Report: Hyperledger and Ethereum Write Latency Benchmark

Pablo Osinaga, Robertus Vincent, Konrad Stegemann, and Ayaz Ali Qureshi
Technische Universität Berlin
Berlin, Germany

https://github.com/paguos/blockmark

I. Introduction

Over the past few years, blockchain technology has started to flourish. Real world use-cases on blockchain technology keep increasing. Cryptocurrency, mobile payment system, and security are some popular examples. This wide range of use-cases also lead the developers to develop a lot of varieties of blockchain ecosystem, which enables them to build an application in a specific method.
In this project, we will discuss and compare the performance of two of blockchain systems based on their write latency, namely, Ethereum and Hyperledger. These two blockchain systems have fundamentally different network permission. Ethereum is used mainly as a permissionless (public) network and Hyperledger is a permissioned (private) network. However, for benchmarking purpose, a private network will be built for Ethereum as well to make both network comparable.

II. System Under Test (SUT)

To begin with this project, it is important to establish an understanding of the concepts of the technology that are used during these tasks.

A. Ethereum

Ethereum is an open blockchain platform that focuses on letting anyone built a decentralized application (Dapp) that run on blockchain technology. It is a programmable blockchain, that runs through Ethereum Virtual Machine (EVM).

An Ethereum network is a public network, but a private or private network can also be built on that. In this context, private only means reserved or isolated, rather than protected or secure. Ethereum is currently powered by Proof of Work (PoW) for mining its blocks. This consensus protocol is primarily based on solving costly computer computations, also called mining, to verify the legitimacy of a transaction or avoiding double-spending. Whenever a new block is added, the blockchain updates and propagates to the entire network, so that all nodes are in sync.

PoW based network does not have an authority who authorizes a transaction on the ledger. Anyone can join as a public node, validate transactions and participate in the consensus process (mining) without permission.

However, a PoW based network requires the miners to spend a lot of resources (i.e. computers and electricity). Proof of Stake (Pos) is a green alternative to PoW and Ethereum is currently working to switch its network to a PoS based network.

Because blockchain is a system that exists between all permitted parties, it also introduces smart contract, otherwise also called self-executing contracts. A smart contract allows the user to do a transaction in a transparent, conflict-free way while avoiding the services of a middleman. It enables the network to create a platform for distributed applications. In Ethereum, a smart contract is written in Solidity.

More information about Ethereum can be found on the documentation site: *http://www.ethdocs.org/*

B. Hyperledger

Hyperledger is a blockchain consortium under 'The Linux Foundation' and different from Ethereum, it is a private network. It has one or multiple authorities that control the network, it means one requires an invitation and must be validated by either the network authorities or by a set of rules set in the network. The access control mechanism could vary: existing participants could decide future entrants; a regulatory authority could issue licenses for participation, or a consortium could make the decisions instead. Once an entity has joined the network, it will play a role in maintaining the blockchain in a decentralized manner. Network's participants also cannot do a transaction (read/write to the blockchain network) unless it has the permission to do so. In private network such as hyperledger, the owner of the network can override/delete permission that its participants have if needed. In this sense, this network is not decentralized.

However, in comparison to a public network, it is much faster and cheaper because one doesn't have to spend a lot of resources to reach a consensus. But, it is also less secure because it can be read or written as when wished and deemed fit by the owner/benefiting parties.

An important aspect of Hyperledger is that it does not require cryptocurrencies for transactions. It does not have a built-in native cryptocurrency like Ethereum's token, Ether. Hence, there is no requirement of mining at all. This allows

for scalable consensus algorithm that is capable of handling high transaction rates required by most enterprise applications.

Hyperledger Fabric, which is used for this project, is an open source enterprise-grade permissioned distributed ledger technology (DLT) platform, designed for use in enterprise contexts. As it is a private network, the participants are known to each other, rather than anonymous.

Hyperledger Fabric supports pluggable consensus protocol that enables the platform to be more effectively customized to fit particular use cases and trust models. For example, when operated within a single enterprise or by a trusted authority, fully Byzantine fault tolerant (BFT) consensus might be considered an unnecessary and excessive drag on performance and throughput. In such a situation, a crash fault-tolerant (CFT) consensus (Kafka) might be adequate whereas, in a decentralized use-case, a BFT consensus might be required.

The consensus that we are going to used is called SOLO, which is the simplest consensus that is available for Hyperledger 1.2. It only involves a single ordering node and only broadcasts the transaction without establishing any real consensus. However, this consensus is not suitable for production as it is only intended for developers to experiment with Hyperledger Fabric network.

While public network such as Ethereum requires a PoW-based consensus to execute and validate a transaction, Hyperledger 1.2 introduces a new consensus that they call execute-order-validate. It breaks the consensus into 3 phases:

- *execute* a transaction and check its correctness, thereby endorsing it,
- *order* transactions via a consensus protocol,
- *validate* transactions against an application-specific endorsement policy before committing them to the ledger.

In Hyperledger Fabric, a transaction needs to be executed through a smart contract, or what they call a chain code. It is the central element as transactions are operations invoked on the chaincode. Hyperledger Fabric allows chaincode to be authored in general-purpose programming languages such as Java, Go and Node.js, rather than constrained domain-specific languages (DSL).

Hyperledger Fabric as a private network also provides confidentiality to their clients, what a public network cannot do because of their transparencies. Participants on a Hyperledger Fabric network can establish a channel between a subset of participants that should be granted visibility to a particular transaction.

More information about Hyperledger Fabric can be found in: *https://hyperledger-fabric.readthedocs.io/*

The following chart describes briefly the comparison between both blockchain networks:

Features	**Hyperledger**	**Ethereum**
Confidentiality	Confidential Transactions	Transparent
Participation	Private (Permissioned)	Public (Permissionless)
Consensus	Execute-Order-Validate No mining required	PoW / PoS Mining required
Prog. Language	Chaincode written in Java, Go, or Node.js	Smart Contracts written in Solidity
Cryptocurrency	No Cryptocurrency	Ether

III. Benchmark Metric

In the blockchain technology, latency describes the time it takes from the creation of a transaction until the initial confirmation of it being accepted by the network (and how the confidence of acceptance increases over time).

Here, we want to measure the write (transaction) latency in both networks. We define write latency as the amount of time taken for a transaction to be accepted and available across all nodes/peers in the network. This includes the propagation time and any settling time due to the consensus mechanism in place.

IV. Methodology

Node Topology

Fundamentally, Hyperledger and Ethereum work in a different concept, which makes these two networks not comparable. For the purpose of the benchmark, we built the ethereum network to reflect the hyperledger network. Where all nodes will be connected with only one node; main node (in Ethereum) /ordered (in Hyperledger).

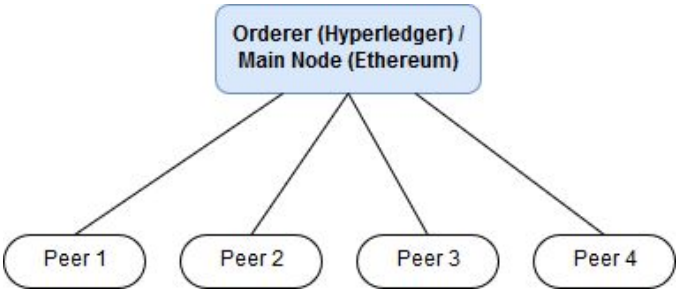

Figure 1: Network Topology

Because there is no such thing as consensus difficulty in Hyperledger due to Hyperledger Certificate Authority, we set the consensus difficulty in Ethereum to the minimum to make both networks as comparable as possible.

Workload

For benchmark purpose, we set up a simple workload scenario where the client wants to send an amount of money to someone. As seen in Figure 2, we will have two participants in the network, one Sender and one Receiver. The Sender will send a fixed amount of money to the Receiver 100 times. In the process, the transaction submission time and the time when the transaction is accepted in the network will be captured.

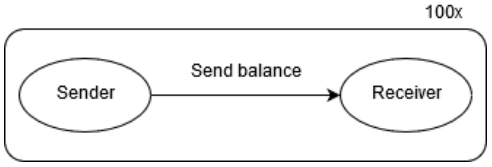

Figure 2: Workload Scenario

Deployment

In the Ethereum network, the network is deployed with five peers and two of them are miners. After setting up all accounts for each peer and creating the genesis block, the network is ready and the miners will begin mining in the background.

Benchmark experiment in Ethereum network will be done through web3.js. The official Javascript library to interact with ethereum networks. To warm up the network, two ether will be sent from the sender to the receiver account and sent back by the receiver. This process repeats itself three times. After that, the workload for this benchmark is executed; Sending a fixed amount of balance (by default one ether) from sender node to receiver node 100 times, capturing the starting time and the time the receiving node receives the transaction receipt, meaning that the transaction can be viewed by all the node in the network. Both timestamps are used to calculate the latency.

In the Hyperledger Fabric network, the network is built through docker (based on IBM Fabric-sample repository). Here, the network is deployed with one Certificate Authority, one Orderer, two Organization with two peers each. In the network (or channel in this case) will then be 4 peers communicating with the orderer.

Benchmark experiment in the Hyperledger Fabric network will be done through chain code (smart contract in Hyperledger). The chain code performs a transaction of sending a fixed amount of balance from sender account to receiver account. While executing the workload of sending the transaction 100 times, the submission time and response time will be captured to calculate the latency.

V. Results

As seen in Figure 3, there is a significant difference between the write latency in the Hyperledger network and the Ethereum network. Ethereum's write latency fluctuates from 2 seconds to around 8 seconds with 2 spikes that reach 14 and 16 seconds. In comparison to Ethereum, Hyperledger's write latency as seen in Figure 4 only fluctuates from 2,17 seconds to 2,23 seconds. From Figure 3, it can also be seen that in general Hyperledger transaction is not only faster but also more stable in term of the write latency.

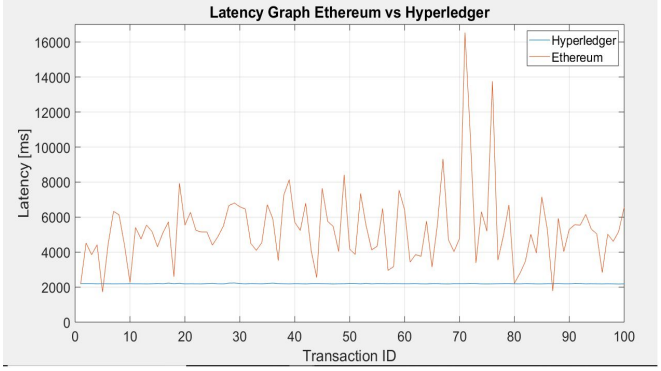

Figure 3: Write Latency Ethereum and Hyperledger

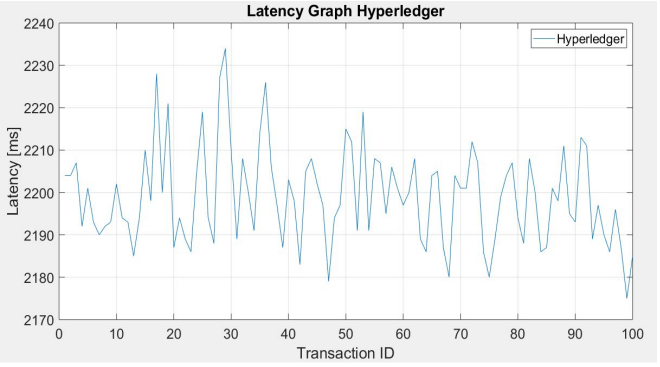

Figure 4: Write Latency Hyperledger

VI. Conclusion

As seen in Figure 3, overall we can see that Hyperledger performs faster than Ethereum. This result is to be expected because of the difference in the consensus algorithm. As a private network, Hyperledger does not require mining to validate a transaction. The client only needs to endorse the transaction to achieve a certain number of peers for the transaction to be validated by ordered and added to the network. In contrast to Hyperledger, PoW based Ethereum network requires its miner to mine for the block to validate the transaction. As mining involves solving a complex problem, it will require more time for a transaction/block to be committed.

Also when observing the results of the Latency Benchmarking Ethereum vs Hyperledger, we can see that

Hyperledger has a more stable latency time (Δtmax= 60 ms) in comparison with Ethereum (Δtmax=14809 ms) which is the vivid representation of the difference in speed between them, making Hyperledger a better platform for private network scenario.

VII. Bibliography

1. Ethereum Web Site (2018) *Ethereum/Go-Ethereum.GitHub,* Available at: *github.com/ethereum/go-ethereum/wiki/Private-network.* (Accessed: 25th January 2019).

2. Blockgeeks Web Site (2018) *Proof of Work vs Proof of Stake: Basic Mining Guide,* Available at: *https://blockgeeks.com/guides/proof-of-work-vs-proof-of-stake/* (Accessed: 25th January 2019).

3. ‘‘Sudhir Khatwani (2018) *What Are Private Blockchain & How Are They Different From Public Blockchains?,* Available at: *https://coinsutra.com/private-blockchain-public-blockchain/* (Accessed: 25th January 2019).

4. Praveen Jayachandran (2017) *The Difference Between Public And Private Blockchain,* Available at: *https://www.ibm.com/blogs/blockchain/2017/05/the-difference-between-public-and-private-blockchain/* (Accessed: 25th January 2019).

5. Hyperledger Web Site (2019) *A Blockchain Platform for the Enterprise,* Available at: *https://hyperledger-fabric.readthedocs.io/* (Accessed: 25th January 2019)

6. Ethereum Community (2019) *Ethereum Homestead Documentation,* Available at: *http://www.ethdocs.org/en/latest/* (Accessed: 25th January 2019).

7.