Performance Analysis of Private Blockchain Platforms in Varying Workloads

Suporn Pongnumkul, Chaiyaphum Siripanpornchana, and Suttipong Thajchayapong
National Electronics and Computer Technology Center, Thailand
Email: suporn.pongnumkul@nectec.or.th

***Abstract*—This paper conducts a performance analysis of two popular private blockchain platforms, Hyperledger Fabric and Ethereum (private deployment), to assess the performance and limitations of these state-of-the-art platforms. Blockchain, a decentralized transaction and data management technology, is said to be the technology that will have similar impacts as the Internet had on people's lives. Many industries have become interested in adopting blockchain in their IT systems, but scalability is an often-cited concern of current blockchain technology. Therefore, the goals of this preliminary performance analysis are twofold. First, a methodology for evaluating a blockchain platform is developed. Second, the analysis results are presented to inform practitioners in making decisions regarding adoption of blockchain technology in their IT systems. The experimental results, based on varying number of transactions, show that Hyperledger Fabric consistently outperforms Ethereum across all evaluation metrics which are execution time, latency and throughput. Additionally, both platforms are still not competitive with current database systems in term of performances in high workload scenarios.**

I. Introduction

Recent years have seen significant growth of blockchain applications both in variations and quantities. Even though well-established blockchain platforms have already been adopted to meet demands of these new applications, more independent and hands-on assessment on performance of these blockchain platforms are still necessary. These information would be essential for practitioners to understand limitations and designate which platform to adopt for their own applications.

The focus on this paper is on performance analysis of private blockchain platforms. Even though blockchain was first introduced to the world as the technology behind Bitcoin, some characteristics of the bitcoin blockchain themselves are unsuitable for certain business applications. In this respect, private blockchain concept is introduced to allow business to use blockchain technology. Unlike in the bitcoin blockchain, only the computers which are granted permission can participate in private blockchain network. Transactions are also processed at a much faster rate compared to Bitcoin's ten minutes per block.

There are still many challenges that lie ahead for adoption of private blockchain platforms. A recent to-be-published paper explores the performance of private blockchain platforms [1] and cited performance evaluation as a research opportunity. Performance is one of the biggest concerns in adopting blockchain platforms as it is necessary to provide a viable alternative to existing financial platforms. Some startups, such as Bitshares [2], are specifically exploring approaches to address this limitation.

An extensive review on research topics on blockchain in [3] has revealed that throughput and latency are amongst the main technical challenges and limitations that have not been widely assessed. Also, preliminary results that reflect large scale deployments are still needed as future blockchain solutions are anticipated to involve tens of millions of people [3]. Quantification of throughput and latency would enable practitioners to have insight understandings of performance and limitations of existing blockchain technologies.

Even though throughput and latency of well-known blockchain platforms have previously been quantified, the scenarios when transactions are made by large number of users have not yet been explicitly assessed. Earlier studies often use bitcoin as a study case for insight analysis such as the relationship between propagation delay and blockchain forks [4]. A more recent study in [5] focuses more blockchain's throughput and latency where the adjustment of block sizes and intervals are proposed as a first step. Recently, a more generic framework has been proposed in [1], where the results on throughput and latency presented in [1] are assessed mainly by varying number of nodes.

In this paper, the analysis is focused on varying number of transactions, which is expected to complement the findings in [1]. The contributions of this paper are summarized as follows. First, a repeatable methodology for evaluating a blockchain platform is presented. This performance evaluation methodology is used to assess the current states of Hyperledger Fabric and Ethereum are presented, where both blockchain platforms are assessed in respect to throughout and latency with up to 10,000 transactions. Second, results from assessing these blockchain platforms and their implications are discussed, which practitioners can take into consideration when adopting for their own applications. We also note that as consensus protocols are found to induce bottlenecks in [1], they are not taken into account in this analysis.

This paper is organized as follows. Section II provides a brief overview of the target blockchain platforms in this analysis. In Section III, the methodology for assessing blockchain platforms is presented. Then, results and their implications are discussed in Section IV. Finally, Section V concludes this paper.

978-1-5090-2991-4/17/$31.00 ©2017 IEEE

II. Blockchain Platforms

This section briefly discusses the target blockchain platforms. Two blockchain platforms are chosen for this evaluation, namely Hyperledger Fabric and Ethereum, due to their popularity and potential in being developed to use in a wide variety of applications.

A. Ethereum

Ethereum [6] is an open-source, public, blockchain-based distributed computing platform featuring smart contract functionality. It extends bitcoin and leverages virtual machine technology (Ethereum Virtual Machine) to allow custom business logic, i.e. smart contract, to be used for new applications. The main platform is a public network where anyone can download the software to run on their computer. The incentive mechanism for users to run the software is to get Ether, which is a digital currency.

While the main Ethereum platform is a public blockchain network, the software is open-source and allows software developers to download and configure the network to be a private network, where computer nodes to participate are those that are granted permission only.

B. Hyperledger Fabric

Hyperledger Fabric [7] is an implementation of private (permissioned) blockchain technology that is intended as a foundation for developing blockchain applications for a wide variety of industry. Therefore, its architecture is modular, allowing components, such as consensus and membership services, to be plug-and-play. It leverages container technology (docker) to enable smart contracts called "chaincode" that comprises the application logic of the system. Hyperledger Fabric is an open-source distributed ledger software built and maintained by the Hyperledger community, which is collaborative effort aimed to advance cross-industry blockchain technologies.

III. Methodology

This section describes the methodology used to evaluate the blockchain platforms. Fig. 1 shows the architecture of the evaluation.

A. Blockchain platforms and Infrastructure Setup

Two blockchain platforms are investigated in this evaluation, Hyperledger Fabric and Ethereum. The infrastructure that the experiments are conducted on is Amazon AWS EC2 (c4.2xlarge instance) with the Intel E5-1650 8 core CPU, 15GB RAM, 128GB SSD hard drive and running Ubuntu 16.04. For each platform, one blockchain node is deployed by downloading and installing appropriate softwares, which are Ethereuem's geth 1.5.8 [8] and Hyperledger Fabric 0.6 [9]. As one blockchain node is deployed, consensus mechanism is turned off by configuration and consensus protocol is excluded from this study as later discussed in Section IV-D.

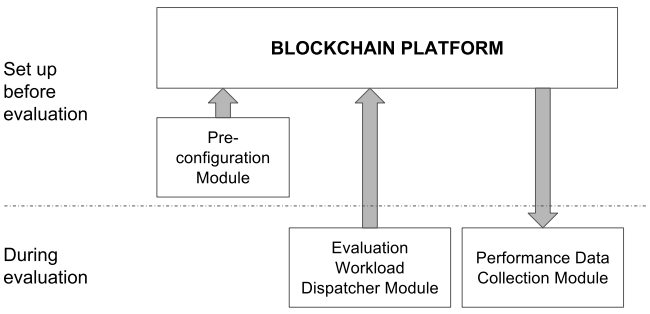


Fig. 1: Evaluation architecture. There are four main modules in the evaluation architecture: the target blockchain platform, the pre-configuration module, evaluation workload dispatcher module, and performance data collection module. The first two modules are set up prior to the evaluation. The last two modules run during the evaluation, where the workload dispatcher sends the transactions to the blockchain platform and the performance data collection module collects the transaction statuses.

B. Pre-configuration: Simulated Application and Smart Contracts

In order to evaluate the platforms, we construct a synthetic application for the experiments as a cash transfer application, where a user account can be created (with function **CreateAccount**), money can be issued to an account (with function **IssueMoney**), and money can be transferred from one account to another (with function **TransferMoney**). The implementation of each function is done separately for each platform. In Ethereum, we create an account by specifying a passphrase, which is required to decrypt the private key on disk. In Hyperledger Fabric, we create two different key-value tuples in the contract, one for storing the account name, and another for account balances. To issue money to an account, we add the money into an account by specifying account name as the key. Finally, cash can be transfered between accounts by subtracting money from the source account and adding the same amount to the target account. TransferMoney's code snippets of Ethereum smart contract and Hyperledger Fabric chaincode are shown in Fig. 2 and 3 respectively. During pre-configuration, all smart contracts are written and deployed for each platform ready to be invoked during evaluation period.

C. Evaluation Workloads and Clients

We perform experiments where a client sends N requests of transactions of type f to a target blockchain platform, B, in an asynchronous manner, i.e. all requests are sent without waiting for a response from the blockchain. The number of requests (N) are set to 1, 10, 100, 1000, and 10000 requests. The transaction type (f) can be CreateAccount, IssueMoney and TransferMoney. There are two options for blockchain platforms (B): Ethereum and Hyperledger Fabric. The data (e.g. the amount of money or the account to be transferred to) are simulated by randomization. In addition, the results of each experiment are averaged over ten independent runs.

```
contract TransferMoney {
  mapping (address => uint) balances;
  ...
  function sendCoin(address receiver,
  uint amount) returns(bool sufficient)
  {
    if (balances[msg.sender] < amount)
    return false;
      balances[msg.sender] -= amount;
      balances[receiver] += amount;
      return true;
  }
  ...
}
```

Fig. 2: A code snippet from TransferMoney function for Ethereum smart contract, written in Solidity.

```
...
func (t *Chaincode) TransferMoney
    (stub shim.ChaincodeStubInterface,
    args []string) ([]byte, error) {
  Avalbytes, err := stub.GetState(A)
  Aval = strconv.Atoi(string(Avalbytes))
  ...
  Aval = Aval - amount
  Bval = Bval + amount
  ...
  stub.PutState(A,
  []byte(strconv.Itoa(Aval)))
  ...
}
...
```

Fig. 3: A code snippet from TransferMoney function for Hyperledger Fabric chaincode (i.e. smart contract), written in Go.

The interactions between the client and the blockchain platform are accomplished with HTTP requests in Node.js application. For Ethereum, all of queries are implemented via web3.js and it communicates to a local node through JSON-RPC call APIs. For Hyperledger Fabric, all of queries are implemented with the RESTful APIs.

D. Performance Data Collection

This subsection explains the data that are collected for the analysis and the evaluation metrics used to discuss the results.

1) Transaction Data: In order to evaluate the performance of blockchain platforms, data are collected for each transactions as follows:

- **Transaction deployment time** (t_1) is the unix time when transaction was deployed.
- **Transaction completion time** (t_2) is the unix time when transaction was confirmed by the blockchain. In order to get the completed time, for Ethereum, this data is collected directly by web3js APIs that return transaction details. For Hyperledger Fabric, it is implemented via a chaincode that can connect to a peer and receive block events.

2) Evaluation Metrics: The parameters chosen to be measured for this evaluation are execution time, latency and throughput.

Execution Time For each set of transactions, the *execution time* is the total amount of time (number of seconds) during which the blockchain platform took to execute and confirm all transactions in the data set (maximum t_2 - minimum t_1).

Latency For each transaction, the *latency* is the difference between the completion time and the deployment time ($t_2 - t_1$). For a set of transactions, the *average latency* is the average of latency of all transactions in the data set.

Throughput *Throughput* is measured as the number of successful transactions per second starting from the first transaction deployment time. *Average throughput* is the average of the throughput over the execution time.

IV. Results and Discussion

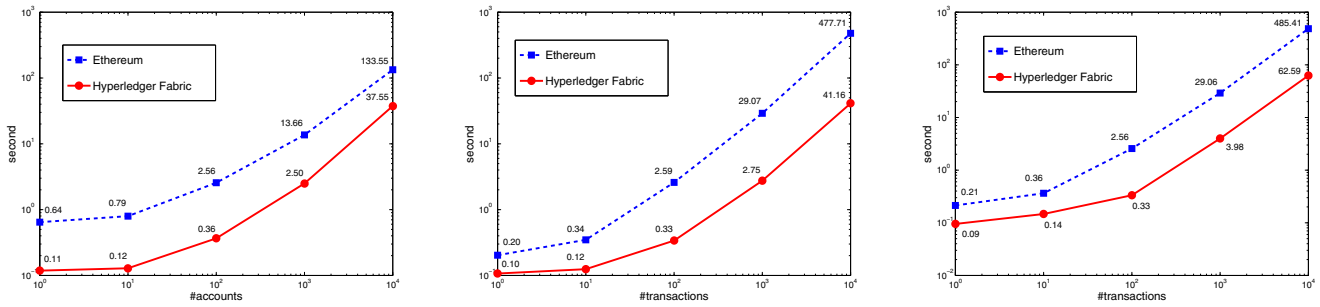
A. Performance Assessment

In this section, we assess the performance of blockchain platforms in term of the average execution time, average latency and average throughput. It can be observed that Hyperledger Fabric outperforms Ethereum across all scenarios.

Comparing Execution Time We explore the differences in execution time of varying number of transactions, with different platforms and different functions in Fig. 4. The execution times grow as the number of transactions in the data set increases. Hyperledger's execution time is consistently lower than Ethereum's in all data sets. The gap between the execution time of Hyperledger's and Ethereum also grows larger as the number of transactions grow.

We note that CreateAccount utilizes a native function of both system as creating an identity is native functionality provided, while IssueMoney and TransferMoney are custom functions written for this synthetic application. At large number of transactions, Ethereum's execution time is considerably lower for CreateAccount (133.55 seconds) than IssueMoney (477.71 seconds) and TransferMoney (485.41 seconds). IssueMoney adds money to one account, while TransferMoney subtracts money from one account and adds the same amount to another account. The amount of workload for IssueMoney is approximately half of the amount of workload for TransferMoney. For the batch of 10,000 transactions, IssueMoney and TransferMoney takes 41.16 and 62.59 seconds respectively for Hyperledger, and 477.71 and 485.41 seconds respectively for Ethereum. This results show a large difference in data access and management for the two platforms.

Comparing Average Latency Fig. 5 shows the log-log plot of average latency that TransferMoney transactions experience in five sets of experiments for each platform. For the data set with one transaction, the average latency of Hyperledger is 0.09 seconds and the average latency of Ethereum is 0.21 seconds. At low number of transactions, Ethereum's latency is about 2x of Hyperledger's. As the number of transactions in the data set grows, Ethereum's latency is considerably worse than Hyperledger's. Similar to the log-log plot of average latency, Fig. 6 shows a comparison of average latency in five sets of experiments for each platform. It can be observed that when number of transactions in the data set grows, the average latency of both platforms increases rapidly. Specifically,



(a) Create accounts (b) Issuance (c) Transfer value

Fig. 4: Execution time of Ethereum and Hyperledger with varying number of transactions (1,10,100,1000,10000 transactions) in log-log scales. Subplots show the results of three different accounts: (a) CreateAccount, (b) IssueMoney, (c) TransferMoney.

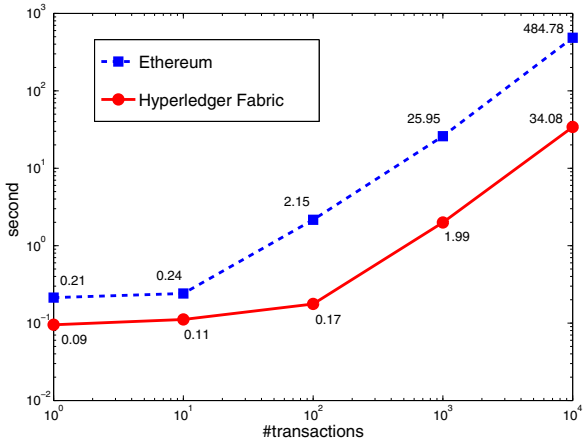


Fig. 5: Average latency of Ethereum and Hyperledger with varying number of transactions of TransferMoney function.

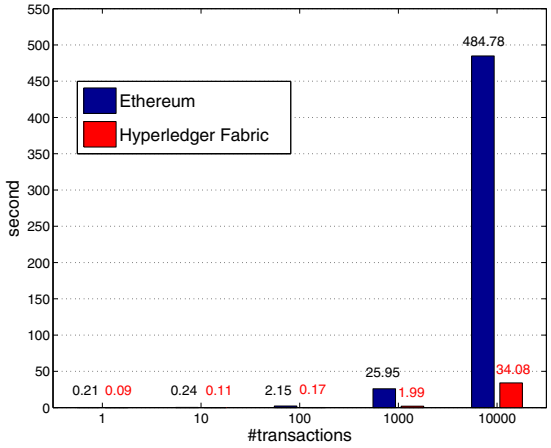


Fig. 6: Comparison of average latency between Ethereum and Hyperledger

when increasing number of transactions from 1000 to 10000 transaction, the average latency of Ethereum and Hyperledger is up to 18.67x and 17.09x, respectively.

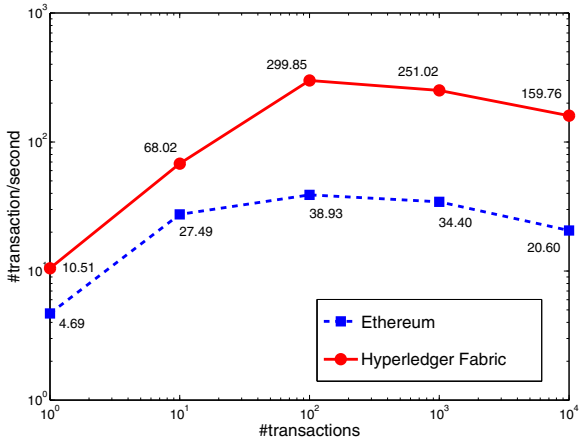


Fig. 7: Average throughput of Ethereum and Hyperledger with varying number of transactions of TransferMoney function.

Comparing Average Throughput Fig. 7 shows the log-log plot of average throughput that TransferMoney transactions experience in five sets of experiments for each platform. Hyperledger has higher throughput than Ethereum in all of the data sets. We observe that the average throughputs of both platforms maximizes when the number of transactions in the data sets is 100.

Similar to the log-log plot of average throughput, Fig. 8 shows a comparison of average throughput in five sets of experiments for each platform. It can be observed that when varying the number of transactions, the change of average throughput of Hyperledger is relative larger than that of Ethereum.

Latency and Throughput of One Trial of Large Workload Fig. 9 and Fig. 10 explore how individual transaction in the data set experience latency and the throughput where 10,000 transactions are deployed. Fig. 9a shows that Ethereum has a long latency for all transactions, where the first transaction is confirmed after 361.36 seconds, and gradually confirms

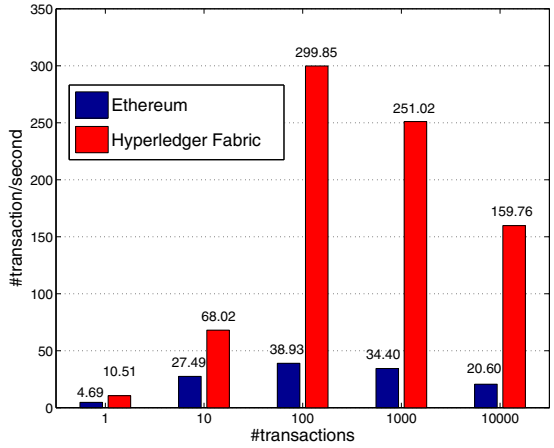
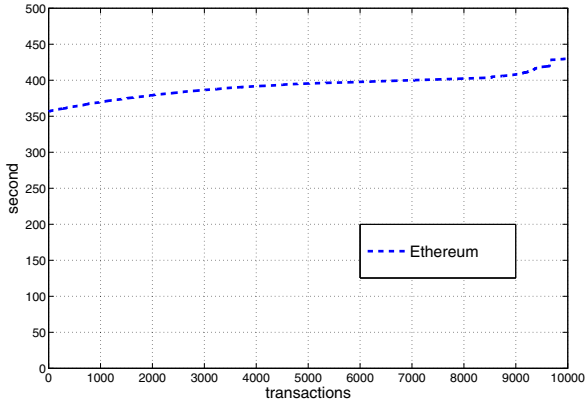
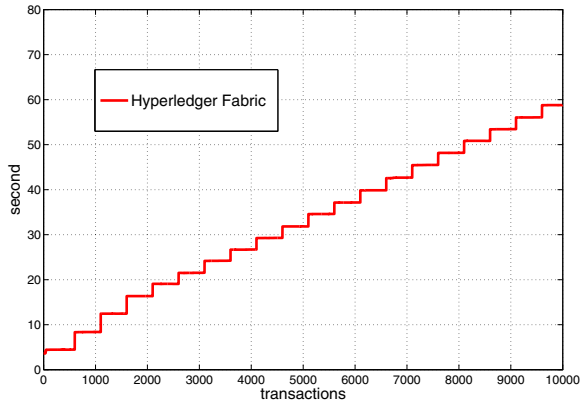


Fig. 8: Comparison of average throughput between Ethereum and Hyperledger

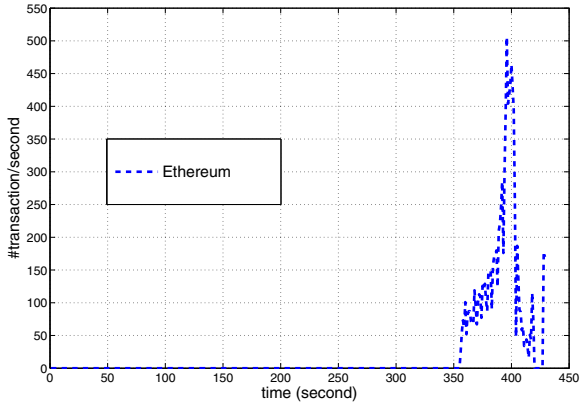


(a) Ethereum

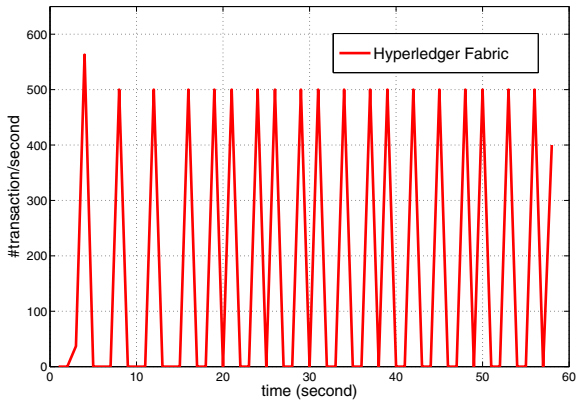


(b) Hyperledger

Fig. 9: Latency of individual transactions where 10,000 transactions are requested



(a) Ethereum



(b) Hyperledger

Fig. 10: Throughput where 10,000 transactions are requested

all transactions after that. Fig. 9b shows that Hyperledger confirms the first transaction after 3.57 seconds, and confirms transactions in batches of 500 transactions.

The large latency that the transactions experience on Ethereum leads to an investigation of the minimum latency, as plotted in Fig. 11. It can be seen that the extreme latency occurs when the number of transactions is large.

B. Maximum concurrent transactions

In order to test the capacity and determine limitation of each platform, we deployed concurrent transactions and report the maximum number of concurrent transactions that each platform can handle. TransferMoney transaction is chosen for this test. Starting from 10,000 the number of concurrent transactions is increased by 10,000, until we find the number of concurrent transactions that the platform cannot handle. Failure is when the platform reports failure or fails to respond within 10 minutes. The results show that Hyperledger Fabric is able to accept 20,000 concurrent transactions, while Ethereum is able to handle 50,000 concurrent transactions. In addition, by observing the resource utilization, as the number of concurrent transactions increases, all CPUs are fully utilized.

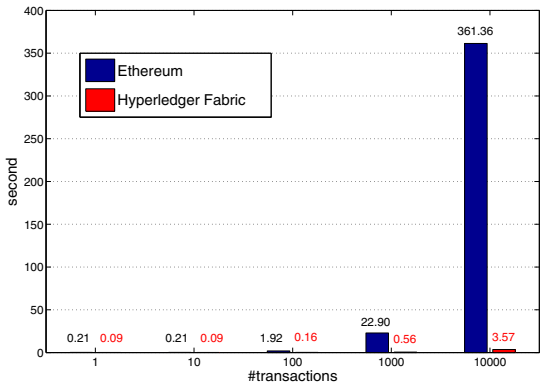


Fig. 11: Comparison of latency when the first transaction's execution between Ethereum and Hyperledger

However, RAM and HDD I/O are not fully utilized in both blockchain platforms.

C. Implications

The analysis in this paper shows that Hyperledger Fabric consistently performs better than Ethereum both in term of throughput and latency, which is also consistent with the findings from the work of Dinh et al [1]. As a compliment to [1], the findings in this paper show that the difference between Ethereum and Hyperledger Fabric becomes even more significant with larger number of transactions.

A direct implication is that, for a given blockchain application, estimating expected number of transactions will be very crucial in selecting suitable platforms as they can alter subsequent throughput, execution time, and latency. Particularly, latency can play a crucial role in applications involving money transfer as well as other forms of trading. We note also that the latencies presented in this paper should be considered the minimum possible latency that can occur when adopting these two private blockchain platforms. At the application level, more calculation/logical processes may be introduced which can induce even larger latencies. Lastly, even though Hyperledger Fabric outperforms Ethereum in all aspects, our findings also show that Ethereum is able to handle more concurrent transactions for similar computational resources.

D. Limitation of the study

Analysis of execution layer of the platforms The setup for this evaluation intentionally analyzes the execution layer of the target blockchain platforms. Consensus layers are excluded from the analysis by configuration. While the distributed aspect is a crucial part of blockchain platforms, the reason behind this deliberate choice is because the two target platforms utilize different consensus protocols, which the nature of the protocols directly impact the performance. The nature of Proof-of-Work mechanism of Ethereum's consensus mechanism is much slower than the nature of PBFT mechanism that is deployed in Hyperledger Fabric. As a result, the set up for evaluation in this paper evaluates the capacity and limitation of the execution layer (smart contract infrastructure) of both platforms. The presence of consensus provides security, but reduces the performance of the system; therefore, the results presented in this paper provide the best case scenario of the performance. We will explore the effects of the consensus protocols in our future work.

Analysis of the current version of the platforms This paper evaluates and compares two blockchain platforms using the latest version available at the time of the study. While this might not represents what the blockchain platforms will be in the future due to the continuous improvements of the blockchain platforms, this fills the gap of the knowledge about the capacity and limitation of the state-of-the-art blockchain platforms. The results presented in this paper should be of interests to blockchain practitioners looking to adopt blockchain in their applications or solutions.

V. Conclusion

This paper presents performance analysis of Ethereum and Hyperledger Fabric as private blockchain platforms with varying number of transactions. Assessment shows that Hyperledger Fabric achieves higher throughput and lower latency compared to Ethereum when the workloads are varied up to 10,000 transactions. Also, differences between these two platforms in respect to execution time and average latency become more significant as the number of transactions grow. The average throughput of Hyperledger Fabric also changes at a much faster rate than that of Ethereum. However, for similar computational resources, Ethereum is able to handle more number of concurrent transactions. For future work, we plan to perform assessment with consensus protocols and with larger number of transactions on newer versions of private blockchain platforms. Furthermore, we are interested in exploring the performance differences between private blockchain platforms and public blockchain platforms. Another aspect worth further investigation is to assess fundamental differences in code-level implementations between the two platforms and how they influence the overall performances.

References

[1] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "Blockbench: A framework for analyzing private blockchains," *arXiv preprint arXiv:1703.04057*, 2017.

[2] "Industrial performance and scalability - bitshares," https://bitshares.org/technology/industrial-performance-and-scalability/, accessed: 2017-03-23.

[3] 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, p. e0163477, 2016.

[4] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *IEEE P2P 2013 Proceedings*, Sept 2013, pp. 1–10.

[5] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. Gün Sirer, D. Song, and R. Wattenhofer, *On Scaling Decentralized Blockchains*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 106–125.

[6] "Ethereum project," https://ethereum.org/, accessed: 2017-03-10.

[7] "Hyperledger blockchain technologies for business," https://www.hyperledger.org/, accessed: 2017-03-10.

[8] "Ethereum : Release peachest (v1.5.8)," https://github.com/ethereum/go-ethereum/releases/tag/v1.5.8, accessed: 2017-03-10.

[9] "hyperledger/fabric at v0.6," https://github.com/hyperledger/fabric/tree/v0.6, accessed: 2017-03-10.