





**Please cite the Published Version**

Ucbas, Yusuf , Eleyan, Amna , Hammoudeh, Mohammad  and Alohal, Manar  (2023)  
Performance and scalability analysis of ethereum and hyperledger fabric. IEEE Access, 11. pp.  
67156-67167. ISSN 2169-3536

**DOI:** <https://doi.org/10.1109/ACCESS.2023.3291618>

**Publisher:** Institute of Electrical and Electronics Engineers (IEEE)

**Version:** Published Version

**Downloaded from:** <https://e-space.mmu.ac.uk/634428/>

**Usage rights:**  [Creative Commons: Attribution-Noncommercial-No Derivative Works 4.0](#)

**Additional Information:** This is an open access article which first appeared in IEEE Access

**Enquiries:**

If you have questions about this document, contact [openresearch@mmu.ac.uk](mailto:openresearch@mmu.ac.uk). Please include the URL of the record in e-space. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from <https://www.mmu.ac.uk/library/using-the-library/policies-and-guidelines>)

Received 18 June 2023, accepted 27 June 2023, date of publication 3 July 2023, date of current version 7 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3291618

## RESEARCH ARTICLE

# Performance and Scalability Analysis of Ethereum and Hyperledger Fabric

YUSUF UCBAS<sup>1</sup>, AMNA ELEYAN<sup>1</sup>, (Member, IEEE),  
MOHAMMAD HAMMOUDEH<sup>2</sup>, (Senior Member, IEEE),  
AND MANAR ALOHALY<sup>3</sup>

<sup>1</sup>Department of Computing and Mathematics, Manchester Metropolitan University, M15 6BH Manchester, U.K.

<sup>2</sup>Information and Computer Science Department, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

<sup>3</sup>Department of Information Systems, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

Corresponding author: Manar Alohalay (mfalohaly@pnu.edu.sa)

This project was supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R383), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**ABSTRACT** Blockchain emerged in the last decade as a promising technology with possible applications in numerous fields such as healthcare, supply chain, and finance. Its immutability, transparency, security, and decentralisation gained significant attention in academia and industry. One technology that blockchain can support is the Internet of Things (IoT). However, there are still challenges hindering the real-world adoption of blockchain due to concerns about its performance, scalability, and complexity. This study contributes to a comparative study that analyses blockchain platforms in terms of their performance and scalability, with specific reference to IoT applications. We focus on the Ethereum and Hyperledger Fabric blockchain platforms. As part of the implementation, an IoT healthcare use case is developed. We conducted performance and scalability tests on private platform networks to measure the throughput and latency parameters. To evaluate scalability, we examined the behaviour of the studied platforms in response to an increase in the number and rate of transactions. Hyperledger Caliper is used to collect these parameters. Experiment analysis shows that Fabric outperforms Ethereum in terms of latency and throughput. As for performance and scalability analysis, Fabric was found to be more suitable than Ethereum for private networks such as IoT healthcare ecosystems.

**INDEX TERMS** Blockchain, Ethereum, hyperledger fabric, Internet of Things, performance, scalability.

## I. INTRODUCTION

Blockchain is a promising technology that has gained significant attention in the last decade. Its capabilities have gone far beyond cryptocurrency since Nakamoto introduced it in 2008. It reforms the applications of the existing technologies and creates new application areas that have been thought impractical before. In applications such as healthcare, supply chains, high-level businesses, and the financial industry, blockchain empowers people through recognised identity and asset ownership. The strength of the blockchain comes

from its immutability, transparency, security, and decentralisation [1].

Swan et al. [2] described the blockchain evolution as Blockchain 1.0 being a cryptocurrency with Bitcoin, Blockchain 2.0 being smart contracts with Ethereum, and Blockchain 3.0 being Decentralised Applications (DApps) enabled by smart contracts. Considering this evolution, blockchain capabilities have improved, and leading companies have started blockchain projects worldwide [3]. This has created possible integration with other technologies, such as the Internet of Things (IoT) [4]. IoT is another emerging technology that has yet to reach maturity. With the growth of its applications, IoT's downsides, such as security, privacy, interoperability, heterogeneity, and maintenance, have

The associate editor coordinating the review of this manuscript and approving it for publication was Liang-Bi Chen<sup>1</sup>.

arisen [5]. Blockchain offers a potential solution to alleviate some of the flaws facing IoT applications.

Unlike existing transaction processing systems like VISA, where thousands of transactions per second can be processed in a few seconds, blockchain has serious throughput and latency limitations. For instance, Bitcoin can process seven transactions per second (TPS) in about 10 minutes [6]. After Bitcoin, more efficient platforms were proposed, e.g., Ethereum and Hyperledger Fabric. Nevertheless, their performance, scalability, and complexity have shortcomings that need to be addressed. This paper aims to analyse and compare the Ethereum and the Hyperledger Fabric in terms of their performance and scalability for IoT applications. Ethereum is chosen since it has a wide range of features, including smart contracts and DApps. It is also the second-largest used blockchain platform; thus, it has more resources and support than the majority of platforms. The reason behind the Fabric choice is that it is the leading platform in private blockchains with features such as modularity, plug-and-play ability, and smart contract support. The main contributions of this article can be summarised as follows:

- 1) A comparative analysis of the Ethereum and Fabric networks using a stable benchmark tool, Caliper, under the same controlled environments.
- 2) Empirical methodology to understand the behaviour of these platforms under varying workloads.
- 3) Applied various performance metrics, including throughput, and latency, to perceive the scalability manners of the platforms with increased transaction rates and numbers.
- 4) A proof-of-concept implementation with a use case in the IoT healthcare field to expose the challenges facing the adoption of blockchain in IoT applications.

The rest of this article is organised as follows: Section II provides a review of the related work. Section III presents the background information needed to understand this study. Section IV describes the design choices and decisions and the IoT use case scenario in healthcare. Section V describes how to implement the design choices and contains the experiments conducted. Section VI presents the results of our experiments and evaluates the findings. Lastly, Section VII summarises the conducted work and includes recommendations and possible future works.

## II. RELATED WORK

In the last ten years, a large body of research has focused on blockchain as an emerging technology. These efforts resulted in a wealth of scientific reports, white papers, and community blogs. However, a focused study on particular aspects of this technology is still needed to better understand its practicality, specifically its performance and scalability aspects. Chowdhury et al. [7] proposed a comparative analysis of several leading DLT platforms, including Ethereum, Fabric, and IOT. Unlike other studies, this research did not include only blockchain platforms. Instead, it collected DLT platforms

regardless of their data structures. The authors categorised the analysis into quantitative and qualitative criteria. These criteria vary depending on the platform type and include, among other aspects, scalability and robustness. Although the comparison covered a sufficient number of platforms and criteria, the evaluation done in the work needs deeper analysis.

Several studies investigated the throughput and latency of Ethereum or Fabric private networks, or both. Kuzlu et al. [8] analysed the performance of Hyperledger Fabric, considering its throughput, latency, and scalability. The authors used a Hyperledger Caliper to measure these metrics under the workloads of the Open and Query functions. The Open function includes one read and one write operation per transaction, whereas Query has only one read. Further, they studied the impact of transaction rates, the number of transactions, and multiple simultaneous transactions. Their findings include, the type of transaction highly affects performance, and latency is especially influenced by the increase in simultaneous transactions. However, the experiment only considered low-capability workloads for Open and Query. Testing the network with a transfer operation is a better way to understand how it reacts in more realistic scenarios.

In [9], the authors considered a private Ethereum deployment in addition to the Hyperledger Fabric. For the analysis, they used simple smart contracts that create accounts, issue money, and transfer money. The authors followed a different approach for measuring the performance metrics than the one used in [8]. According to their findings, Fabric outperforms Ethereum in all performance metrics. The method they used for data collection differs for the platforms which might result in an unhealthy evaluation.

In [10], the authors identified IOTA as the most suitable blockchain platform for the IoT ecosystem owing to its feeless transactions, potential higher TPS values, and lower energy consumption. However, they pointed out decentralisation and scalability as the main limitations of IOTA. IOTA smart contracts, which brought functionalities with the release of IOTA 2.0, were reexamined for scalability. IOTA was proven more scalable than its popular alternative, Ethereum, during comparisons that draw attention to the execution of smart contracts. IOTA executes smart contracts in a parallel manner, whereas in Ethereum, they are executed by the whole network. Although this paper compares different blockchain platforms in terms of scalability and performance, it lacks experiments and data. On the other hand, the authors of [11] investigated IOTA for offline scalability manners. The findings reveal limitations in IOTA's offline transaction capabilities, emphasizing the need for a more scalable offline blockchain solution.

Reviewing existing studies showed that most blockchain performance comparison experiments did not ensure a stable environment with consistent conditions across platforms. This article addresses this gap as described in Sections Section IV and Section V.

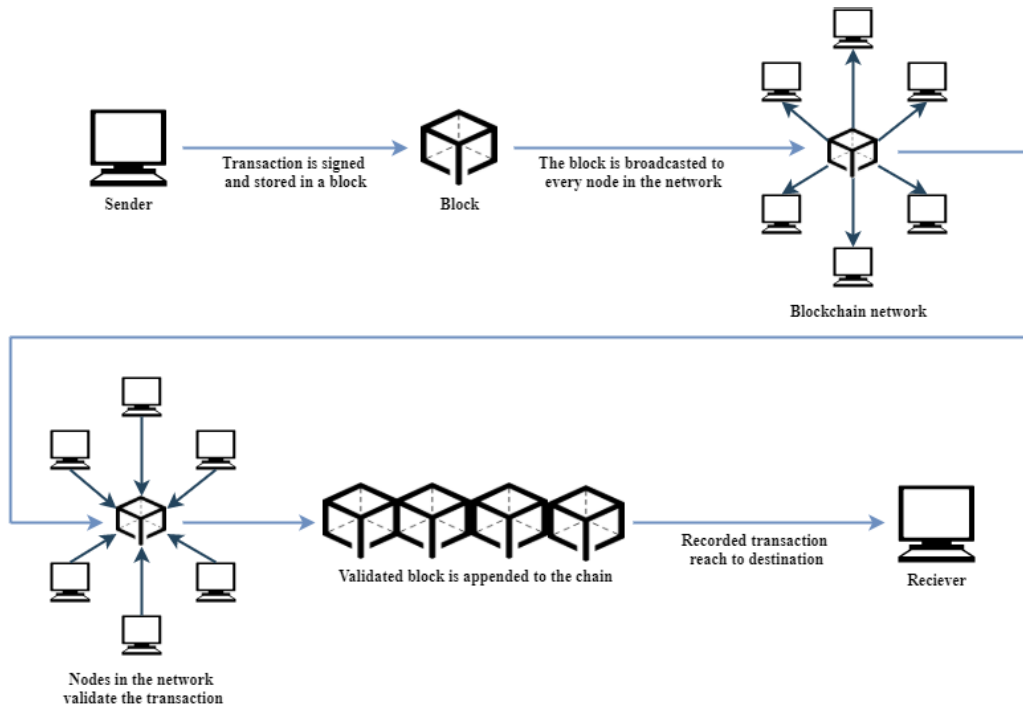


FIGURE 1. Blockchain working mechanism.

### III. BLOCKCHAIN CONCEPT

Blockchain is the underlying technology of Bitcoin proposed in 2008 by an anonymous entity called Nakamoto. Nakamoto developed the idea of digital currency in peer-to-peer systems without a centralised authority. The main motivation was to dispose of a financial institution's increasing transaction costs and minimum transaction size limits.

According to Nakamoto's blockchain, every transaction created is stored as blocks that form a chain. The blocks are appended to the chain in a linear, growing manner with cryptographic connections. Each block references the hash of the previous block. Therefore, a hash chain that provides immutability for the data in the ledger is created. Furthermore, transactions are validated by distributed consensus algorithms as an additional security measure.

In general, blockchain has key merits regarding its architectural advantages. The distributed P2P network makes it decentralised and decreases server costs and bottlenecks in traditional central server models. The second merit is that the data in transactions is hard to tamper with thanks to its distribution across the whole network and the validation process. Blockchain also provides anonymity since each user's identity is defined by their generated addresses. Lastly, owing to the validation process and timestamps that are used in recording, transactions can be traced and audited [12].

On the other hand, the usage of the blockchain has gone beyond cryptocurrency due to its high potential. Especially with the foundation of Ethereum in 2014 by Buterin [13], blockchain gained a high level of programmability and

autonomy. He proposed that smart contracts be embedded into the system in order to execute if a transaction triggers a specific condition written into the contract. Therefore, Ethereum has become a platform for decentralized applications that make use of smart contracts as their logic.

Novo et al. [14] described transactions as the transfer of values between entities. The transactions are grouped together to form a block. Each transaction is signed by its owner and broadcast to the network for validation. The digital signatures are used for the signing and verification phases [15]. In particular, a transaction is encrypted with the owner's private key in the signing phase. The signed transaction becomes visible to all of the nodes through the network. There is a particular node or set of nodes in the blockchain structure, called miners. The miners compute a cryptographic puzzle to validate the transaction. Once a miner solves the puzzle, it broadcasts the solution to other nodes. Other nodes in the network confirm it, and the transaction is validated. Thus, the consensus is achieved among networks. Finally, the receiver can access the value of the validated transaction and check its integrity by decrypting the data with the sender's public key [15]. A visual summary of the blockchain's working mechanism can be seen in Fig. 1.

#### A. ETHEREUM

Ethereum is the second-largest blockchain after Bitcoin in terms of its usage as a cryptocurrency. However, Ethereum's capabilities are not limited to cryptocurrencies. According to its white paper [13], the ideas behind the creation of Ethereum

were building decentralised applications, assuring efficient trade-offs between these applications, and providing security for small-scale applications. These ideas were implemented by presenting a Turing-complete programmable blockchain that enables anyone to write smart contracts and decentralised applications. Ethereum has three key features that, taken together, make it unique. These are smart contracts, decentralized applications (DApps), and Ethereum virtual machines (EVM).

Smart contracts are programmes built on the blockchain and executed only when a specific condition is met. They are executed by the EVM, which runs on every participant's computer in the network. DApps make use of smart contracts as the logic of applications. Considering the typical architecture of applications, DApps have several advantages, including resilience, transparency, and censorship resistance [16].

## B. HYPERLEDGER FABRIC

Hyperledger is an open-source umbrella project hosted by the Linux Foundation. It aims to develop enterprise-grade blockchain technologies [17]. Fabric is one of the sub-projects in Hyperledger. It provides modular distributed ledger technology for various industrial uses [18]. There are several critical pillars of Fabric that make it valuable across blockchain platforms. They are permissioned blockchain, chaincode, modularity, and certificate authorities (CAs).

Fabric is a permissioned blockchain, which means only authorized entities can be part of the network. The permissioned architecture allows businesses to remain private, confidential, and robust scalability. Unlike permissionless blockchains, these features could be desirable for enterprise uses [18]. The smart contracts in Fabric are called chaincode. The distributed applications that run in Fabric can be written in standard programming languages [19]. Fabric also provides a plug-and-play consensus structure. This modularity allows custom usage of consensus mechanisms to fit particular use cases. Since Fabric is a permissioned blockchain, it requires some secure authentication and authorization mechanisms. The three types of certificates that improve the security of Fabric are TLS certificates, enrollment certificates, and transaction certificates. Enrollment certificates are used to connect to the network, while transaction certificates are needed for submission [20].

## C. BLOCKCHAIN PARAMETERS

There are a number of parameters to be considered when analysing blockchain platforms. At the broadest classification, these platforms are separated into public and private DLTs. The main difference is that users' identities are controlled by an organisation in Fabric, whereas any user can join Ethereum freely. Therefore, confidentiality and transparency are the parameters of the security aspect affected by this classification. Confidentiality is the state of being private for the data, whereas transparency is defined as visible data for the view. Another parameter is availability, which is

**TABLE 1. Comparison between Ethereum and Hyperledger Fabric.**

Aspect	Ethereum	Hyperledger Fabric
Consensus Algorithm	Proof of Stake (transitioning from Proof of Work)	Practical Byzantine Fault Tolerance (PBFT)
Scalability	Moderate	High
Smart Contracts	Yes	Yes
Privacy	Limited	Flexible (configurable privacy options)
Governance	Decentralized governance model (EIPs, Core Devs)	Consortium governance model
Permissioning	Public blockchain by default, can have private chains	Flexible permissioning (public, private, consortium)
Performance	Lower performance due to higher decentralization	Higher performance due to PBFT consensus

the platform's availability when there are not several active nodes in the network. In this comparison, authentication is the method of secure verification of the participant's identity. Lastly, any technique for resistance to quantum computing attacks is considered one of the security parameters.

Under the aspect of functionality, supporting the smart contract usage and the type of consensus algorithms play a significant role. In addition, some platforms provide plug-and-play component options, which can be defined as modularity. As the last functionality parameter, the ability of the nodes to manage their identities to control the network can be presented.

The applicability covers several performance parameters in addition to other parameters that affect the pertinence of the platforms. Scalability is thought of as the reaction of the platform when the network grows in the number of nodes and workload. The existence of transaction fees also has an impact on this aspect. An overall comparison of Ethereum and Hyperledger Fabric is demonstrated in Table 1.

## IV. DESIGN CHOICES

Section III-C listed all the parameters for analysing the platforms under three aspects, some of which are outside the scope of this study. Selected parameters are focused on for in-depth analysis and comparison of the chosen platforms. This subsection thus explores and presents the parameters to be implemented. The parameters are chosen considering their applicability. Each platform is tested in terms of these parameters. The applicability of the platform will be evaluated through its performance and scalability. It is believed that performance and scalability are the two strongest considerations for a DLT platform. They are directly related to the possibility of a platform's adoption in a domain. Low performance and scalability limit the usage of a DLT platform.

### A. TECHNICALITY OF THE PARAMETERS

Elaborating on the determined parameters for implementation, the performance of the platform is investigated on



two metrics of transactions, namely, throughput and latency. Throughput is defined as the number of transactions per second successfully processed by the blockchain network. A transaction is successfully processed when it is included in a block and committed as part of the ledger. On the other hand, latency is the time it takes for a client to obtain a response after sending a request. These two metrics are the core of the experiment since they are related to other parameters as well during the whole process. The parameters are as follows [21]:

$$L = t_c - t_s \quad (1)$$

In Equation (1),  $L$  refers to transaction latency, where  $t_c$  is the confirmation time at the network threshold and  $t_s$  is the submit time.

$$T = n_c - t \quad (2)$$

In Equation (2),  $T$  refers to transaction throughput, where  $n_c$  is the number of total committed transactions and  $t$  is the total time in seconds. Throughout the experiments, transaction throughput and latency are measured as direct indicators of performance. As for scalability, it is measured by an increased number of transactions and transaction rates in the network. An illustration of the parameters' relationship can be found in Fig. 2.

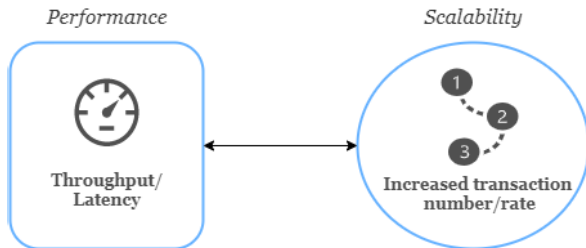


FIGURE 2. Relationship of the Parameters.

## B. SYSTEM ARCHITECTURE DESIGN

Although each platform's system contains several different components, the general architecture could be described with common components. In this study, the laptop used as the host has the following specifications:

*Intel Core i7-5500U CPU @ 2.40GHz, 12GB RAM, 1TB HDD*

As shown in Fig. 3, the laptop is used as the host for the virtual machine (VM). It is practical to deploy the network in a VM because of the isolation of the environment it offers, and resource utilisation is vital for the sake of the experiments. Therefore, it is important to keep the host untouched during the experiments to prevent any system failures. Moreover, using a VM gives the ability to create instances and take snapshots of the machines. When a failure pops up or tests need to be repeated, a snapshot of the working VM could be reloaded quickly.

The created VM was equipped with the necessary software and tools for the planned experiments. The experiments are conducted separately for each DLT platform. In other words, a snapshot of the same VM is used in turn for Ethereum and Fabric. Once the necessary software is installed and configurations are made, the private networks of Ethereum and Fabric are launched. These networks are the platforms where the performance and scalability experiments are conducted. A benchmark tool, as shown in Fig. 3, is used to compare and analyse the platforms. This tool generates the throughput and latency results of the investigated platforms under varying workloads.

As an overview of the approach conducted, each platform is evaluated in a local private environment. Fabric is already a private DLT platform, and Ethereum is known as the public one. However, private networks have greater performance and scalability values than public ones. It would not be fair to evaluate them in this way. Thus, the experiments are done regarding private networks since Ethereum also supports local private networks. They each have different architectures, configurations, and software requirements. However, certain parameters should be the same for the sake of the research. Therefore, the same workloads in a simple use case of DLT platforms are needed to be applied. In order to find an answer to the research question, quantitative data comprise primary experimental data and secondary data extracted from the literature are used. The experimental data is generated by manipulating certain parameters under controlled conditions.

Once the test environment is set up, the experiments are carried out. However, to measure the determined parameters in Section IV-A, a testing tool is needed. At this point, Hyperledger Caliper<sup>1</sup> is the official benchmark tool developed by Huawei. Caliper is a blockchain performance benchmark framework, which allows users to test different blockchain solutions with predefined use cases and get a set of performance test results. Currently, it supports several blockchain solutions, including Ethereum and Fabric. Caliper provides performance indicators such as throughput, latency, success rate, and resource utilisation. The architecture of Caliper is demonstrated in its official documentation. It requires benchmark and network configuration files as inputs, as well as Workload modules and benchmark artefacts. Then, it creates a report of the system under test [22].

## C. USE CASE SCENARIO: IoT HEALTHCARE APPLICATION

IoT is an emerging technology that promises opportunities in a large number of critical domains. However, due to its nature, it can not satisfy some security, traceability and interoperability features [23]. At this point, IoT can take advantage of DLT platforms. The architecture explained in the previous sections is designed considering IoT use cases. One of the domains where IoT and DLT platform cooperation is highly needed in healthcare. Especially in remote patient monitoring, the data collected from IoT devices such as wearables and biosensors

<sup>1</sup><https://github.com/hyperledger/caliper>

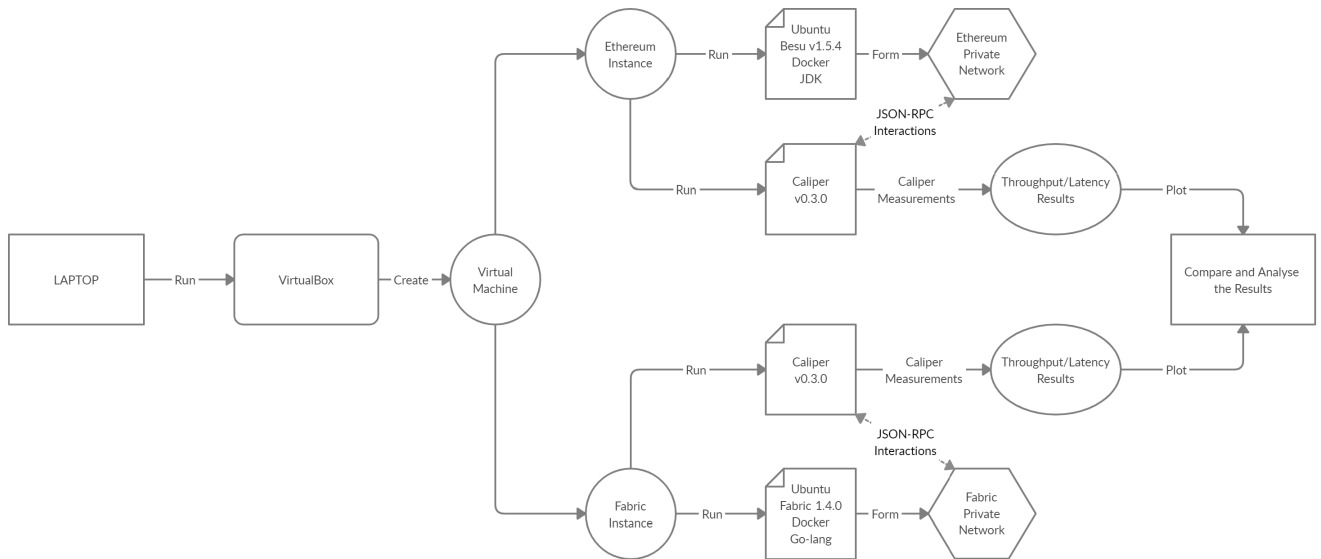


FIGURE 3. General Framework.

requires secure transfer to a healthcare centre and collective movement of the devices [24]. The record of a patient's medical data is extremely sensitive since it is directly related to the patient's health. The data must be transmitted to the physician who monitors and administers the patient for it to be correctly interpreted. Additionally, no one else should be able to seize it and use it maliciously.

In this use case, a patient is equipped with a wearable sensor that generates data about his blood pressure. It is measured periodically and transmitted to his physician. If it exceeds the predetermined values, the physician takes action. To fit our design architecture to this use case, the VM is pretended as an IoT device that runs a node in either a private Ethereum or Fabric network in Fig. 4. Then the generated data from the IoT device is transferred from this node to another node, which is a physician's computer via the DLT platform network. This process is represented in the rounds called "Transfer" during the implementation. The DLT platform network could be the private network of a healthcare centre. Another entity that could access the network would be an insurance company. The patient records could be stored in the chain, thus becoming immutable. Correct billing could be provided in this way. Considering the above scenario, the throughput and latency of the network are measured under the described conditions. The performance and scalability of Ethereum and Fabric are tested to determine which one is more suitable for such an IoT application scenario.

## V. IMPLEMENTATION AND EXPERIMENTS

This section presents the implementation of the design choices described in the previous section. According to the design, how the work is conducted is explained. The necessary configurations to set up the environment, software,

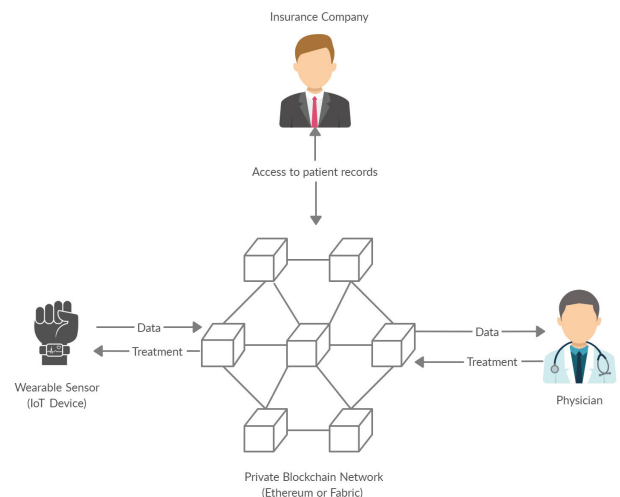


FIGURE 4. Healthcare IoT Use Case.

and tools with the proper codes are concise. Afterwards, the results produced from the experiments are demonstrated for evaluation.

The steps to implement the design are:

- 1) Setup the environment for each DLT platform, namely, Ethereum and Hyperledger Fabric.
- 2) Setup the benchmarking tool, namely, Hyperledger Caliper.
- 3) Bind the DLT platform network with Hyperledger Caliper.
- 4) Write the codes for the configuration files necessary for Hyperledger Caliper.
- 5) Pass the files to Hyperledger Caliper and make the tests.
- 6) Display the results produced from the tests.

The experiments are carried out on a VM running Ubuntu 20.04 LTS. Since setting up a private DLT platform network and analysing it is a complex task, it is described here. Due to the nature of the platforms, they have a few different prerequisites, binaries, and Docker images; thus, they are dealt with separately. Git, cURL, Docker Engine, Docker-compose, Golang, nodejs, npm, Python, and the Java Development Kit are the main tools and software necessary.

The next thing to do is bind Caliper to the platform's environments. Caliper defines the platform's environment as a system under test (SUT) and requires its type and version. In this implementation, Hyperledger Besu 1.5.4 and Hyperledger Fabric 1.4.0 are used as SUTs.

Subsequently, the only thing that is left before running a benchmark is creating configuration files. There are two types of files, namely benchconfig and networkconfig. Caliper requires them in order to run a benchmark and generate throughput and latency values.

### A. BENCHMARK CONFIGURATION

Benchconfig is the benchmark configuration file responsible for the execution of the defined workload and the collection of the results. It contains three types of settings: test, observer, and monitor. Under the test set, the send rate of the transactions, the number of transactions, and the type of rounds can be defined. On the other hand, monitor and observer settings deal with monitoring and observing, as their names indicate. The benchmark configuration file is independent of the type of SUT [25]. Thus, there is no harm in this file being the same for both Hyperledger Fabric and Besu. Moreover, it would be more beneficial to have the same configuration for healthy evaluation.

Fig. 5 is a screenshot of the part of the created benchmark configuration file called 'config.yaml'. Three types of rounds, open, query, and transfer are written in the file. Transaction numbers are set to 100 per second, while transaction rates are set to 50 per second for each type of round. In this 'yaml' file, callback functions refer to javascript files where the behaviours of the rounds are coded.

### B. NETWORK CONFIGURATION

Unlike the benchmark configuration file, the network configuration file must be distinct for different DLT platforms. Therefore, particular files are created for Hyperledger Fabric and Besu to be run by Caliper.

#### 1) FABRIC'S NETWORK CONFIGURATION FILE

The file created for Fabric is called 'fabric-go.yaml'. Within this file, Docker images are run in order to raise up the network by using the commands in 'start' and 'end' sections. This can be seen in Fig. 6, as well as network information. The network configuration file also contains client, channel, chaincode, organisation, orderer, peer, and certificate authority information. However, as it can be seen in Fig. 7, a channel is created, and two peers from two organisations join it.

```
rounds:
- label: open
  description: opening of an account through the deployed chaincode
  txNumber: 100
  rateControl:
    type: fixed-rate
    opts:
      tps: 50
  arguments:
    money: 10000
  callback: benchmarks/scenario/simple/open.js
- label: query
  description: query performance of the deployed chaincode
  txNumber: 100
  rateControl:
    type: fixed-rate
    opts:
      tps: 50
  callback: benchmarks/scenario/simple/query.js
- label: transfer
  description: transferring value between accounts
  txNumber: 100
  rateControl:
    type: fixed-rate
    opts:
      tps: 50
```

FIGURE 5. Benchmark Configuration File.

```
name: Fabric
version: "1.0"
mutual-tls: false

caliper:
  blockchain: fabric
  command:
    start: export FABRIC_VERSION=1.4.0;export FABRIC_CA_VERSION=1.4.0;
    docker-compose -f networks/fabric/docker-compose/2org1peer1couchdb/
    docker-compose.yaml up -d;sleep 3s
    end: docker-compose -f networks/fabric/docker-compose/2org1peer1couchdb/
    docker-compose.yaml down;(test -z "$(docker ps -aq)" || docker rm
    $(docker ps -aq);(test -z "$(docker images dev* -q)" || docker rmi
    $(docker images dev* -q);rm -rf /tmp/hfc.*

info:
  Version: 1.4.0
  Size: 2 Orgs with 1 Peer
  Orderer: Solo
  Distribution: Single Host
  StateDB: CouchDB
```

FIGURE 6. Fabric's Network Configuration File - 1.

```
channels:
  mychannel:
    configBinary: networks/fabric/config_solo/mychannel.tx
    created: false
    orderers:
      - orderer.example.com
    peers:
      peer0.org1.example.com:
        eventSource: true
      peer0.org2.example.com:
        eventSource: true

chaincodes:
  - id: simple
    version: v0
    language: golang
    path: fabric/scenario/simple/go
```

FIGURE 7. Fabric's Network Configuration File - 2.

Furthermore, a chaincode called 'simple' is deployed to the network.

#### 2) BESU'S NETWORK CONFIGURATION FILE

The file created for Besu is called 'networkconfig.json'. Similar to Fabric, the network is raised up by bypassing docker commands in 'start' and 'end' sections. Fig. 8 illustrates the docker-compose commands and information about the nodes and smart contracts in the network. It includes the address of the node and the path to the smart contract. It is significant to mention that the Ethereum network is initialised with a



```
{
  "caliper": {
    "blockchain": "ethereum",
    "command": {
      "start": "docker-compose -f networks/besu/inode-clique/docker-compose.yml up",
      "end": "docker-compose -f networks/besu/inode-clique/docker-compose.yml down"
    }
  },
  "ethereum": {
    "url": "http://localhost:8545",
    "contractDeployerAddress": "0x0D1cf9D73a910E6636c2bb0608a5FdF9F0Eac09",
    "contractDeployerAddressPrivateKey": "0x797c13f7235c627f6bd013dc17ff4c12213ab49a",
    "fromAddressSeed": "0x3f841bf589fd83a521e55d51afddc34fae5351161ead24f664855f529",
    "transactionConfirmationBlocks": 2,
    "contracts": {
      "simple": {
        "path": "src/besu/simple/simple.json",
        "gas": {
          "open": 45000,
          "query": 100000,
          "transfer": 70000
        }
      }
    }
  }
}
```

**FIGURE 8.** Besu's Network Configuration File.

[illegible]

**FIGURE 9. Besu's Genesis File.**

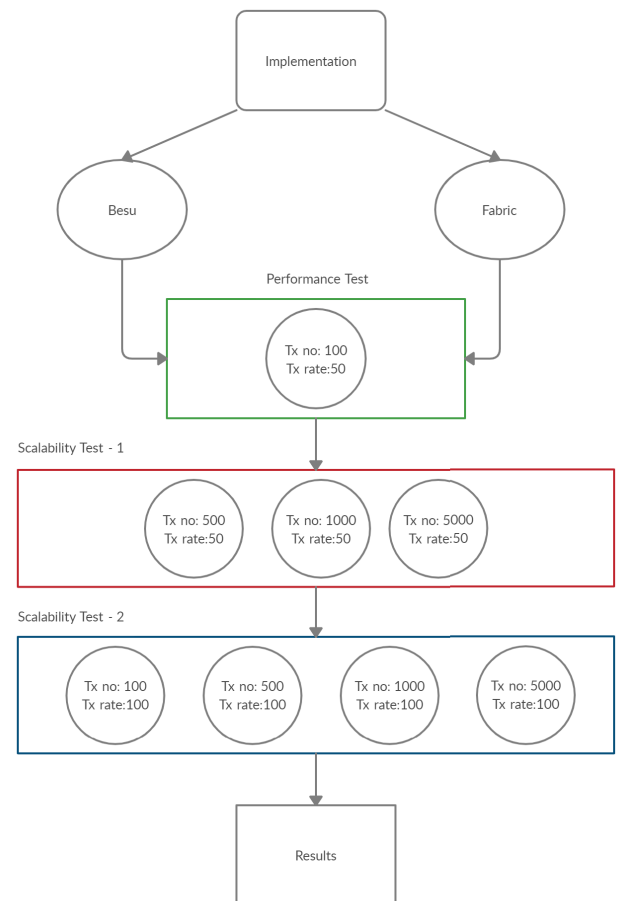
file called ‘genesis.json’. As it can be seen in Fig. 9, the file has network information such as network ID (48122) and consensus type (Clique) and a set of parameters that define the network, such as gas limit and difficulty.

### C. EXPERIMENTS

By passing the configuration files to Caliper and launching it, a report is generated with throughput (tps) and latency (s) values. Although these are the core performance metrics, they are also indicators of the scalability of DLT platforms. When the number of transactions in the network increases, the reaction of these metrics determines how well the platforms scale. Therefore, the benchmark configuration file is manipulated during the scalability test.

## 1) PERFORMANCE TESTS

Apparently, the first test to be performed is the performance test. It is evaluated considering the throughput and average latency values generated for Fabric and Besu by Caliper. The benchmark configuration file in Fig. 5 is applied as it is with three rounds; open, query, and transfer. This means the transaction number would be 100 while the transaction rate would be 50. These three sets of rounds are repeated 10 times to obtain average values in order to eliminate instant misleading values. Fig. 10 shows a diagram of the implementation that contains the performance part.



**FIGURE 10. Implementation Diagram.**

## 2) SCALABILITY TESTS

For scalability measurements, the number of transactions and the transaction rates in these rounds have increased. The transaction numbers rise to 500, 1000, and 5000 in turn. With these transaction numbers and 50 and 100 transaction rates, the above-mentioned process is performed again in order to perceive the scalability manners of the platforms. The scalability tests can be seen in Fig. 10. The results are presented in the following section.

## VI. RESULTS AND EVALUATION

The previous section described how to implement the suggested design choices. According to the implementation, the findings are presented in this section. They are supported by associated graphs. The results are interpreted and discussed. In this way, the comparison and analysis of Hyperledger Fabric and Besu are put into practice. As an evaluation method for the findings, secondary data from the related works are used as well.

### A. PERFORMANCE TEST RESULTS

Considering open rounds, Besu has an average latency of 5,06s while Fabric exhibited less latency with 1,09s.

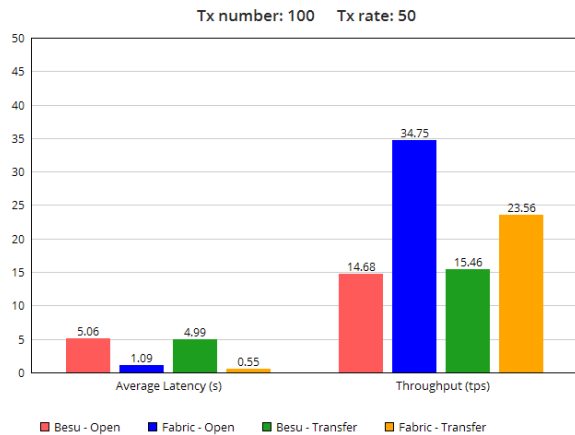


FIGURE 11. Performance Test Results.

In transfer rounds, Besu remains almost the same with 4, 99s and Fabric decreases by nearly half to 0, 55s. Throughput results are also differentiated for the platforms. Besu's throughput accounts for 14, 68 and 15, 46 in open and transfer rounds, respectively. On the other hand, Fabric's values correspond to 34, 75 for the open round and 23, 56 for the transfer round. These results are presented in Fig. 11.

### B. SCALABILITY TEST RESULTS

As shown in Fig. 10, there are two types of scalability tests. The first one is the fixed 50 transaction rate with an increased number of transaction numbers 500, 1000, and 5000. The second one has the same transaction numbers with a fixed 100 transaction rate.

Fig. 12 displays the test results in the open round with a transaction rate of 50. The first three average latency values for Besu increase by nearly 3, starting with 5.06s. However, the last value is the highest at 96.58s. On the other hand, the average latency values for Fabric change between 1.09s and 0.46s. Dealing with throughput numbers, Besu has the lowest value of 11.7 with 5000 transactions and the highest number of 31.84 with 1000 transactions in the benchmark configuration file. Differently, with the exception of the first throughput, Fabric has similar results, which are around 49 for throughput.

Fig. 13 presents the test results in the transfer round with a transaction rate of 50. The average latency values change between 4.24s and 5.14s for Besu whereas they vary between 0.43s and 0.77s in Fabric. While transaction numbers are increasing, throughput values are also increasing for both platforms. It is from 15.46 to 47.6 when Besu is conducted and 23.56 to 49.2 in the case of Fabric.

The results of the scalability test, which was done with a fixed transaction rate of 50, are shown above. However, starting from this point, the transaction rate is 100, and the test results will be prompted in line with it. Before highlighting the results, it is important to note that Caliper was unable to measure the average latency and throughput values for Besu

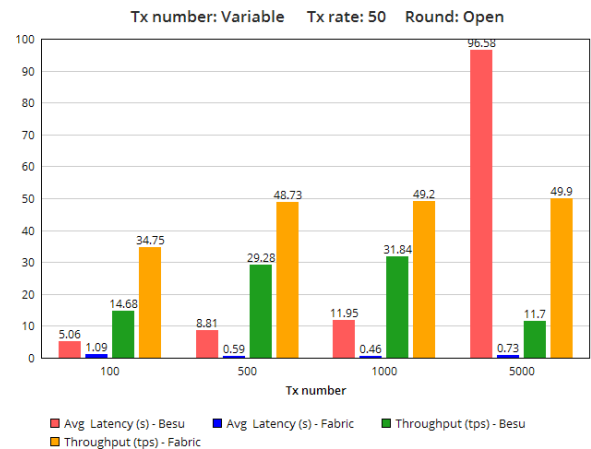


FIGURE 12. Scalability Test - Open - Tx rate: 50.

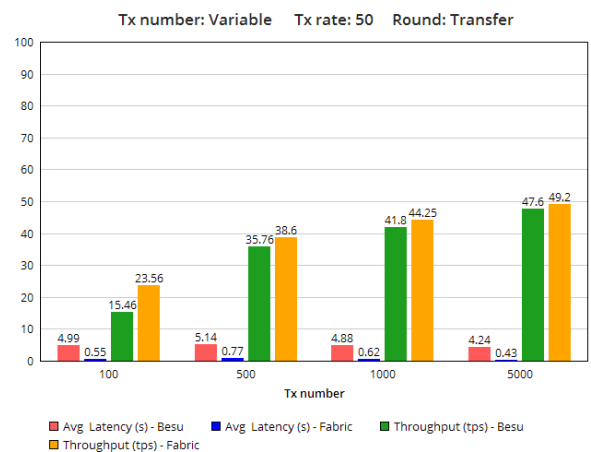


FIGURE 13. Scalability Test - Transfer - Tx rate: 50.

when the transaction number equals 5000. Therefore, they are shown as 0 in the diagrams.

Fig. 14 displays the test results in the open round with a transaction rate of 100. The average latency value of Besu starts at 5s and goes up to 21.18s, whereas Fabric follows a relatively stable trend of around 2.2s. Considering the throughput findings, Besu reaches around 26 when the transaction number is 500 or 1000, while it is 15.23 with a transaction number of 100. Differently, Fabric's throughput numbers have grown from 36.29 to 65.5.

Fig. 15 shows the test results in the transfer round with a transaction rate of 100. The average latency value for Besu increases by about 3 starting at 4.37s. On the other hand, Fabric's average latency values vary between 1.02s and 3.02s. From a throughput perspective, both platforms grow when the number of transactions rises. This rise is from 14 to 45 in Besu, whereas in Fabric it is from 19.59 to 46.2.

### C. EVALUATION

In the previous section, the data from the tests was presented. In this section, they are evaluated. Starting with the

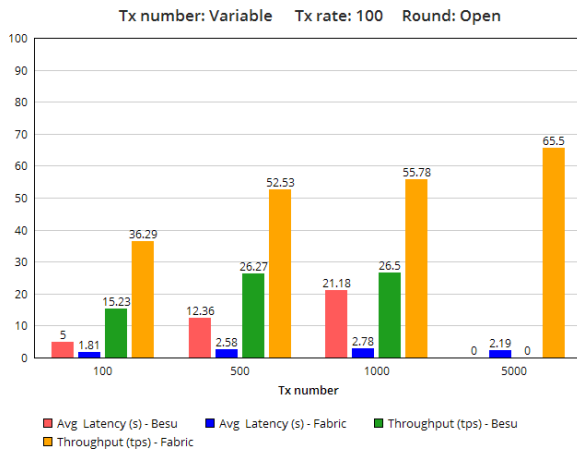


FIGURE 14. Scalability Test - Open - Tx rate: 100.

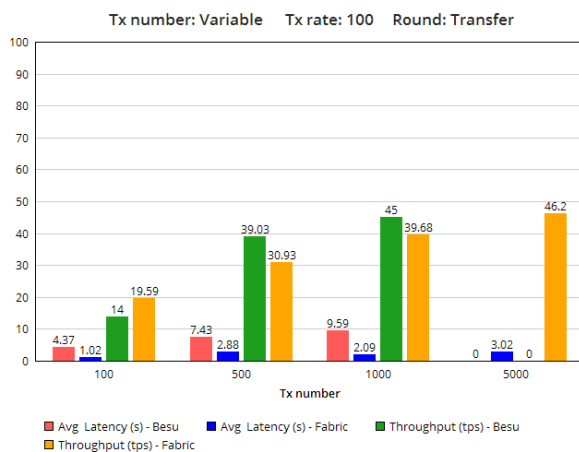


FIGURE 15. Scalability Test - Transfer - Tx rate: 100.

performance test results in Fig. 11, the average latency of the Fabric network for Open and Transfer rounds is almost 4s less than Besu values. This implies that Fabric's throughput values are greater than Besu's. The difference is drastic in Open rounds, which average almost 20 transactions per second. However, in Transfer rounds, Fabric leads with 8 transactions per second. This means Fabric performs better than Besu in general, and the difference is greater when the platform performs simple tasks like opening accounts. In other words, the difference in the performance indicators between the types of operations in Besu is minor.

In order to understand the scalability manner of the platforms when the number of transactions increases with the fixed transaction rate of 50, it is necessary to take a glance at Fig. 12 and Fig. 13. Considering the average latency values, there is no increasing or decreasing pattern for Fabric in both Open and Transfer rounds. With each transaction number, almost all average latency is below 1s. These kinds of balanced values exist in Transfer rounds for Besu. However, they have around a 4s disadvantage in terms of average latency when compared to Fabric. This down-

side of Besu is even more apparent in Open rounds. Thus, when the number of transactions increases, the difference between the average latency values of the platforms also increases. In particular, with the transaction number of 5000, Caliper generates a 96.58s average latency value for Besu, which is an unrealistic value and should be left for further research.

Moving on to the throughput evaluation for Fig. 12 and Fig. 13, Fabric has better values no matter what the transaction number is. However, a remarkable fact from the results is that the difference between Fabric and Besu in Transfer rounds is not as much as in Open rounds. Indeed, there are only minor throughput differences in these rounds. Specific to the platforms, Fabric scales well when the number of transactions increases. Significantly, after 500 transaction numbers, the throughput values improved.

Lastly, to perceive the effect of the increasing transaction rate on the scalability of the platforms, it is necessary to examine the results presented in Fig. 14 and Fig. 15. It is significant to mention one more time that Caliper was unable to generate results for 5000 transaction numbers for Besu. Therefore, there is no comparison between platforms in 5000 transaction numbers. Examining the Open rounds, there is a similarity in the values of Besu and Fabric with the rest of the results. They are better in favour of Fabric. However, the rise in average latency values when the transaction rate increases is significant. The average latency values are nearly doubled for Fabric, whereas Besu does not have such a drastic increment until the transaction number reaches 1000. Finally, the most attention-grabbing finding is that Besu has higher throughput values than Fabric in Transfer rounds when the number of transactions increases.

According to the experimental results, the key findings can be summarised as follows:

- 1) Fabric outperforms Besu (Ethereum's Client) nearly in all tests.
- 2) When the transaction number increases, throughput increases as well.
- 3) Besu was unable to work above 5000 transactions with a transaction rate of 100.
- 4) When the transaction rate increases, Fabric's performance decreases more than Besu's.
- 5) The average latency of Besu is always higher than Fabric's.
- 6) The scalability manners of platforms are similar.
- 7) Both DLT platforms are below the average throughput values, considering existing systems.

## VII. CONCLUSION AND FUTURE WORK

This study was set to compare and analyse the performance and scalability features of two prominent blockchain platforms, namely, Ethereum and Hyperledger Fabric. The main focus of this research was on the latency and throughput parameters of private Ethereum and Fabric networks.

Hyperledger Caliper was chosen as the tool for generating these parameters by integrating it with the private Ethereum and Fabric networks. As these parameters were the direct indicators of performance, scalability tests were conducted by examining the transaction number and rate increase. Specifically, the performance tests are performed with fixed 100-transaction and 50-transaction rates for both networks. On the other hand, two types of scalability tests were taken place; the first one kept the transaction rate at 50 while increasing the transaction number in the order of 500, 1000, and 5000. The second scalability test set the transaction rate to 100, while the transaction numbers were raised in the order of 100, 500, 1000, and 5000. Each test was repeated 10 times to obtain average values. Repeating tests prevented instant peaks in the results.

Based on the empirical results, the throughput values of Fabric are greater than those of Ethereum, whereas the latency values are shorter. Another finding is that Fabric's performance decreases more than Ethereum's with increasing transaction rates. In general, Fabric outperforms Ethereum in terms of performance, but their scalability manners are similar. The organisations should determine their needs and priorities while choosing the DLT platform. Regarding the IoT healthcare scenario in Section IV-C, Fabric would be a more suitable DLT platform since average latency and throughput could be vital for the health of the patient.

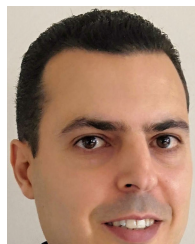
The number of tests conducted and the running of experiments in the same local network are the main limitations of the work. To enhance this study and expand its scope, the tests might be performed many more times in a more stable distributed environment, which ensures better results. In future research, we plan to include IOTA due to its promising scalability feature in addition to Ethereum and Fabric. We also plan to evaluate them in terms of security, performance and scalability. This broader analysis will contribute to a more comprehensive understanding of the strengths and weaknesses of these blockchain technologies.

## REFERENCES

- [1] S. Underwood, "Blockchain beyond Bitcoin," *Commun. ACM*, vol. 59, no. 11, pp. 15–17, Oct. 2016.
- [2] M. Swan, *Blockchain: Blueprint for a New Economy*. Sebastopol, CA, USA: O'Reilly, 2015.
- [3] S. M. Alshurafa, D. Eleyan, and A. Eleyan, "A survey paper on blockchain as a service platforms," *Int. J. High Perform. Comput. Netw.*, vol. 17, no. 1, p. 8, 2021.
- [4] D. Unal, M. Hammoudeh, M. A. Khan, A. Abuarqoub, G. Epiphaniou, and R. Hamila, "Integration of federated machine learning and blockchain for the provision of secure big data analytics for Internet of Things," *Comput. Secur.*, vol. 109, Oct. 2021, Art. no. 102393, doi: 10.1016/j.cose.2021.102393.
- [5] T. Alsoubi, M. Hammoudeh, Z. Bandar, and A. Nisbet, "An overview and classification of approaches to information extraction in wireless sensor networks," in *Proc. 5th Int. Conf. Sensor Technol. Appl.*, vol. 255, 2011, pp. 1–12.
- [6] J. Yli-Huoma, D. Ko, S. Choi, S. Park, and K. Smolander, "Where is current research on blockchain technology?—A systematic review," *PLoS ONE*, vol. 11, no. 10, Oct. 2016, Art. no. e0163477.
- [7] M. J. M. Chowdhury, M. D. S. Ferdous, K. Biswas, N. Chowdhury, A. S. M. Kayes, M. Alazab, and P. Watters, "A comparative analysis of distributed ledger technology platforms," *IEEE Access*, vol. 7, pp. 167930–167943, 2019.
- [8] M. Kuzlu, M. Pipattanasomporn, L. Gurses, and S. Rahman, "Performance analysis of a hyperledger fabric blockchain framework: Throughput, latency and scalability," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, Jul. 2019, pp. 536–540.
- [9] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong, "Performance analysis of private blockchain platforms in varying workloads," in *Proc. 26th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2017, pp. 1–6.
- [10] N. Sealey, A. Aijaz, and B. Holden, "IOTA tangle 2.0: Toward a scalable, decentralized, smart, and autonomous IoT ecosystem," in *Proc. Int. Conf. Smart Appl., Commun. Netw. (SmartNets)*, Nov. 2022, pp. 01–08.
- [11] A. Rawat, V. Daza, and M. Signorini, "Offline scaling of IoT devices in IOTA blockchain," *Sensors*, vol. 22, no. 4, p. 1411, Feb. 2022.
- [12] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, H. Wang, A. S. M. Kayes, M. Alazab, and P. Watters, "Blockchain challenges and opportunities: A survey," *Int. J. Web Grid Services*, vol. 14, p. 352, Jan. 2018.
- [13] V. Buterin. (2014). *A Next-Generation Smart Contract and Decentralized Application Platform*. [Online]. Available: <https://ethereum.org/en/whitepaper/>
- [14] O. Novo, "Blockchain meets IoT: An architecture for scalable access management in IoT," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1184–1195, Apr. 2018.
- [15] S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data*, Oct. 2017, pp. 557–564.
- [16] A. Antonopoulos and G. Wood, *Mastering Ethereum: Building Smart Contracts and DApps*. Sebastopol, CA, USA: O'Reilly Media, 2019.
- [17] (2020). *Hyperledger—Open Source Blockchain Technologies*. Accessed: Aug. 13, 2022. [Online]. Available: <https://www.hyperledger.org/>
- [18] Hyperledger Fabric. *Hyperledger—Open Source Blockchain Technologies*. Accessed: Aug. 13, 2022. [Online]. Available: [https://www.hyperledger.org/wp-content/uploads/2020/03/hyperledger\\_fabric\\_whitepaper.pdf](https://www.hyperledger.org/wp-content/uploads/2020/03/hyperledger_fabric_whitepaper.pdf)
- [19] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, and D. Enyeart, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, 2018, pp. 1–15.
- [20] C. Cachin. (2016). *Architecture of the Hyperledger Blockchain Fabric*. Accessed: Aug. 13, 2022. [Online]. Available: [https://pdfs.semanticscholar.org/f852/c5f3fe649f8a17ded391df0796677a59927f.pdf?\\_ga=2.153008701.585498121.1597310266-259713143.1596530036](https://pdfs.semanticscholar.org/f852/c5f3fe649f8a17ded391df0796677a59927f.pdf?_ga=2.153008701.585498121.1597310266-259713143.1596530036)
- [21] (2018). *Hyperledger Blockchain Performance Metrics*. Accessed: Oct. 17, 2022. [Online]. Available: [https://www.hyperledger.org/wp-content/uploads/2018/10/HL\\_Whitepaper\\_Metrics\\_PDF\\_V1.01.pdf](https://www.hyperledger.org/wp-content/uploads/2018/10/HL_Whitepaper_Metrics_PDF_V1.01.pdf)
- [22] H. Caliper. *Architecture*. Accessed: Oct. 27, 2022. [Online]. Available: <https://hyperledger.github.io/caliper/v0.4.0/architecture/>
- [23] M. Hammoudeh and R. Newman, "Information extraction from sensor networks using the watershed transform algorithm," *Inf. Fusion*, vol. 22, pp. 39–49, Mar. 2015.
- [24] M. Hammoudeh, G. Epiphaniou, S. Belguith, D. Unal, B. Adebisi, T. Baker, A. S. M. Kayes, and P. Watters, "A service-oriented approach for sensing in the Internet of Things: Intelligent transportation systems and privacy use cases," *IEEE Sensors J.*, vol. 21, no. 14, pp. 15753–15761, Jul. 2021.
- [25] H. Caliper. *Installing and Running Caliper*. Accessed: Oct. 7, 2022. [Online]. Available: <https://hyperledger.github.io/caliper/vNext/installing-caliper/>



**YUSUF UCBAS** received the B.Sc. degree in electrical and electronic engineering from Eskisehir Osmangazi University, in 2017, and the M.Sc. degree in cyber security from Manchester Metropolitan University, in 2020. He obtained his Microsoft Certified Professional Qualification, in 2018. He held several positions in the IT industry. He is currently a Cyber Security Engineer at a large corporation.



**MOHAMMAD HAMMOUDEH** (Senior Member, IEEE) is currently a Saudi Aramco Cybersecurity Chair Professor with the Information and Computer Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia. His research interests include zero-trust security, blockchains, and the IoT. He is also the Founder and the Co-Editor-in-Chief of *Distributed Ledger Technology: Research and Practice* (ACM).



**AMNA ELEYAN** (Member, IEEE) received the Ph.D. degree in software engineering from The University of Manchester. She is currently a Lecturer with the Department of Computing and Mathematics, Manchester Metropolitan University. Her research interests include computer networks and security, web services, the IoT, and machine/deep learning. She is a fellow of the Higher Education Academy.



**MANAR ALOHALY** received the Ph.D. degree from the University of North Texas, USA, in 2020. She is currently an Assistant Professor of cybersecurity and the Director of the Innovation Center, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University. Her research interests include access control, usable privacy and security, cyber deception, insider threat detection, and applied machine learning.

...