on the blockchain. It combines read operations (to validate asset ownership or conditions) with write operations (to update the asset state). This chaincode evaluates the network's ability to process complex transactions involving both reads and writes, helping to identify bottlenecks in transaction validation and contention resolution, particularly under high concurrency conditions.

To execute the benchmarking process, the following workload parameters were configured: the Transaction Rate, which defines the rate at which transactions are sent to the blockchain in transactions per second (TPS); the Total Number of Transactions, which specifies the overall volume of transactions to be executed during the test; and the Number of Workers, which represents the total number of concurrent clients generating transactions. These parameters simulate multiple clients interacting with the blockchain and evaluate the network's ability to handle concurrency while maintaining performance.

4. RESULTS AND DISCUSSION

The results section will first discuss the results of the orderer configuration optimization using HLC, where the optimal settings for maximum pending load, block size, and batch timeout can be obtained. These numbers were then used as the orderer parameters. The second part of the results discusses the Caliper measurements for each device running the HLF nodes. The workload settings for each device differ based on its capability to handle the load. The resulting network performance and the resource consumption for each are presented.

4.1. Orderer Configuration Optimization

The tokenize chaincode was used to execute operations for this step, with a fixed transaction load rate of 50 TPS. The simulation involved 100 workers simulating concurrent clients and operating on 10,000 assets. The performance of the orderer node was evaluated using two key metrics, as shown in Figure 6: throughput, the number of successful transactions processed per second (TPS), and success rate, the percentage of submitted transactions successfully processed and committed to the blockchain.

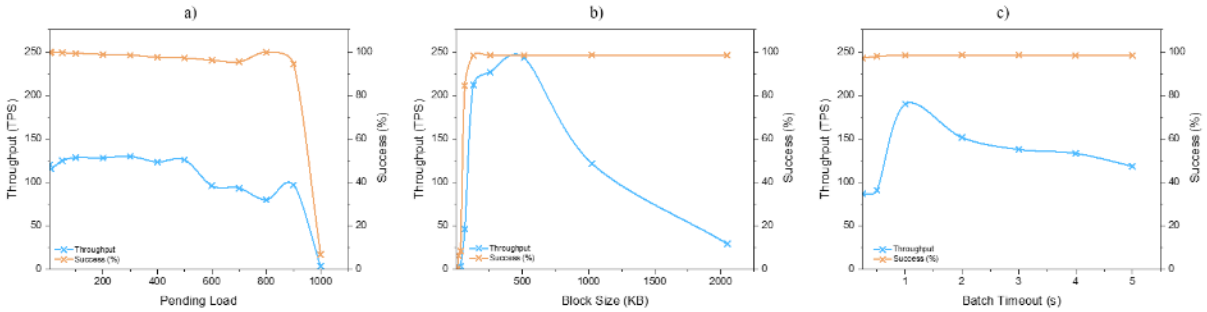

Figure 6. Optimizing configuration for the orderer nodes by modifying and testing: 6a) Pending Load, 6b) Block Size, and 6c) Batch Timeout against Throughput (TPS) and Success (%)

To find the optimum maximum pending load, the simulation was run for 300 seconds. Pending load defines the maximum number of transactions waiting to be grouped into blocks before being sent to peer nodes for validation and commitment. Starting with a pending load of 5, the value was incrementally increased by 100 up to 1,000, while throughput and success

https://doi.org/10.31436/iiumej.v26i3.3610

rate were recorded. As shown in Figure 6a, the orderer could handle a pending load of up to 500, achieving a throughput of approximately 130 TPS while maintaining a success rate above 95%. Beyond this point, latency increases, hence degrading throughput. However, the success rate remained stable until a pending load of 900. At 1,000 pending transactions, the network reached its limit.

Figure 6b shows the effect of block size on network performance. Block size refers to the maximum amount of data in a single block. Starting with an initial block size of 16 KB, the size was incrementally doubled to 2,048 KB. The highest throughput, approximately 243 TPS, was achieved at a block size of 512 KB, while the success rate remained unaffected across all tested sizes.

The effect of batch timeout on performance is shown in Figure 6c. Batch timeout specifies the maximum time the orderer waits before forming a block, even if the block size has not been reached. The simulation tested batch timeouts ranging from 0.25 seconds to 5 seconds. A batch timeout of 1 second gives the highest throughput, with the success rate remaining stable from 1 second to 5 seconds. These tests identified the optimal settings for the orderer node: a pending load of 500, a block size of 512 KB, and a batch timeout of 1 second.

4.2. Peer Nodes Benchmarking Results

The three devices evaluated in this step were Device 1, Device 2, and Device 3, benchmarked across three chaincode operations: read, tokenize, and transfer. Hyperledger Caliper recorded performance metrics, including network performance metrics (throughput and latency) and hardware utilization metrics (CPU usage, memory consumption, and network traffic).

Each device was tested according to its computational capacity. Device 1, a high-performance laptop, handled 10 workers simulating concurrent clients, while Device 2, a mid-range laptop, managed 5 workers. Device 3, a single-board computer (SBC), was assigned just one worker, reflecting its limited capacity and typical use cases where SBCs, such as the Raspberry Pi, are not expected to handle large client loads. The workload settings (TPS) for each device, detailed in Table II, represent the maximum transaction load they could manage as blockchain nodes processing client requests.

Table 2. Transaction Loads In Each Device Per Chaincode

Device	Read	Tokenize	Transfer
1	1500	500	500
2	300	150	150
3	100	50	50

The results are shown in Figure 7, comparing performances across the three devices. As shown in Figure 7a, the high-performance laptop (Device 1) consistently achieves the highest throughput across all chaincode operations, recording 1084.5 TPS for read, 356.8 TPS for tokenize, and 336.8 TPS for transfer. The mid-range laptop (Device 2) delivers lower throughput values, achieving 177.4 TPS, 63.5 TPS, and 19.5 TPS for the same chaincodes. The single-board computer (Device 3), constrained by its limited processing power and storage speed, records peak throughputs of 53.6 TPS, 21.4 TPS, and 18.4 TPS for read, tokenize, and transfer, respectively.

As shown in Figure 7b, Device 1 maintains the lowest latency, with a maximum of approximately 3 seconds for the tokenize chaincode. Devices 2 and 3 exhibit higher latencies,

https://doi.org/10.31436/iiumej.v26i3.3610

although both remain below 17 seconds, even for more complex operations such as tokenization and transfer. While Device 2, the mid-range laptop, experiences higher latency than Device 3, the SBC, it is essential to note that Device 2 handles a significantly larger workload and a greater number of workers than Device 3. In comparison, public blockchains such as Bitcoin and Ethereum typically exhibit much higher latencies, with Bitcoin averaging around 10 minutes per transaction and Ethereum approximately 15 seconds per block confirmation.

To better contextualize CPU performance, CPU utilization was compared across devices at their respective optimal workloads, illustrated in Figure 7c. For the transfer chaincode, Device 1 maintained ~6% CPU usage at 336.8 TPS, Device 2 reached ~25% at only 63.5 TPS, and Device 3 recorded ~23% at 18.4 TPS. While the CPU usage of Devices 2 and 3 is relatively close, their throughput differs significantly, and the effective processing throughput per percentage of CPU usage is far higher on Device 1 due to its faster clock speed, larger cache, and more efficient architecture.

The primary contributors to CPU load were cryptographic signing and verification (ECDSA), chaincode execution involving read/write operations, and ordering service block processing. Two operational profiles emerged from the measurements: (1) blockchain services alone consumed 20–25% of available CPU, and (2) when operating system processes, Docker overhead, and Caliper benchmarking were included, CPU utilization approached maximum capacity. This confirms that CPU contention is a limiting factor during high-TPS workloads.

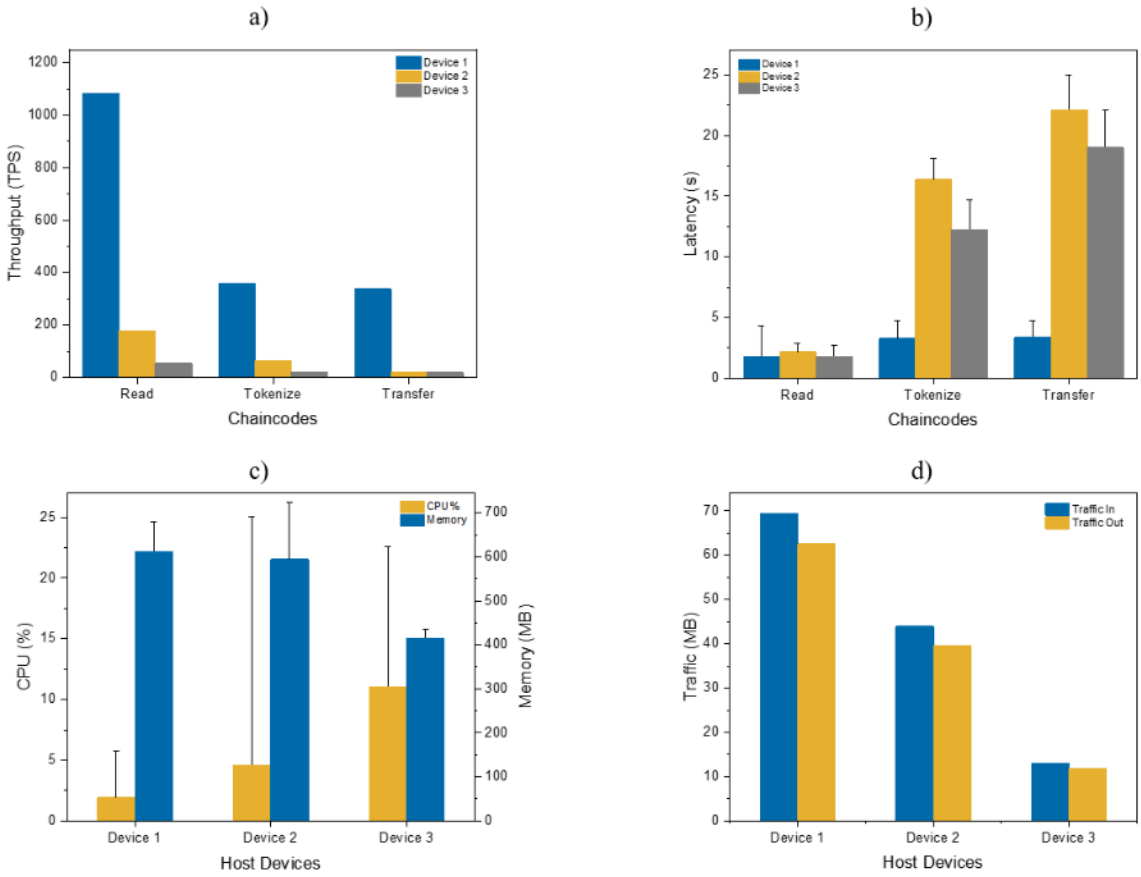

Figure 7. Performance Benchmarking Results: (a) chaincode performance throughput (TPS), (b) chaincode latency (s), (c) hardware usage during benchmarking, and (d) traffic inflow and outflow

https://doi.org/10.31436/iiumej.v26i3.3610

Beyond CPU utilization, network and storage performance played a significant role. All devices were connected via Wi-Fi to a single router, introducing bandwidth sharing and occasional latency spikes. Device 1's SSD allowed faster ledger writes compared to HDD-equipped Device 2, contributing to its lower latency. Device 3's limited network throughput (~4.36 MB/s inflow), combined with slower storage, increased failure rates under heavy load. These factors have direct implications for IoT deployments — in bandwidth-constrained or high-latency environments, transaction confirmation delays could disrupt real-time IoT applications such as sensor event logging or automated control systems.

Transaction payload size and chaincode complexity also impact performance. While the current tests used small payloads, IoT deployments often involve larger datasets or batch transactions. Larger payloads increase serialization, transmission, and cryptographic processing time. Complex chaincodes (e.g., multi-asset transfer with conditional checks) consume more CPU cycles per transaction and extend endorsement times under high concurrency, especially on low-power devices.

Regarding network capability, Figure 7d illustrates the network traffic across devices, measured in megabytes per second (MB/s) over the 3-second benchmark duration. Device 1 exhibits the highest network throughput, with an inflow rate of approximately 23.18 MB/s and an outflow rate of 20.86 MB/s. Device 2 exhibits moderate network usage, with an inflow rate of 14.64 MB/s and an outflow rate of 13.18 MB/s. In contrast, Device 3 experiences significantly lower network traffic, with an inflow rate of 4.36 MB/s and an outflow rate of 3.93 MB/s. This reduced throughput for Device 3 is attributed to increased failure rates when handling high transaction volumes within the short benchmarking period. These results underscore the varying capabilities of each device, suggesting that high-performance nodes are essential for applications with intensive workloads. In contrast, lower-powered devices can still contribute as supportive nodes, effectively balancing the network by handling moderate transactions. The correlation between network throughput and overall TPS suggests that sufficient bandwidth is critical for high-performance blockchain nodes in IoT contexts.

Power efficiency is another critical factor for IoT-blockchain deployments, particularly in regions like Malaysia, where environmental sustainability initiatives aim to reduce energy footprints. Table 1 presents the idle and sustained load power consumption for all devices. When normalized against peak transaction throughput, high-performance devices like Device 1 achieve more TPS per watt under heavy load (≈17.1 TPS/W for the tokenize chaincode) than lower-power devices like Device 3 (≈4.4 TPS/W). This highlights a trade-off: while SBCs consume far less energy overall, their reduced processing capacity lowers TPS efficiency.

From a sustainability perspective, deployment strategies in Malaysia could balance these trade-offs by assigning high-throughput workloads to energy-efficient high-performance nodes and reserving low-power devices for lighter tasks. This approach supports blockchain adoption in green initiatives such as carbon offset tracking, renewable energy certificates, and transparent supply chain management, aligning with Malaysia's environmental goals while minimizing unnecessary energy usage.

4.3. Optimization Considerations

Optimizing heterogeneous networks for IoT requires role allocation that matches device capability. SBCs like Raspberry Pi can serve as commit-only peers or lightweight endorsers to reduce cryptographic load. Switching from CouchDB to LevelDB reduces query latency, and lightweight Docker images with fewer background services lower CPU and memory usage. Data preprocessing at the edge before blockchain submission can also minimize transaction

frequency and payload size. For compute-intensive environments, hardware accelerators such as GPUs or FPGAs could offload signature verification and chaincode execution, boosting throughput without replacing existing hardware.

5. CONCLUSION AND FUTURE WORK

This research explored the performance characteristics of a multi-device Hyperledger Fabric network, deploying a diverse mix of hardware to evaluate blockchain and hardware performance under different configurations. Key insights from the benchmarking tests indicate that while the network demonstrates effective transaction handling across devices, factors such as device processing capabilities and CPU utilization play significant roles in determining overall network efficiency.

One notable observation involves the CPU utilization, which appears modest at approximately 20% - 25% when considering only the blockchain services. However, this measurement does not account for CPU usage by the operating system or the Hyperledger Caliper benchmarking tools. When these additional processes are included, the CPU approaches maximum capacity, indicating that CPU limitations are the primary bottleneck affecting network performance. This contention for CPU resources restricts the system's ability to handle higher transaction volumes efficiently. Optimizing processing capabilities, such as upgrading to more powerful CPUs or enhancing the efficiency of blockchain services, could alleviate these limitations and increase throughput capacity.

Optimization strategies for low-cost devices are crucial to improve further accessibility and adoption of Hyperledger Fabric among SMEs and rural communities. Assigning less resource-intensive node roles, such as peer-only nodes, to low-spec devices, while reserving more demanding services for powerful hardware, can enhance efficiency without increasing cost. Streamlining chaincode logic minimizes computational complexity and reduces the need for large payloads, thereby alleviating the resource burden. Additionally, preprocessing and aggregating data at the edge before submission helps distribute compute load and lessen the number of blockchain transactions required, particularly in resource-constrained environments. Lightweight container configurations and simple consensus mechanisms further contribute to overall system efficiency. These approaches collectively make blockchain deployment more achievable for organizations with limited infrastructure.

Future work could assess hardware configurations and optimization strategies to mitigate CPU bottlenecks and enhance performance. Expanding testing to include more complex or resource-intensive chaincodes may provide insights into how processing demands interact with hardware setups under various conditions. This study provides a foundational understanding that will be valuable for designing Hyperledger Fabric networks on diverse hardware configurations. The planned optimizations are expected to yield further efficiency gains in future iterations.

REFERENCES

[1] Laroui M, Nour B, Moungla H, Cherif MA, Afifi H, Guizani M. (2021). Edge and fog computing for IoT: A survey on current research activities & future directions. Computer Communications, 180, 210-231.

[2] Maraveas C, Piromalis D, Arvanitis KG, Bartzanas T, Loukatos D. (2022). Applications of IoT for optimized greenhouse environment and resources management. Computers and Electronics in Agriculture, 198, 106993.

[3] Kumar NM, Chand AA, Malvoni M, Prasad KA, Mamun KA, Islam FR, Chopra SS. (2020). Distributed energy resources and the application of AI, IoT, and blockchain in smart grids. Energies, 13(21), 5739.

[4] Mathur S, Kalla A, Gür G, Bohra MK, Liyanage M. (2023). A survey on role of blockchain for IoT: Applications and technical aspects. Computer Networks, 227, 109726.

[5] Makhdoom I, Abolhasan M, Abbas H, Ni W. (2019). Blockchain's adoption in IoT: The challenges, and a way forward. Journal of Network and Computer Applications, 125, 251-279.

[6] Ashley MJ, Johnson MS. (2018). Establishing a secure, transparent, and autonomous blockchain of custody for renewable energy credits and carbon credits. IEEE Engineering Management Review, 46(4), 100-102.

[7] Anand P, Singh Y, Selwal A, Alazab M, Tanwar S, Kumar N. (2020). IoT vulnerability assessment for sustainable computing: Threats, current solutions, and open challenges. IEEE Access, 8, 168825-168853.

[8] Siwakoti YR, Bhurtel M, Rawat DB, Oest A, Johnson RC. (2023). Advances in IoT security: Vulnerabilities, enabled criminal services, attacks, and countermeasures. IEEE Internet of Things Journal, 10(13), 11224-11239.

[9] Sharma PK, Kumar N, Park JH. (2020). Blockchain technology toward green IoT: Opportunities and challenges. IEEE Network, 34(4), 263-269.

[10] Honar PH, Rashid MA, Alam F, Demidenko S. (2022). Experimental performance analysis of a scalable distributed hyperledger fabric for a large-scale IoT testbed. Sensors, 22(13), 4868.

[11] Kaushal RK, Kumar N. (2024, March). Exploring hyperledger caliper benchmarking tool to measure the performance of blockchain based solutions. In 2024 11th international conference on reliability, infocom technologies and optimization (trends and future directions)(ICRITO) (pp. 1-6). IEEE.

[12] Introduction - Hyperledger Fabric Docs main documentation. [https://hyperledger-fabric.readthedocs.io/en/release-2.5/whatis.html]

[13] Introduction - Hyperledger Caliper. [https://hyperledger-caliper.github.io/caliper/0.6.0/]

[14] Ucbas Y, Eleyan A, Hammoudeh M, Alohaly M. (2023). Performance and scalability analysis of ethereum and hyperledger fabric. IEEE Access, 11, 67156-67167.

[15] Guggenberger T, Sedlmeir J, Fridgen G, Luckow A. (2022). An in-depth investigation of the performance characteristics of Hyperledger Fabric. Computers & Industrial Engineering, 173, 108716.

[16] Wang C, Chu X. (2020, November). Performance characterization and bottleneck analysis of hyperledger fabric. In 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS) (pp. 1281-1286). IEEE.

[17] Alexandridis A, Al-Sumaidaee G, Zilic Z, Jeon G, Wang J. (2023). An iot ecosystem platform for the evaluation of blockchain feasibility. IEEE Internet of Things Journal, 10(24), 21515-21527.

[18] Honar PH, Rashid M, Alam F, Demidenko S. (2021). Hyperledger fabric blockchain for securing the edge internet of things. Sensors, 21(2), 359.

[19] Iftekhar A, Cui X, Tao Q, Zheng C. (2021). Hyperledger fabric access control system for internet of things layer in blockchain-based applications. Entropy, 23(8), 1054.

[20] Eghmazi A, Ataei M, Landry RJ, Chevrette G. (2024). Enhancing IoT data security: Using the blockchain to boost data integrity and privacy. IoT, 5(1), 20-34.

[21] Xu X, Sun G, Luo L, Cao H, Yu H, Vasilakos AV. (2021). Latency performance modeling and analysis for hyperledger fabric blockchain network. Information Processing & Management, 58(1), 102436.

[22] Khan D, Jung LT, Hashmani MA, Cheong MK. (2022). Empirical performance analysis of hyperledger LTS for small and medium enterprises. Sensors, 22(3), 915.

[23] Foschini L, Gavagna A, Martuscelli G, Montanari R. (2020, June). Hyperledger fabric blockchain: Chaincode performance analysis. In ICC 2020-2020 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE.

[24] Alamsyah A, Hakim N, Hendayani R. (2022). Blockchain-based traceability system to support the Indonesian halal supply chain ecosystem. Economies, 10(6), 134.

[25] Song S, Lu J, Zhao H, Wang W, Shi C, Luo R. (2022, November). Traceability of Product Supply Chain Based on Hyperledger Fabric. In Proceedings of the 4th International Conference on Advanced Information Science and System (pp. 1-6).

[26] PassMark CPU Benchmarks - CPU Test Information. [https://www.cpubenchmark.net/cpu_test_info.html]