Hindawi
Security and Communication Networks
Volume 2018, Article ID 3976093, 14 pages
https://doi.org/10.1155/2018/3976093

Research Article

Performance Analysis of Hyperledger Fabric Platforms

Qassim Nasir,[1] **Ilham A. Qasse**,[2] **Manar Abu Talib**,[2] **and Ali Bou Nassif**[1]

[1]*Electrical and Computer Engineering Department, University of Sharjah, Sharjah, UAE*
[2]*Computer Science Department, University of Sharjah, Sharjah, UAE*

Correspondence should be addressed to Qassim Nasir; qassim.nasir@gmail.com

Received 7 June 2018; Accepted 19 August 2018; Published 9 September 2018

Academic Editor: David Megias

Blockchain is a key technology that has the potential to decentralize the way we store, share, and manage information and data. One of the more recent blockchain platforms that has emerged is Hyperledger Fabric, an open source, permissioned blockchain that was introduced by IBM, first as Hyperledger Fabric v0.6, and then more recently, in 2017, IBM released Hyperledger Fabric v1.0. Although there are many blockchain platforms, there is no clear methodology for evaluating and assessing the different blockchain platforms in terms of their various aspects, such as performance, security, and scalability. In addition, the new version of Hyperledger Fabric was never evaluated against any other blockchain platform. In this paper, we will first conduct a performance analysis of the two versions of Hyperledger Fabric, v0.6 and v1.0. The performance evaluation of the two platforms will be assessed in terms of execution time, latency, and throughput, by varying the workload in each platform up to 10,000 transactions. Second, we will analyze the scalability of the two platforms by varying the number of nodes up to 20 nodes in each platform. Overall, the performance analysis results across all evaluation metrics, scalability, throughput, execution time, and latency, demonstrate that Hyperledger Fabric v1.0 consistently outperforms Hyperledger Fabric v0.6. However, Hyperledger Fabric v1.0 platform performance did not reach the performance level in current traditional database systems under high workload scenarios.

1. Introduction

Blockchain is a new technology that changes the way we store and record data and transactions. It is similar to a traditional database, but the idea behind a blockchain is that we can get rid of the middleman [1].

Blockchain was first introduced in 2008 when Bitcoin, a digital currency, was proposed by Nakamoto [2]. However, this technology is now seen as the backbone of most cryptocurrencies that are currently in circulation. The main feature of this technology is a publicly distributed ledger, where no one owns the ledger and every node in the network has an identical copy of the ledger. The applications of blockchain are not limited to financial applications but can also be used in nonfinancial applications, such as public services [1], reputation systems [3], security, and Internet of Things (IoT) [4, 5]. Like any other rising technology, blockchain is facing some technical challenges such as scalability, privacy, and performance [6]. One of the main challenges in adopting blockchain implementations as an alternative to the traditional databases is performance. Swan [7] introduced seven future technical challenges for adapting blockchain, which are throughput, latency, size and bandwidth, security, wasted resources, usability, and versioning and hard forks. From these challenges, a comprehensive study on research topics on blockchain in [8] has shown that the main limitations and challenges that have not been extensively evaluated are latency and throughput. Also, scalability evaluation is required, since implemented blockchain frameworks are expected to involve large scale of nodes [8]. The study also showed that one of the research gaps in the blockchain field is that most of the current research is addressing the Bitcoin platform, rather than any other blockchain platforms.

One of the recently established blockchain platforms is Hyperledger Fabric [9], an open source, permissioned blockchain that was introduced by IBM, first as Hyperledger Fabric v0.6 [10], and then more recently, in 2017, IBM released Hyperledger Fabric v1.0 [11].

Although there are many blockchain platforms, there is no clear methodology to evaluate and assess the different blockchain platforms in various aspects, such as performance, security, and scalability.

The importance of conducting performance analysis for blockchain platforms must not be overlooked as the results of these studies can be a great input for all blockchain users when they try to choose the best platform for their work especially critical application which include scalable networks.

This paper is one response to this need, where our objective is to conduct performance analysis on two versions of Hyperledger Fabric v0.6 [10] and v1.0 [11]. The main contributions of the paper are the following:

(1) The new version of Hyperledger Fabric was never evaluated against any other blockchain platform

(2) Found deficiencies in both versions, Fabric v0.6 and Fabric 1.0

(3) New measurement tools added to the open source of the performance evaluation tool

(4) Effect of system configurations (e.g., number of transactions, type of nodes) on the performance had been studied first, especially scalability

(5) The analysis results presented in this paper will help the business industry to choose the best blockchain platform for their work

This paper is organized as follows: Section 2 provides an overview of the blockchain technology, and the target platforms in this analysis. In Section 3, related work in performance and scalability evaluation of the blockchain platform is listed. In Section 4, the methodology for evaluating blockchain implementations is presented. Then, a discussion of the results and their implications are covered in Section 5. Finally, Section 6 concludes this paper.

2. Background

2.1. Blockchain Technology. Blockchain is a peer to peer distributed ledger of transactions, stored and saved in a chain of connected blocks [12]. It can be identified as the technology that permits records to be shared by all the nodes in the network while being maintained and owned by no one [7]. Nodes or the peers in the network are clients' computers or mobile devices. This technology has been considered as one of the major revolutionary paradigms since the Internet [6, 7, 12].

A blockchain transaction is a sequence of requests applied on some states and values saved on the blockchain. The nodes in the network need to verify the transaction and add the block to the blockchain. This process is called consensus mechanism. There are many approaches to add the blocks to the network, but the most commonly used in the blockchain are proof of work [13] and Byzantine Fault Tolerance [14].

Blockchain can be classified mainly into three types: public, private, and permissioned blockchain. The classification is based on the ability of the node to access or add new block in the network [15]. In public blockchain, any node can join the network, participate in the transactions, and create the blocks [16]. Bitcoin [1], Ethereum [17], Dash [18], and Litecoin [19] are examples of public blockchain. On the other hand, both the read and write permissions and participation in the network are restricted in private blockchain [16]. The following platforms are examples of private blockchain: Hydrachain [20], Ripple [21], Monax [22], and Multichain [23].

The third blockchain type, which is a middle solution between private and public blockchain, is known as consortium or permissioned blockchain [24, 25]. In this type of blockchain the network is controlled by a group of nodes. Available permissioned blockchain platforms are Hyperledger Fabric [9] and Corda [26].

2.2. Hyperledger Project. Hyperledger project is an open source, permissioned, distributed ledger founded by Linux Foundation [9]. This project is divided into five subprojects: Fabric [10], Sawtooth [27], Indy [28], Burrow [29], and Iroha [30]. In this paper, we will only focus on the Hyperledger Fabric project.

Hyperledger Fabric is an enterprise-grade open source platform that is maintained by IBM and Linux Foundation. Unlike Bitcoin and Ethereum, Hyperledger Fabric does not have any cryptocurrency, where the access to the network is restricted to the network members only, and not anyone can join the network. The mechanism used to validate the transactions and create blocks in Hyperledger Fabric is PBFT [31]. The transactions are controlled in Hyperledger Fabric using chaincode (smart contract), which is a program code that provides the ability to write and design the applications to interact with the network. The privacy of the transactions between the participant in the network can be obtained using an isolation mechanism known as channel. The channel ensures that the transaction and data are available only to the nodes that are members in the channel.

According to the official documentation of Hyperledger Fabric [9], a transaction is an invoke or an instantiate request that is submitted by the peer for ordering and validation. The instantiate request initializes a chaincode in a particular channel, while the invoke transactions execute read/write operation on the ledger.

The main components of the Hyperledger Fabric architecture are peer nodes, ordering nodes, and client applications [32].

The identities of the components are generated from the certificate authorities. The ordering nodes collect and order transactions from different applications in a block.

In this paper, we will conduct performance analysis on two versions of Hyperledger Fabric: version 0.6 and 1.0. There are few differences in the architecture of the two versions v0.6 and v1.0. Most of differences between the two architectures are in relation to how more roles are added and defined in the new version. For the peers (nodes) in v1.0, there are endorsing peers, committing peers, and ordering service, whereas in v0.6, there are only two types of peers: a validating peer and a nonvalidating peer. Several major concerns with the architecture of v0.6 were recognized and addressed in v1.0, such as performance and scalability [33].

Table 1 highlights the main differences between Fabric v0.6 and Fabric v1.0.

Table 1: Main differences between Fabric v0.6 and Fabric v1.0.

Feature	Fabric 0.6	Fabric 1.0
Ordering service	×	√
Channel Mechanism	×	√
Endorsement Policy	×	√
Types of Peers	1	2 (endorser and committer)
API	REST API	gRPC
World State	levelDB	levelDB, CouchDB

3. Related Work

In this section, the research and experiments done to enhance and evaluate the performance of blockchain platforms and their protocols are addressed in this section.

Bartoletti et al. [34] proposed a framework that supports data analytics only on Bitcoin and Ethereum platforms. The proposed tool permits relative blockchain data to be integrated with data from external sources. The framework also allows it to organize the data in a database.

Xu et al. [35] classified the blockchain implementations and compared them with the blockchain-based frameworks to study the impact of the blockchain architecture on software architectures. The introduced classification highlights major blockchain platforms architectural characteristics and the impact of the blockchain design on the quality of the blockchain-based software, such as performance and scalability.

A recent and related paper [36] identified the quality attributes for the blockchain technology and investigated the quality issues, solutions, and requirements for blockchain implementation. The results show that the blockchain platforms need to be improved in many aspects, such us security, scalability, privacy, and performance, in terms of latency, cost-effectiveness, etc.

Yasaweerasinghelage et al. [37] proposed the use of simulation framework and performance modelling to predict the latency of blockchain-based systems. Most of the predicted results have a relative error of less than 10%. This approach also aims to help in evaluating different blockchain design options.

Kocsis et al. [38] proposed a performance evaluation model for blockchain technology, which was used to evaluate Hyperledger Fabric v0.6. The main purpose of the model is to evaluate the software design in its early stages, where changing the requirement of the software will not have a big impact on the overall cost and time.

Croman et al. [39] studied the challenges in blockchains scalability, specially Bitcoin. The results showed that, to get significant throughput and latency improvements, reparametrization of the Bitcoin's interval and the block size is suggested.

Jermy Rubin [40] introduced an open source software "BTCSpark" for analyzing Bitcoin and building blockchain analysis tools. The tool provides an environment that is easy to use with good performance.

A recent study [41] designed a performance model for the Practical Byzantine Fault Tolerance (PBFT) consensus mechanism which is used in Hyperledger Fabric. The study studied the possibilities for performance bottlenecks in networks that have large numbers of nodes.

Kokoris-Kogias et al. [42] introduced a scalable consensus protocol for public blockchain platforms called ByzCoin. The proposed protocol provides better security and performance when it was tested on Bitcoin platform.

Behl et al. [43] presented a parallelization approach to scale BFT systems and increase their performance. The evaluation results showed that using this approach will increase the throughput of the system compared to the throughput achieved with the traditional approach.

Eyal et al. [44] proposed Bitcoin-NG protocol to address the scalability of Bitcoin. The paper also addressed the security and efficiency of similar protocols. The evaluation results of the protocol demonstrated that Bitcoin-NG with limited bandwidth provides optimal scalability.

Vuckolic et al. [45] compared between proof of work based blockchains and those based byzantine fault tolerance in terms of scalability and performance. The paper showed that the performance of bft based blockchains (such as Hyperledger) is better compared to those that are based on PoW (such as Bitcoin). On the other hand PoW based blockchains provide better scalability than bft based blockchains.

Aniello et al. [46] presented implementation and evaluation for a two-layered blockchain platform for a federated database. The proposed architecture provides high performance and data integrity but is weak in terms of scalability and data availability.

Aniello et al. also investigated different consensus mechanisms to overcome the current issues in architecture, such as Byzantine Fault Tolerant (BFT) and Distributed Hash Table (DHT).

Suankaewmanee et al. [47] proposed Mobichain which is a mobile commerce application that uses blockchain as a core technology. The aim of this application is to make the transaction in m-commerce more secure. The paper also conducted performance evaluation for the Mobichain application, which showed that the proposed module is efficient solution for m-commerce applications.

A recent study [48] measured the performance of two permissioned blockchain implementations: Ethereum and Hyperledger Fabric, by varying the number of transactions (from 1 to 10000 transactions). This methodology was used on

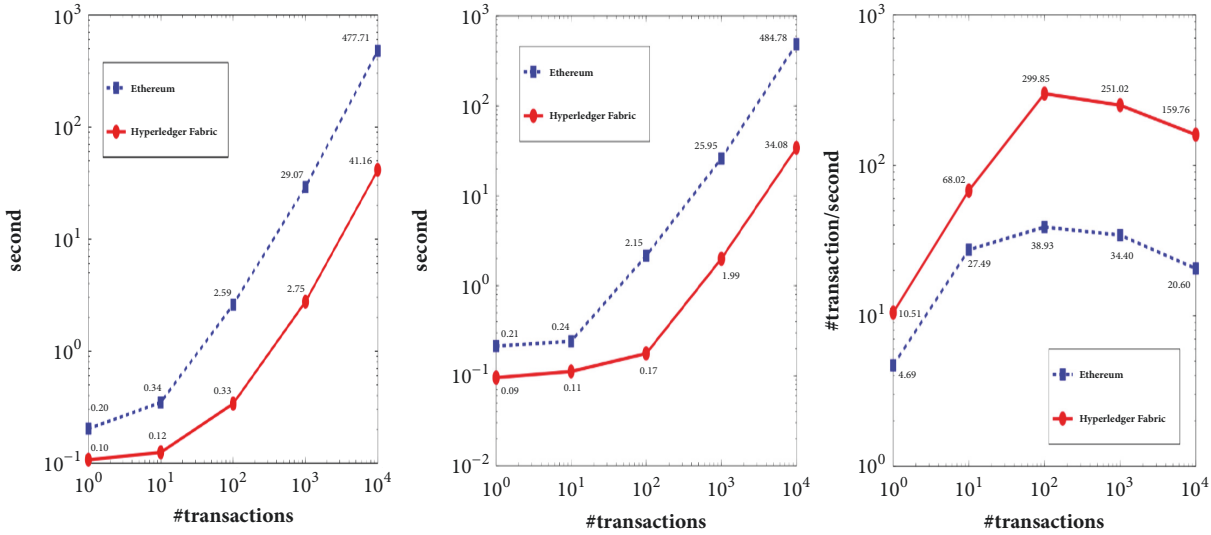

Figure 1: Execution time, latency, and throughput of Ethereum and Hyperledger Fabric, respectively [48].

simple cash transfer applications, with three major functions: Create Account, Issue Money, and Transfer Money. The Create Account function is used to create new users, while the other two functions issue and transfer money and are used, respectively, to issue money into an account and transfer money from one user to another. The results for this experiment showed that the Hyperledger outperforms Ethereum in all specified evaluation metrics, which are execution time, throughput, and latency, as shown in Figure 1.

Another recent experiment studied the scalability of three blockchain platforms, Hyperledger, Ethereum, and Parity [49, 50]. The scalability is analyzed by setting the transaction rate at constant and changing both the number of users and the number of servers. The experiment showed that the throughput and latency of the Ethereum platform reduced almost linearly beyond 8 servers. On the other hand, the Hyperledger Fabric platform stops responding beyond 16 nodes due to the overhead of communication between nodes in the consensus protocol (see Figure 2).

Table 2 summaries the related work for evaluating the performance of blockchain platforms.

4. Methodology

Two versions of the Hyperledger blockchain platform are evaluated in this experiment, Hyperledger Fabric v0.6 and v1.0. The experiments are conducted on an HPC server with the Intel(R) Xeon(R) CPU E5-2690, 2.60 GHz, 24 core CPU, 64 GB RAM, and running Ubuntu 16.04. For each version, two experiments are conducted. The first experiment is to conduct a performance analysis of the two versions of Hyperledger Fabric platform, v0.6 and v1.0. The performance evaluation of the two platforms will be assessed in terms of execution time, latency, and throughput, by varying the workload (number of transactions and requests (query or invoke) that was requested simultaneously by the peers) in each platform up to 10,000 transactions. For the two versions, the transaction is measured from the submission of the transaction for consensus by the peers, to adding the transaction to a block. Execution time is the time required for a platform to add and execute a transaction successfully. Throughput can be defined as "the number of successful transactions per second" [40]. Finally, latency in blockchain can be measured as the time that a specific platform will take to respond to each transaction.

In the second experiment, the scalability of the two platforms will be analyzed by varying the number of nodes up to 20 nodes in each platform, while still measuring the same metrics execution time, throughput, and latency for the number of peers in the network.

4.1. Blockchain Evaluation Framework. To analyze the performance for the Hyperledger, we used a modified version of Hyperledger caliper [51], which is a performance benchmark tool for multiple blockchain platforms that relies on a running blockchain network as the target platform.

This tool will generate html reports that contain some of the performance characteristics, such as resource usage and transactions per second (TPS). We have modified Hyperledger caliper to calculate the execution time of the transactions. Also in addition to the execution time we have designed Fabric v0.6 that is compatible with Hyperledger caliper.

The architecture of evaluating the performance of the blockchain platform consists of four main layers: the performance analysis layer, the adapter layer, the interface layer, and the blockchain framework, as shown in Figure 3.

The adaptation layer is the main component of the caliper architecture. The key function of this layer is to integrate different blockchain implementations into the evaluation system. For every blockchain platform to be tested, the adaptor is responsible to translate between the blockchain protocol and the caliper north bound interfaces (NBIs).

The interface layer is responsible for providing multiple blockchain north bound interfaces that are used to deploy,

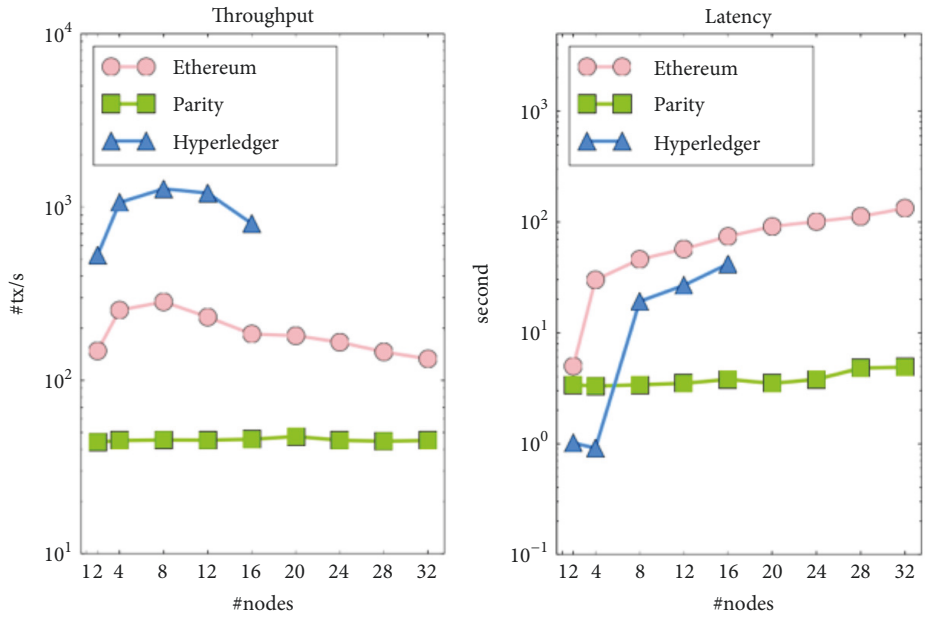

Figure 2: Scalability for Hyperledger Fabric, Ethereum, and Parity [49, 50].

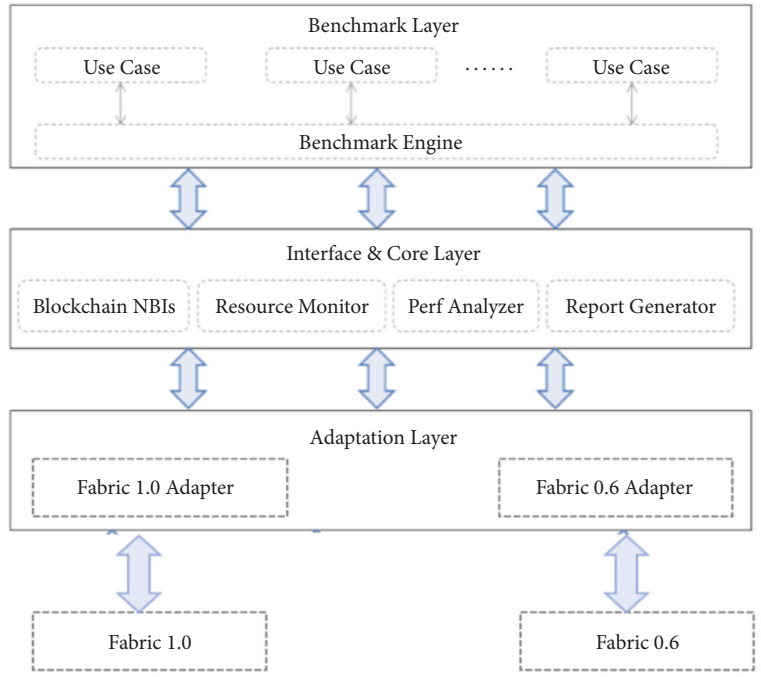

Figure 3: Evaluation framework architecture.

invoke, and query smart contracts. This layer also implements functions to monitor resources such as memory and CPU.

The function of the performance evaluation layer is to perform stress tests on the implemented blockchain platform, where the blockchain network details and test parameters are provided as input for each test as shown in Figure 4.

4.2. Smart Contracts. To analyze the blockchain platform's performance, we have deployed a simple money transfer application (chaincode). The main functions in the chaincode are transfer money (invoke) and query function.

The transfer money function is used to transfer cash from one account to another, while the query function is

Table 2: Blockchain performance evaluation overview.

Author	Year	Summary
Bartoletti et al.	2017	A general framework for blockchain analytics
Xu et al.	2017	Classified the blockchain implementations and compared them with the blockchain-based frameworks to study the impact of the blockchain architecture on software architectures
Koteska et al.	2017	Identified the quality attributes for the blockchain technology, and investigated the quality issues, solutions and requirements for the blockchain implementation
Yasaweerasinghelage et al.	2017	Proposed the use of simulation framework and performance modelling to predict the latency of blockchain-based systems
Kocsis et al.	-	Proposed performance evaluation model for blockchain technology and it was used to evaluate Hyperledger fabric v0.6
Croman et al.	2016	Studied the challenges in blockchains scalability, specially Bitcoin
Jermy Rubin	2015	Introduced an open source software "BTCSpark" for analyzing Bitcoin and building blockchain analysis tools
Sukhwani et al.	2017	Performance Modeling of PBFT Consensus Process for Permissioned Blockchain Network
Kokoris-Kogias et al.	2016	Introduced a scalable consensus protocol for public blockchain platforms called ByzCoin
Behl et al.	2014	Presented a parallelization approach to scale BFT systems and increase their performance.
Eyal et al.	2015	Proposed Bitcoin-NG protocol to address the scalability of Bitcoin.
Vuckolic et al.	2016	Comparison between proof of work based blockchains and those based byzantine fault tolerance in term of scalability and performance.
Aniello et al.	2017	Implementation and evaluation for a two-layered blockchain platform for a federated database.
Suankaewmanee et al.	2017	Proposed Mobichcain which is a mobile commerce application that uses blockchain as a core technology.
Suporn et al.	2017	Performance evaluation for two private blockchain implementations; Ethereum and Hyperledger fabric, by varying the number of transactions
Dinh et al. Anh et al.	2017	Performance and scalability evaluation of three blockchain platforms: Hyperledger, Ethereum and parity

used to query the available amount in the specified account. Deploying the chaincode in the blockchain platforms is done before evaluating the performance of the platforms.

4.3. Evaluation Flow. The performance analysis test flow depends on the blockchain platform that will be evaluated, where different blockchain implementations will have different test workflows. Figures 5 and 6 show the test workflow for Fabric v0.6 and Fabric v1.0, respectively.

5. Results and Discussions

In this section, the performance of Hyperledger Fabric versions will be analyzed based on the execution time, average latency, throughput, and scalability. This experiment can be divided into two main parts, assessing the performance of a single peer network and testing the scalability of the implemented blockchain network. Based on the conducted experiment, Fabric v1.0 provides better performance results and outperforms Fabric v0.6 across all test cases.

5.1. Performance Evaluation for Single Peer

Evaluating Execution Time. We evaluated the execution time of the two platforms by changing the number of the transactions and analyzing the execution time for the different functions. In general, the execution times increase as the number of transactions grows. The execution times for Fabric v1.0 are better than the execution times for Fabric v0.6 in all datasets for the query function, as shown in Figure 7. The gap between the execution times of both platforms in query function increases as the datasets grow larger. On the other hand, for the invoke function, Fabric v0.6 outperforms Fabric v1.0 execution times when the dataset is small (10–100 transactions), but as the number of transactions increases, the execution times for Fabric v1.0 are lower than the execution times of Fabric v0.6. The difference between the execution times of the two platforms in invoke function is not large compared to difference in the query function. Figure 8 demonstrates the execution times of invoke function for both implementations.

Evaluating Average Latency. Figures 9 and 10 demonstrate the average latency of each implementation for executing the invoke and query functions, respectively, using different datasets. In the query function the average latency for Fabric v0.6 is almost 4x that of the average latency in Fabric v1.0 for all datasets, except when the dataset is 100 transactions, where the difference of the average latency between the two platforms is less. For the invoke function, it can be noted that as the number of transactions increases the average latency for the two platforms grows larger, and the difference in the average latency between the versions will be more noticeable, especially when the number of transactions increases from 1000 to 10000, as shown in Figure 10.

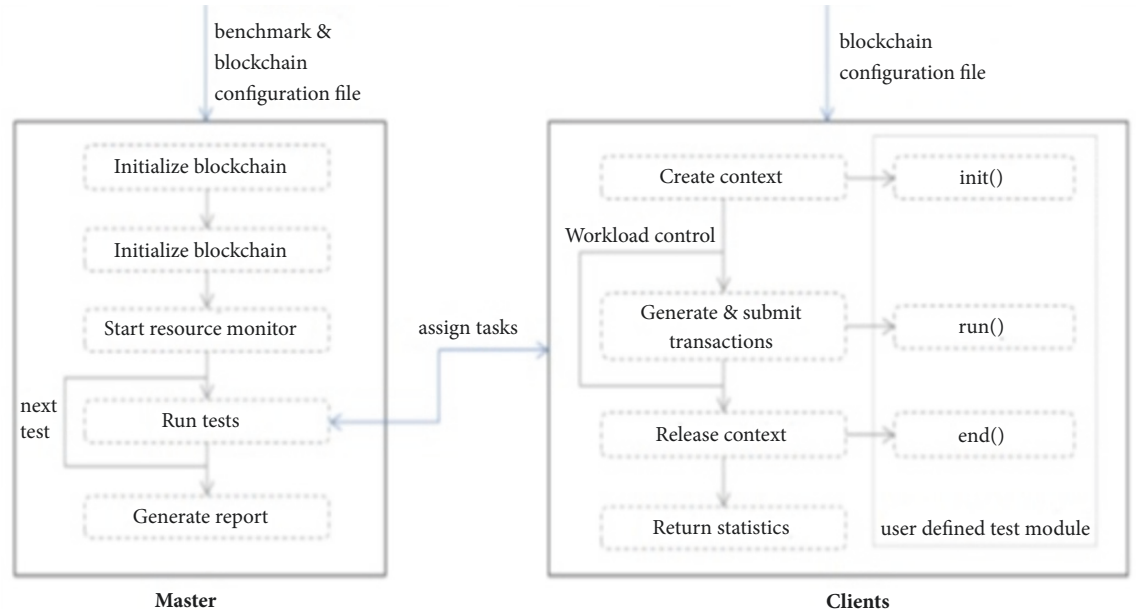

Figure 4: Performance evaluation layer [51].

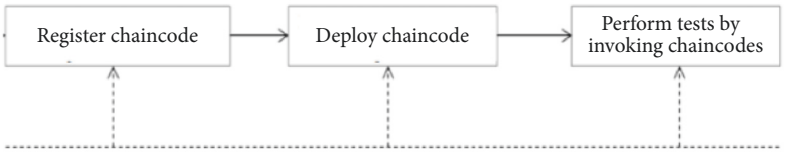

Figure 5: Test flow for Fabric v0.6.

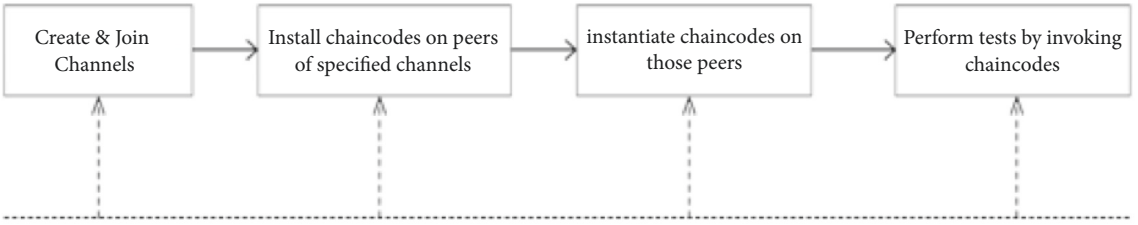

Figure 6: Test flow for Fabric v1.0 [51].

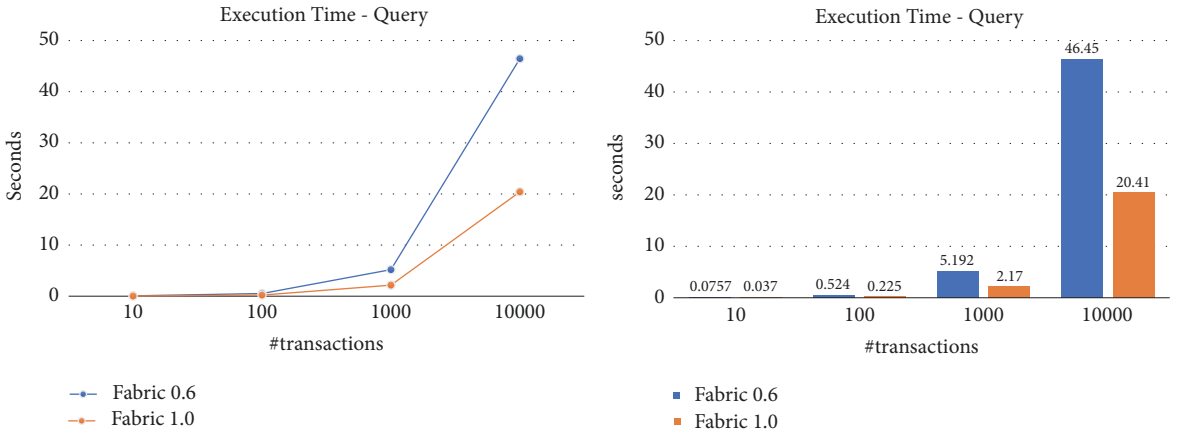

Figure 7: Execution time for query function.

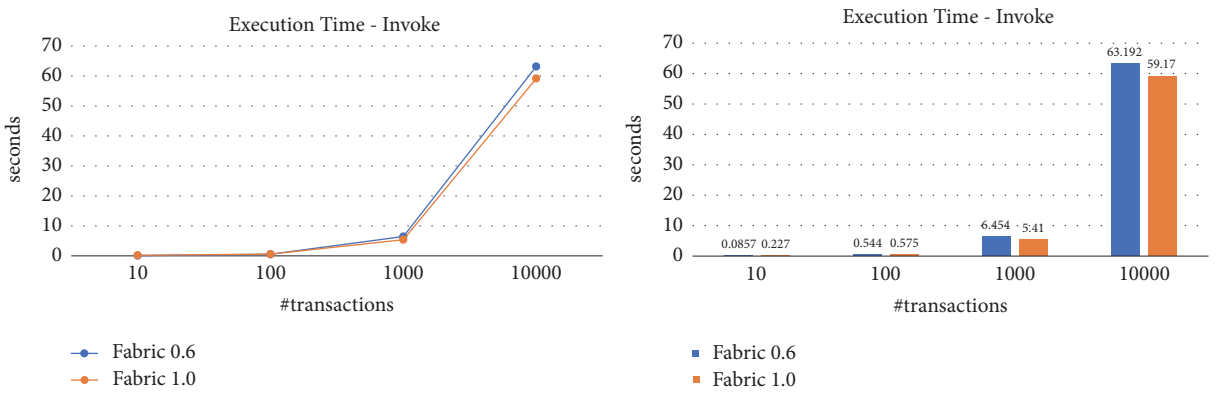

Figure 8: Execution time for invoke function.

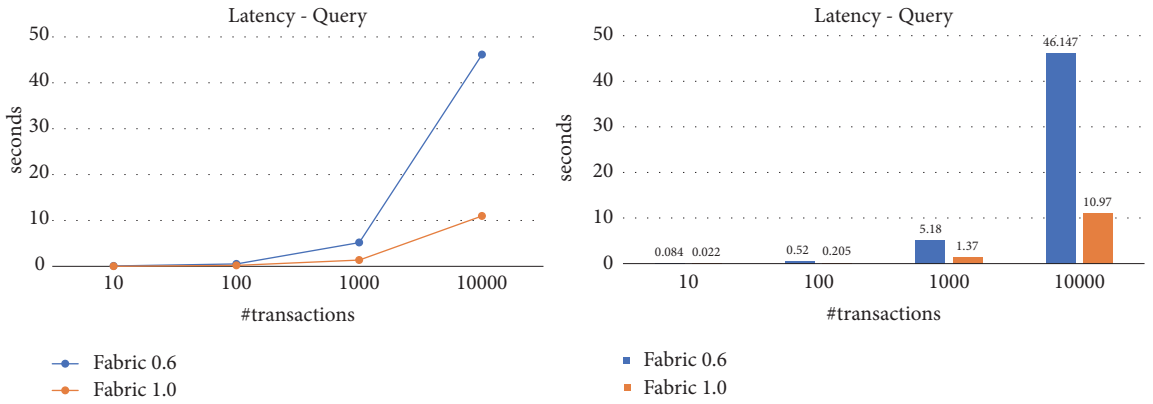

Figure 9: Average latency for query function.

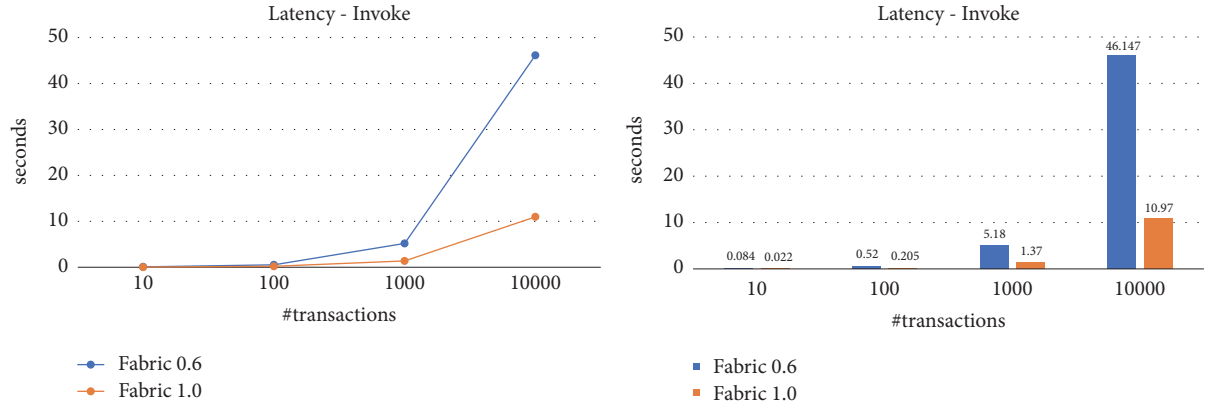

Figure 10: Average latency for invoke function.

Evaluating Throughput. Figure 11 shows the throughput of each of the implementations for executing the query function using different datasets. Fabric v1.0 has higher throughput than Fabric v0.6 in all number of transactions. We observed that the gap of the average throughputs between the two implementations increases as the number of transactions increases, when the number of transactions in the datasets is 100.

Similar to Figure 11, Figure 12 shows the average throughput of the two versions when executing the invoke function using different datasets. When the number of transactions in the dataset is small (10 – 100), Fabric v0.6 has higher throughput than Fabric v1.0. As the number of transactions grows, the throughput of Fabric v0.6 decreases, and the average throughput for Fabric v1.0 will be higher.

5.2. Scalability Evaluation. In this section, the scalability of the two platforms will be analyzed by varying the number of nodes up to 20 nodes in each platform, while still measuring the same metrics execution time, throughput, and latency in

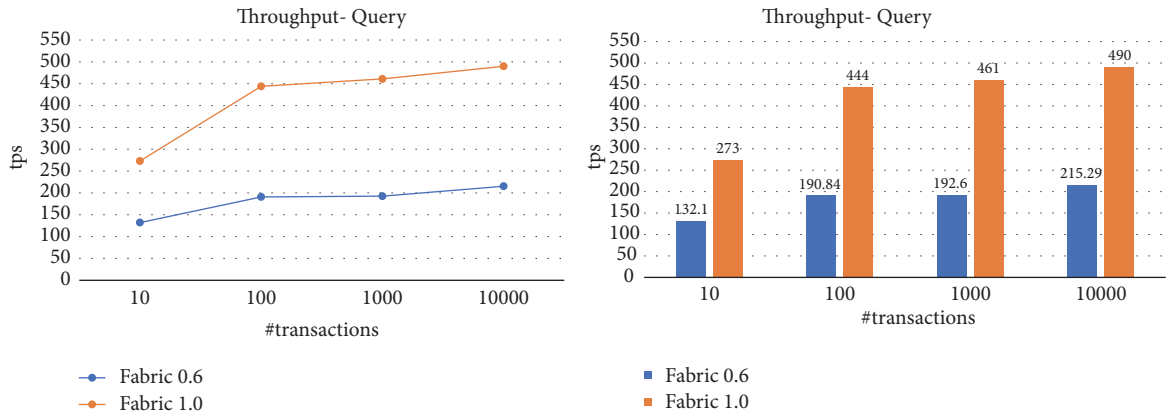

Figure 11: Throughput for query function.

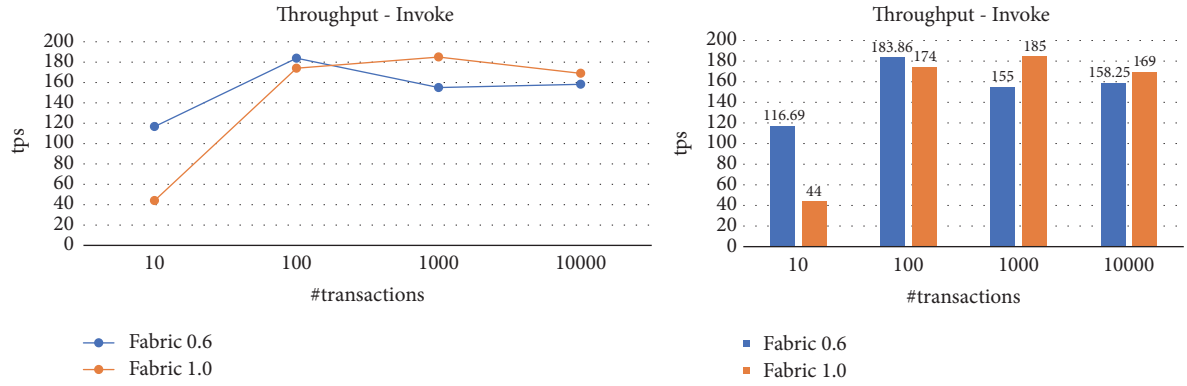

Figure 12: Throughput for invoke function.

datasets of 1000-10000 transactions. The results showed that the maximum number of nodes that Fabric v0.6 can have is 16.

The results also showed that Fabric v1.0 cannot handle 10000 concurrent transaction when the number of nodes in the network exceeds 6. In general, Fabric v1.0 maintains the same performance range in all evaluation metrics, regardless the number of peers (nodes) in the network. On the other hand, Fabric v0.6 performance is affected if the number of peers in the network increases.

Evaluating Average Latency. Figure 13 demonstrates the average latency of each implementation for executing the invoke function when the number of transactions in the dataset is 1000. Fabric v1.0 has lower average latency compared to Fabric v0.6, regardless the number of the nodes in the network. Also, we can observe that Fabric v1.0's average latency is within a certain range for all different scenarios, while the average latency for Fabric v0.6 increases as the number of nodes in the network increases. Similar to Figure 13, Figure 14 shows the average latency of each implementation for executing the invoke function when the number of transactions in the dataset is 10000. Fabric v1.0 outperforms Fabric v0.6 when the number of peers in the network is less than 6 nodes. Fabric v1.0 fails to execute 10000 transactions when the number of nodes in the network is greater than 6, while Fabric v0.6 succeeded in executing 10000 transactions, but the average latency increases as the number of nodes in the network grows.

Evaluating Execution Time. Figures 15 and 16 show the execution time of each implementation for the invoke function when the number of transactions in the dataset is 1000 and 10000, respectively. Similar to the average latency results, Fabric v1.0 has a lower execution time compared to Fabric v0.6 regardless of the number of nodes in the network. Also, we can indicate that Fabric v1.0's execution time is within a certain range for all different numbers of nodes in the network, while the execution time for Fabric v0.6 increases as the number of the nodes in the network increases.

When the number of transactions is 10000, Fabric v1.0 outperforms Fabric v0.6 when the number of peers in the network is less than 6 nodes. As shown in Figure 16, Fabric v1.0 fails to execute 10000 transactions when the number of nodes in the network is greater than 6, while Fabric v0.6 succeeded in executing 10000 transactions, but the execution time increases as the number of nodes in the network grows.

Evaluating Throughput. Similar to the average latency and execution time results, in terms of throughput in a scaling environment, Fabric v1.0 outperforms Fabric v0.6. Figures 17 and 18 show the throughput of each implementation for

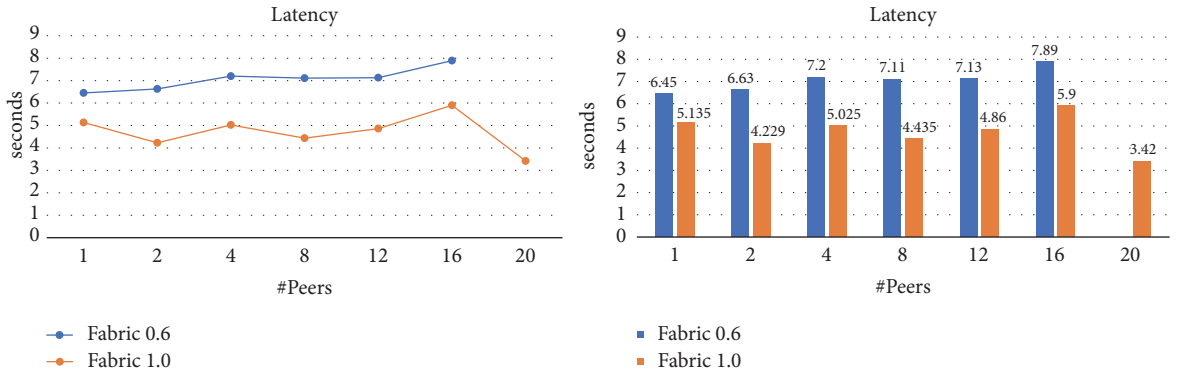

Figure 13: Average latency for dataset of 1000 transactions.

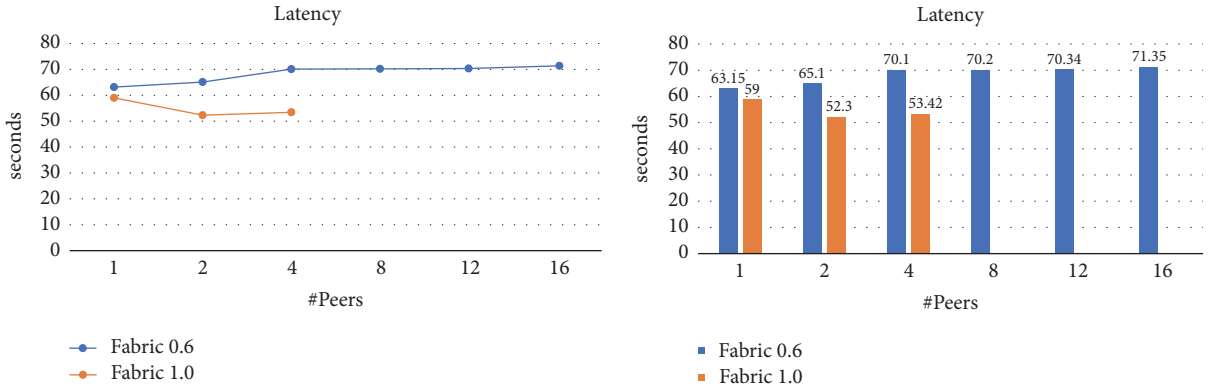

Figure 14: Average latency for dataset of 10000 transactions.

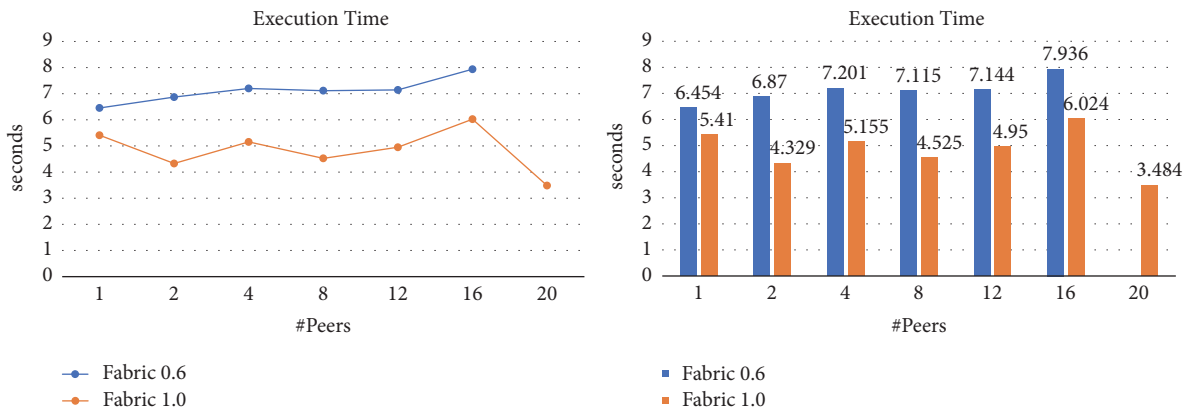

Figure 15: Execution time for dataset of 1000 transactions.

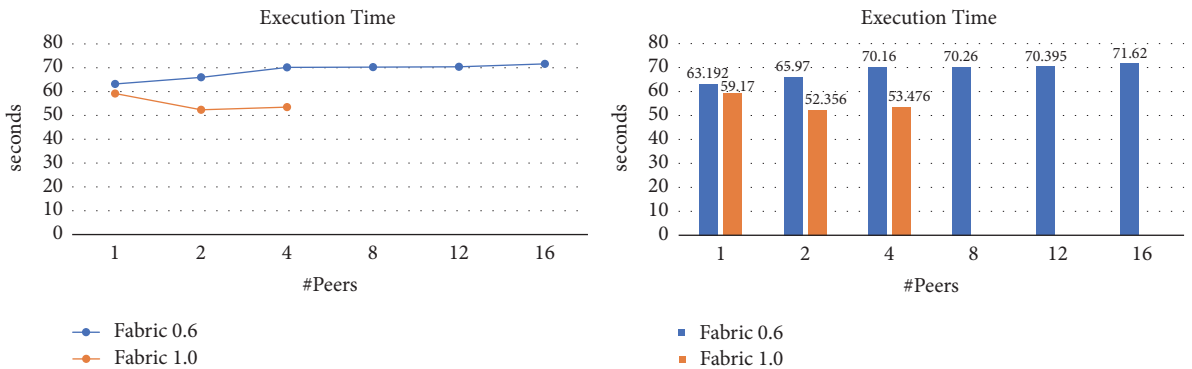

Figure 16: Execution time for dataset of 10000 transactions.

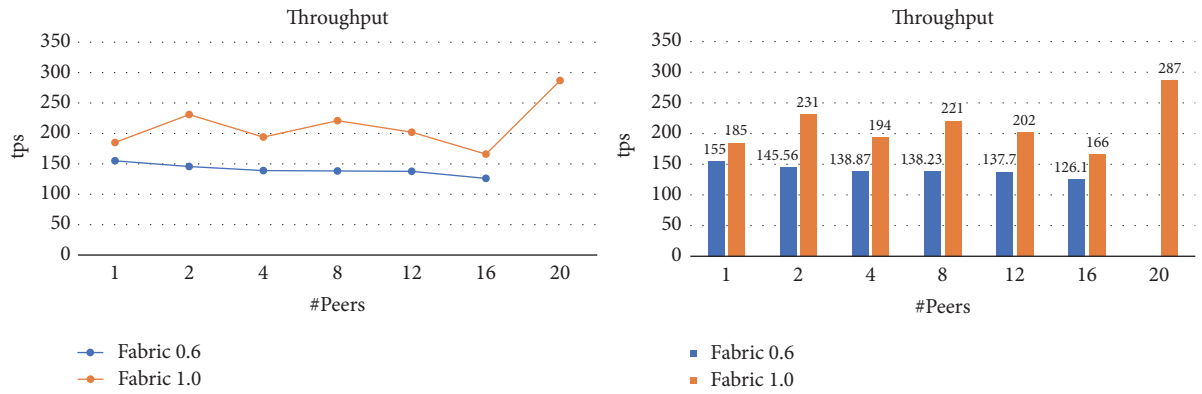

Figure 17: Throughput for dataset of 1000 transactions.

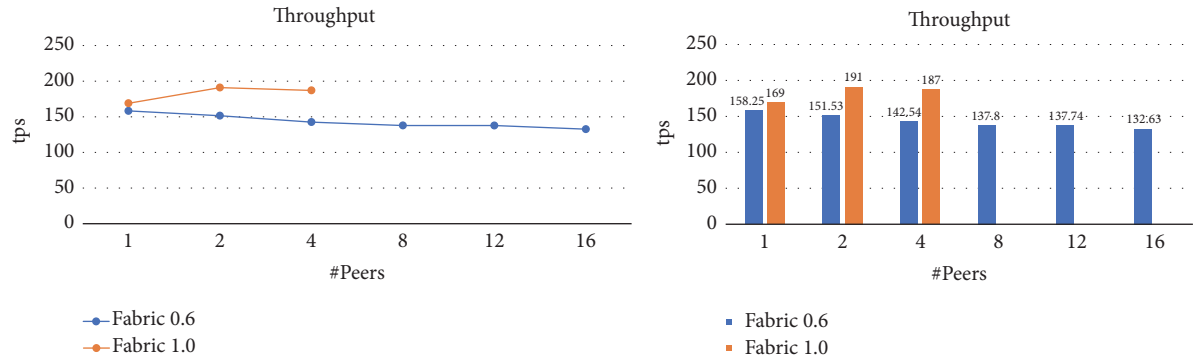

Figure 18: Throughput for dataset of 10000 transactions.

the invoke function when the number of transactions in the dataset is 1000 and 10000, respectively.

5.3. Fabric Multiple Organization. To assess the scalability of Fabric v1.0, we have also evaluated the scenario when there are multiple organizations in the network. The multiple organization concept was introduced in Fabric v1.0 and was not used in Fabric v0.6. In this section, the scalability of Fabric v1.0 will be applied to two organizational cases and will be analyzed by measuring the same performance metrics, execution time, throughput, and latency and by comparing them to the results of Fabric v1.0 with only one organization. Fabric v1.0 with two organizations has better performance in all evaluation metrics as shown in Figure 19.

The results also show that Fabric v1.0 can scale up to 26 nodes (13 peers in every organization).

Table 3 summarizes performance analysis results for the two platforms Fabric v0.6 and Fabric 1.0.

6. Conclusion

This paper presents performance analysis of the two versions of Hyperledger Fabric, Fabric v0.6 and Fabric v1.0, with varying numbers of workload and numbers of nodes in the network. Overall, the performance analysis results across all evaluation metrics, throughput, execution time, latency, and scalability, demonstrate that Hyperledger Fabric v1.0 consistently outperforms Hyperledger Fabric v0.6.

In general, Fabric v1.0 maintains the same performance range in all evaluation metrics, regardless the number of peers (nodes) in the network. On the other hand, the performance of Fabric v0.6 is affected if the number of peers in the network increases. In terms of scalability, the results showed that the maximum number of nodes that Fabric v0.6 can have is 16, while Fabric v1.0 can have up to 26 nodes in the network. The evaluation also showed that Fabric v1.0 cannot handle 10000 concurrent transactions when the number of nodes in the network exceeds 6. Also, in this paper we have evaluated Fabric v1.0 where the assessment demonstrated that Fabric v1.0 with two organizations has better performance in all evaluation metrics compared with Fabric v1.0 with only one organization. For future work, we plan to perform assessments on newer versions of Hyperledger Fabric and explore more test cases such as the impact of having multiple orderers in the network on the overall performance. Additionally, we are interested in comparing the performance of permissioned blockchain and public blockchain platforms with traditional databases.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

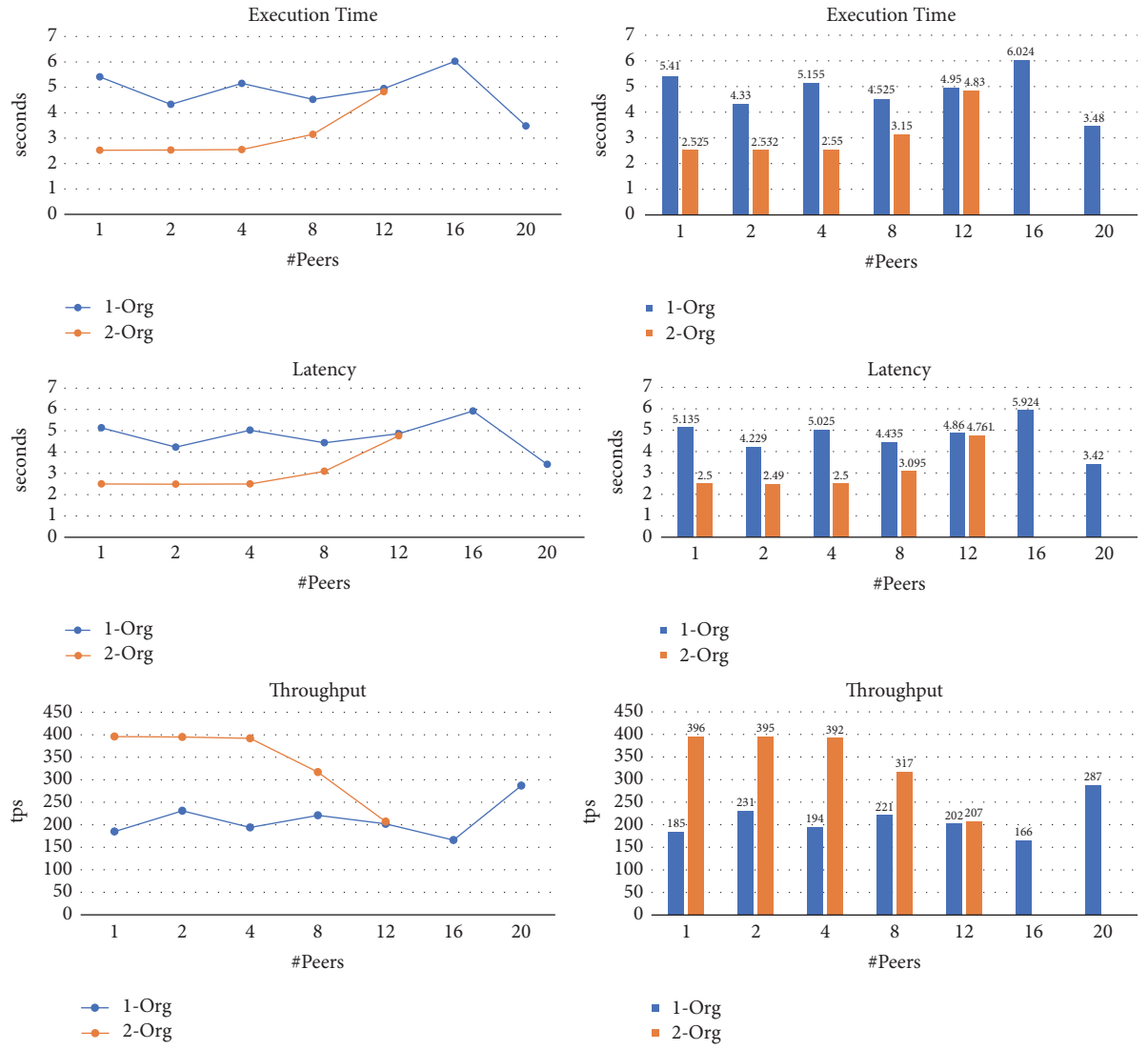

Figure 19: Evaluation for two organizations.

Table 3: Performance analysis results for the two platforms Fabric v0.6 and Fabric 1.0.

Network	Comparison Metrics	Results
Single Peer	Maximum Concurrent Transactions	Both platforms can handle up to 20000 transactions
	Execution Time	Execution times for Fabric v1.0 are lower than the execution times of Fabric v0.6, especially when the number of transactions is high (greater than 100)
	Latency	Average latency for Fabric v1.0 is lower than the average latency of Fabric v0.6, especially when the number of transactions is high (greater than 100)
	Throughput	Fabric v1.0 has higher throughput than Fabric v0.6
Multiple Peers	Maximum Number of Peers	(i) Maximum number of nodes that Fabric v0.6 can have is 16 (ii) Maximum number of nodes that Fabric v1.0 can have is 26
	Maximum Concurrent Transactions	(i) Fabric v0.6 can handle up to 20000, regardless the number of the nodes in the network. (ii) Number of concurrent transactions that fabric v1.0 can handle depends on the number of the nodes in the network. Fabric v1.0 cannot handle 10000 concurrent transaction when the number of nodes in the network exceeds 6.
	Execution Time	Fabric v1.0 has lower execution times compared to Fabric v0.6, regardless the number of the nodes in the network.
	Latency	Fabric v1.0 has lower average latency compared to Fabric v0.6, regardless the number of the nodes in the network.
	Throughput	Fabric v1.0 has higher throughput than Fabric v0.6

Acknowledgments

This research was supported by the University of Sharjah and OpenUAE Research and Development Group. The paper falls under two research projects that are funded by the University of Sharjah: Towards Blockchains Benchmarking & Its Security Metrics for UAE Governments project no. 1702040388 to Qassim Nasir and Open Source Software Use for Law Enforcement in the UAE project no. 1602141139 to Manar Abu Taleb. We are grateful for the fund that we got from the Research and Graduate Studies Office at University of Sharjah (UOS) as well as Dubai Electronic Security Center (DESC), a governmental entity in the UAE. We also thank OpenUAE Research and Development Group for using the lab and the allocated resources at University of Sharjah.

References

[1] M. Nofer, P. Gomber, O. Hinz, and D. Schiereck, "Blockchain," *Business & Information Systems Engineering*, vol. 59, no. 3, pp. 183–187, 2017.

[2] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," http://Www.Bitcoin.Org, p. 9, 2008.

[3] R. Dennis and G. Owen, "Rep on the block: A next generation reputation system based on the blockchain," in *Proceedings of the 10th International Conference for Internet Technology and Secured Transactions (ICITST '15)*, pp. 131–138, December 2015.

[4] N. Kshetri, "Can Blockchain Strengthen the Internet of Things?" *IT Professional*, vol. 19, no. 4, Article ID 8012302, pp. 68–72, 2017.

[5] M. H. Miraz and M. Ali, "Applications of blockchain technology beyond cryptocurrency," *Annals of Emerging Technologies in Computing*, vol. 2, no. 1, pp. 1–6, 2018.

[6] M. Pilkington, "Blockchain technology: principles and applications," *Research Handbook on Digital Transformations*, pp. 1–39, 2015.

[7] M. Swan, "Blockchain: Blueprint for a new economy," 2015.

[8] J. Yli-Huumo, 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, 2016.

[9] "Hyperledger Fabric Project," https://hyperledger-fabric.readthedocs.io/en/release/ [Accessed: 05-Aug-2017].

[10] "Hyperledger Fabric v0.6 Project," https://github.com/hyperledger/fabric/tree/release-1.0 [Accessed: 07-Aug-2017].

[11] "Hyperledger Fabric v1.0 Project," https://github.com/hyperledger/fabric/tree/v0.6 [Accessed: 07-Aug-2017].

[12] M. Crosby, P. Nachiappan, S. Verma, and V. Kalyanaraman, "Blockchain technology - beyond bitcoin," *Berkeley Engineering*, p. 35, 2016.

[13] R. Grinberg, B. Primer, B. Ecosystem, I. B. Sustainable, and L. Issues, "Bitcoin: an innovative alternative digital currency," *Hastings Science & Technology Law Journal*, p. 50, 2012.

[14] C. Cachin and M. Vukolić, "Blockchain Consensus Protocols in the Wild," 2017.

[15] V. Buterin, "On Public and Private Blockchains [Blog post]," Ethereum Blog, 2015, https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/.

[16] V. Buterin, "Visions, Part 1: The Value of Blockchain Technology," Ethereum Blog, 2015, https://blog.ethereum.org/2015/04/13/visions-part-1-the-value-of-blockchain-technology/.

[17] "Ethereum project," https://github.com/ethereum/wiki/wiki/White-Paper [Accessed: 13-Jan-2018].

[18] E. Duffield and D. Diaz, "Dash: A Privacy Centric CryptoCurrency," vol. 15, pp. 1–11, 2014.

[19] M. Haferkorn and J. M. Q. Diaz, "Seasonality and interconnectivity within cryptocurrencies - An analysis on the basis of bitcoin, litecoin and namecoin," *Lecture Notes in Business Information Processing*, vol. 217, pp. 106–120, 2015.

[20] "Hydrachain Project," https://github.com/HydraChain/hydrachain [Accessed: 13-Jan-2018].

[21] "Ripple Project," https://monax.io/platform/db/ [Accessed: 13-Jan-2018].

[22] "Monax Projet," https://monax.io/platform/db/ [Accessed: 13-Jan-2018].

[23] "Multichain Project," http://www.multichain.com/white-paper/ [Accessed: 13-Jan-2018].

[24] A. Ian, "Bank of England: Central banks looking at'hybrid systems' using Bitcoin's blockchain technology," *International Business Time*, 2015.

[25] R. Brown, "Blockchain is where banks have the most obvious opportunity. But you ignore Bitcoin at your peril," Thought on the future of finance Blog, 2015, http://gendal.me/2015/05/12/blockchain-is-where-banks-have-the-most-obviousopportunity-but-you-ignore-bitcoin-at-your-peril/.

[26] "Corda Project," https://docs.corda.net/_static/corda-technical-whitepaper.pdf [Accessed: 13-Jan-2018].

[27] "Hyperledger Sawtooth Project," https://www.hyperledger.org/projects/sawtooth [Accessed: 05-Aug-2017].

[28] "Hyperledger Indy Project," https://www.hyperledger.org/projects/hyperledger-indy [Accessed: 05-Aug-2017].

[29] "Hyperledger Burrow Project," https://www.hyperledger.org/projects/hyperledger-burrow [Accessed: 05-Aug-2017].

[30] "Hyperledger Iroha Project," https://www.hyperledger.org/projects/iroha [Accessed: 05-Aug-2017].

[31] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proceedings of the 3rd Symposium on Operating Systems Design and Implementation (OSDI '99)*, pp. 173–186, February 1999.

[32] C. Cachin, "Architecture of the hyperledger blockchain fabric," *IBM Research*, 2016.

[33] E. Androulaki et al., "Cryptography and protocols in hyperledger fabric," *Real-World Cryptography Conference*, 2017.

[34] M. Bartoletti, S. Lande, L. Pompianu, and A. Bracciali, "A general framework for blockchain analytics," in *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers (SERIAL '17)*, December 2017.

[35] X. Xu, I. Weber, M. Staples et al., "A taxonomy of blockchain-based systems for architecture design," in *Proceedings of the IEEE International Conference on Software Architecture (ICSA '17)*, pp. 243–252, April 2017.

[36] B. Koteska, E. Karafiloski, and A. Mishev, "Blockchain implementation quality challenges: a literature review," in *Proceedings of the 6th Workshop on Software Quality Analysis, Monitoring, Improvement, and Applications (SQAMIA '17)*, vol. 1938, September 2017.

[37] R. Yasaweerasinghelage, M. Staples, and I. Weber, "Predicting latency of blockchain-based systems using architectural modelling and simulation," in *Proceedings of the IEEE International Conference on Software Architecture (ICSA '17)*, pp. 253–256, April 2017.

[38] I. Kocsis and A. Klenik, "Towards Performance Modeling of Hyperledger Fabric".

[39] K. Croman, C. Decker, I. Eyal et al., "On Scaling Decentralized Blockchains," *International Financial Cryptography Association*, vol. 1, pp. 1–31, 2016.

[40] J. Rubin, "BTCSpark : Scalable Analysis of the Bitcoin Blockchain using Spark," pp. 1–14, 2015.

[41] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of PBFT consensus process for permissioned blockchain network (hyperledger fabric)," in *Proceedings of the 36th IEEE International Symposium on Reliable Distributed Systems (SRDS '17)*, pp. 253–255, September 2017.

[42] E. Kokoris-Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford, "Enhancing Bitcoin Security and Performance with Strong Consistency via Collective Signing," 2016.

[43] J. Behl, T. Distler, and R. Kapitza, "Scalable BFT for multi-cores: actor-based decomposition and consensus-oriented parallelization," in *Proceedings of the 10th USENIX Conference Hot Topics in System Dependability*, 2014.

[44] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-NG: A Scalable Blockchain Protocol," 2015.

[45] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 9591, pp. 112–125, 2016.

[46] L. Aniello, R. Baldoni, E. Gaetani, F. Lombardi, A. Margheri, and V. Sassone, "A prototype evaluation of a tamper-resistant high performance blockchain-based transaction log for a distributed database," in *Proceedings of the 13th European Dependable Computing Conference (EDCC '17)*, pp. 151–154, September 2017.

[47] K. Suankaewmanee, D. T. Hoang, D. Niyato, S. Sawadsitang, P. Wang, and Z. Han, "Performance Analysis and Application of Mobile Blockchain," pp. 1–6, 2017.

[48] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong, "Performance analysis of private blockchain platforms in varying workloads," in *Proceedings of the 26th International Conference on Computer Communications and Networks (ICCCN '17)*, August 2017.

[49] T. T. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K. Tan, "Blockbench: A framework for analyzing private blockchains," in *Proceedings of the the ACM International Conference*, pp. 1085–1100, Chicago, IL, USA, May 2017.

[50] D. T. T. Anh, M. Zhang, B. C. Ooi, and G. Chen, "Untangling Blockchain: A Data Processing View of Blockchain Systems," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[51] "Hyperledger Calipar," https://github.com/hyperledger/caliper.git [Accessed: 08-Jan-2018].