Computer Science and Information Systems 19(2):659–678 https://doi.org/10.2298/CSIS210507002W

Performance and Scalability Evaluation of a Permissioned Blockchain Based on the Hyperledger Fabric, Sawtooth and Iroha

Arnold Woznica and Michal Kedziora[1]

Wroclaw University of Science and Technology, Wroclaw, Poland
michal.kedziora@pwr.edu.pl

Abstract. This paper shows the performance and scalability evaluation of different blockchain platform implementations. Hyperledger Iroha implementing YAC consensus, Sawtooth implementing PoET algorithm, and Hyperledger Fabric framework implementation. Performance evaluation and scalability assessment were done by varying different sets of parameters such as block size, transaction sending rate, network traffic distribution, and network size. Performance evaluation was done based on average transaction latency, network throughput, and transaction failure rate. Scalability was assessed based on changes in transaction latency and throughput with increasing network size. Test results let to study the impact of a particular parameter on the private blockchain network performance and show how they can be adjusted to improve performance.

Keywords: Blockchain, Hyperledger Fabric, Sawtooth, Iroha.

1. Introduction

The popularity of blockchain technology paid attention to various businesses that want to adapt it to achieve their business goals. In a public blockchain network, everyone can join the network and be its participant without any authentication. To prevent fraud and malicious operations on the blockchain network, a particular blockchain platform must provide appropriate security mechanisms. In public blockchains, security is generally provided by the underlying consensus algorithms, such as proof of work in Bitcoin or proof of stake in Ethereum[12]. The consensus is the process by which a network of nodes provides a guaranteed ordering of transactions and validates the block of transactions [9]. High scalability is another issue that public consensus protocols must handle well. Unfortunately, those factors result in low throughput and high transaction latency in those systems, i.e, Bitcoin has throughput around 3 transactions per second and transaction latency around 10 minutes[14]. On the other hand, companies adapting blockchain technology work in a partially trusted environment, therefore the security of business blockchain networks might be resolved with mechanisms other than used consensus algorithms[11][10][3]. Scalability in business blockchain networks is also a less important factor. Business solutions need high throughput and low latency blockchain networks to handle quickly big amounts of transactions and that are the major demands on the consensus algorithms used. In 2015, companies, such as IBM and Intel, joined their efforts to create a common business blockchain framework under the Hyperledger project. Hyperledger is an open-source collaborative effort created to advance cross-industry blockchain

technologies. It is a global collaboration, hosted by The Linux Foundation, including leaders in finance, banking, Internet of Things, supply chains, manufacturing, and Technology [5]. It is a very fast-growing project concentrated on the development of a common framework for private and permissioned networks.

The goal and contribution of this work is evaluation of the performance of private blockchain platform implementations and algorithms. As there are many different consensus algorithms [16] the range of algorithms under evaluation is bounded to algorithms used in private and permissioned blockchain networks. The incentive behind such boundaries was a very fast growing Hyperledger project. Its growth means there is a very big demand on the market for private and permissioned blockchain solutions. The work aims to assess those consensus algorithms which are implemented in existing solutions. This work contains a performance evaluation of consensus protocols implemented in the following platforms: Hyperledger Fabric v1.4.0, Hyperledger Sawtooth v1.0.5, and Hyperledger Iroha v1.0.0_rc5. Performance Evaluation is performed by measuring: transaction latency, throughput, network, and scalability. It was done by testing with a different set of parameters such as transaction sending rate, network size, block size, and network traffic distribution.

2. Related work

This section describes articles that concentrate on the performance analysis of different consensus protocols of private blockchain platforms. Contributions of those papers were ideas on how to improve performance and they are implemented in newer versions of Hyperledger Fabric. There were not found any existing articles regarding performance analysis of Hyperledger, Iroha, and Sawtooth platforms, this makes a gap in the research field.

Nasir et al. [15] is the most related work to this paper. Authors concentrate on the performance evaluation of Hyperledger fabric v0.6 and fabric v1.0, those versions differ significantly, also other consensus protocols are used. Fabric v0.6 uses Practical Byzantine Fault Tolerance with order-execute architecture, on the other hand, fabric v1.0 uses Kafka ordering service with execute-order-validated architecture. To assess the performance, measurements such as transaction latency, execution time, throughput, and scalability are taken. The authors deployed the whole blockchain network on a single server machine with 24 core CPU and performed two tests: the first was an evaluation for a single peer network, the second was assessing scalability by varying the number of nodes to 20 in the network while performing the first test for each network topology. In the results section, the authors claimed that fabric v1.0 outperforms fabric v0.6. The second observation was that for fabric v1.0 the latency is within a certain range for different scenarios, while for fabric v0.6 the latency increases with network size.

Thakkar et al. [23] contains a very comprehensive performance evaluation of Hyperledger Fabric v1.0. The authors of that paper tried to answer the following questions: What should be the block size to achieve lower latency? What type of endorsement policy is more efficient? What is the performance difference between CouchDB and GoLevelDB when they are used as local world state databases? Authors divided transaction commit latency into endorsement, broadcast commitment, and ordering latency, which was measured by analyzing Hyperledger Fabric logs. The created test network was distributed

over different machines and consisted of four organizations with two endorsed peers each. For each experiment, many observations with guidelines on how to improve performance were given. Experiments included checking the impact of transaction arrival rate, block size, endorsement policy, ledger, database channels, and resource allocation. The main contributions of this paper were identifying three major performance bottlenecks, providing and studying improvements, changing the overall performance 16-times. All proposed improvements are contained in newer versions of the Hyperledger Fabric platform.

Sousa et al. [19] proposed a new Byzantine fault-tolerant Ordering Service for Hyperledger Fabric 1.0 instead of crash fault Kafka Ordering Service cluster. The proposed solution is based on BFT-SMaRt and WHEAT protocol. The authors implemented their solution and tested in LAN and WAN networks with nodes geolocated across the Americas.

Rüsch [18] mentions the performance and security issues of Byzantine fault tolerance schemes in blockchain and does an investigation for permissioned networks. The author refers to other papers to show the scalability problems of Byzantine fault-tolerant protocols for an increasing number of nodes.

Dinh et al. [6] presented the first Benchmark for private Blockchain evaluation called Blockbench. Blockbench can have an extended backend to evaluate different blockchain platforms. At the time of writing this article backend to test Ethereum, Parity, and Hyperledger fabric existed. Blockbench has a set of different macro and microbenchmarks to test different aspects of the working blockchain network. Blockbench supports evaluation of security by simulation network-level attacks. The authors compared three different platforms: Ethereum, Parity, and Hyperledger Fabric v0.6 by doing comprehensive tests. Empirical results have shown that Hyperledger Fabric v0.6 outperformed Ethereum and Parity across different benchmarks, but failed to scale over sixteen nodes. Their results have shown that tested platforms are not well suited to large-scale data processing workloads.

Li et al. [13] proposes a new architecture to improve blockchain network scalability. New architecture proposes satellite chains to create a network of networks, and it mentions the problem of scalability of single Byzantine Fault Tolerant based networks like Hyperledger Fabric v0.6. The presented architecture provides the ability to transfer assets between different networks in a secure way.

Gorenflo et al. [8] propose performance improvements on Hyperledger Fabric v1.2 without the change of its API. Authors test their implemented ideas to show that this platform can have a throughput of 20 000 transactions per second. Improvements consisted of I/O operations, caching, parallelism, and efficient data access. In their design, orderers receive only transaction IDs instead of full transactions, and validation on peers is more parallelized. Authors determined critical paths and leveraged light-weight data structures for fast data access.

Sukhwani et al. [20] [22] [21] presented a Stochastic Reward Net performance model of Hyperledger Fabric v1.0. This model can be used to compute the throughput, utilization, and mean queue length for each peer in a network. The model is validated with the usage of Hyperledger Caliper. Authors discovered the performance bottlenecks of the ordering service and ledger write operations. Authors claimed that the bottleneck can be mitigated using larger block sizes, although with transaction latency increase.

Turki et al. [24] proposed another Ordering service for Hyperledger fabric v1.0+ built on top of the HoneyBadgerBFT protocol. The authors tested their solution on a local machine and compared it to the results of Solo Ordering Service available for the Hyperledger Fabric v1+ platform.

Feng et al. [7] proposed a new Byzantine Fault Tolerant consensus called scalable dynamic multiagent PBFT (SDMA-PBFT), which is a modification of Practical Byzantine Fault Tolerance protocol. The proposed method decreases latency and improves efficiency and throughput.

Angelis et al. [4] compared the Practical Byzantine Fault Tolerance algorithm with proof of authority algorithms(Aura and Clique). Performance analysis was qualitative and based on how the algorithms worked in terms of message exchanging.

Vukolic [25] performed a theoretical comparison of Byzantine fault-tolerant algorithms versus Proof of work consensus algorithms with a literature review describing what was done to improve their performance and scalability.

Bakaman [1] made a comprehensive review of the state-of-the-art blockchain consensus algorithms. Then proposed an analytical framework to evaluate the pros and cons of consensus mechanisms.

Rui Wang [26] analyzed four mainstream blockchain systems (Ethereum, Fabric, Sawtooth and Fisco-Bcos), and then performed a performance comparison through open source blockchain benchmarking tools. After that, they propose optimization methods and discuss the future development of blockchain technique.

3. Methodology

Each of the evaluated platforms implements a different consensus mechanism. To test the performance of consensus, there is a need to submit transactions, which change the World State of a particular blockchain network. It is important to notice, that the database operation needed to trigger consensus mechanisms has an impact on the performance of a blockchain network. To reduce this impact simple transactions were needed to be implemented. This 'Simple' test works in the following way: Every transaction generates some random number based on the system time, which is appended to a literal value. Each of those literals creates a key-value pair with a constant value. Every transaction is an insert transaction causing World State change. The randomness of the key makes with negligible probability impossible to insert a key that was previously in the database, therefore transactions are protected against being invalid.

3.1. Research Environment

Experiments were conducted in the cloud on the Azure platform. A single virtual machine D64 v3 was used, with 64 vCPU, 256GB of RAM, and 1600GB storage. The running operating system was Ubuntu 18.04 LTS.

To analyze the performance of different Hyperledger platforms, a dedicated benchmark was used. Hyperledger Caliper [2] is a performance benchmark tool for multiple blockchain platforms that can cooperate with blockchain networks in two ways. First is connecting to an existing and working blockchain network, the second is starting a

blockchain network before the benchmark test using the networks defined in docker-compose.yaml files. Hyperledger Caliper produces .html reports containing several performance indicators, such as transactions per second, transaction latency, throughput, and resource utilization. Hyperledger Caliper architecture consists of four main layers: the benchmark layer, adapter layer, an interface layer, and blockchain framework layer. The adaptation layer is the main component of the caliper tool. It is used to integrate different blockchain network implementations into the benchmark. For every blockchain network, the adaptor implements the Caliper Blockchain NBIs(North Bound Interface) by using the corresponding blockchain's native SDK or RESTful API. Currently supported platforms are: Hyperledger Fabric v1.4+, Hyperledger Sawtooth 1.0+, Hyperledger Iroha 1.0 beta-3, and Hyperledger Burrow 1.0.

Benchmark layer contains the tests for typical scenarios, each test has a configuration file that defines test arguments and a backend blockchain network. Such configuration files define use cases that are then used by the underlying benchmark engine to perform the test. The role of this layer is to perform stress tests on the implemented blockchain platform. The blockchain framework layer contains different implementations of blockchain networks, which can be run separately from the benchmark. Caliper communicates with a particular network with the utilization of an appropriate adaptor.

3.2. Key Metrics Definitions

The first key metric is transaction latency, which is the time difference between transaction submission and transaction commit confirmation time across the network. It includes the propagation time and any setting time due to the consensus mechanism in place. This measure is computed per single transaction. The calculation of transaction latency differs according to the used protocol. This definition is appropriate for blockchain systems with deterministic transaction finality. Blockchain systems with a lottery-based consensus like Bitcoin, have probabilistic transaction finality, hence the latency calculation should follow other rules [17]. For measurements in this paper, the finality of the underlying systems under test is assumed. Second metric is transaction throughput, which is the rate at which the blockchain system under test commits valid transactions in the defined time frame. It is calculated as the number of transactions per second (tps). This rate refers to the entire blockchain network, rather than to a single node. Since only valid transactions are considered, the throughput consists of only positively verified transactions [21]. The transaction here refers to a commit transaction. This paper does not consider the throughput of query operations. Third metric is scalability, which is measured as the change in throughput and latency when increasing the number of nodes and the number of concurrent workloads. [6]

3.3. Test plan

There were four parameters for each blockchain platform test case: Transaction sending rate, block size, network size, and network traffic mode. A transaction sending rate is a number of transactions sent to the blockchain network in a defined time frame range. The transaction sending rate is presented as a transaction number per second. One test case consists of multiple rounds. In each round there is a constant transaction sending rate

being sent within 5 seconds. For example, for 50 Tps the total number of transactions sent in the round is 250 transactions. Each new round has a bigger transaction sending rate.

Block size is the maximum allowed transaction quantity which can be placed in a block being published to the blockchain network. Hyperledger Fabric and Hyperledger Iroha were tested with two different values of the block size(10 and 50). Hyperledger Sawtooth was tested only for a block size equal to 50.

Network size is the number of nodes participating in the blockchain network consensus mechanism. The smallest tested network consists of 5 nodes. The biggest tested network is for Hyperledger fabric and consists of 100 nodes.

The last parameter is the network traffic mode. Transactions can be distributed across all nodes in the network or can be sent to some particular nodes only. This parameter is introduced to check whether blockchain network performance will change when network traffic switches from distributed against all nodes to single node operating all transactions.

4. Results and discussion

This part contains the results of the tests for all parameters defined in the Test plan section. Test results are being shown in the form of figures and tables and the following examinations are being made:

- Impact of transaction sending rate to network latency.
- Impact of transaction sending rate to network throughput.
- Impact of block size to network latency.
- Impact of Block size to network throughput.
- Impact of network traffic mode to latency and throughput.
- Scalability, the impact of network size on latency.
- Scalability, the impact of network size on throughput.
- What are the reasons for failing transactions for some networks?

4.1. Hyperledger Fabric performance evaluation

This section contains the test results for Hyperledger Fabric v1.4. Fabric network consists of one organization owning all peers working on a single channel, therefore the network works as a private blockchain network. The network uses only one Solo ordering node. Endorsing peers are those who have smart contracts installed to be able to execute and endorse transactions. Due to technical reasons with running fabric network with Hyperledger Caliper maximum number of endorsing peers in all tests is 30, for network size exceeding 30 rest of peers works only as storage nodes. To analyze the impact of transaction sending rate on transaction latency, each fabric network was tested against different transaction sending rates. The whole network used a maximal block size of 10 transactions. Network traffic was distributed equally between all peers.

Figure 1 shows how the minimal transaction latency depends on transaction sending rates. To illustrate this relation three tested network sizes were chosen: 10, 50 and 100 peers. It is easy to notice that the network with 10 peers has smaller minimal transaction latency, but generally it is similar for all networks. Other observation is that the transaction sending rate does not have any impact on the minimal latency. For each fabric network size minimal transaction latency was bigger than 200ms.

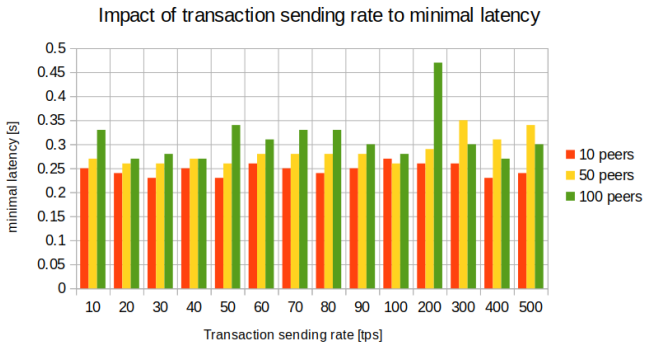

Fig. 1. Impact of Transaction sending rate to minimal latency for Hyperledger fabric platform

Figure 2 shows impact of transaction sending rate to average transaction latency in the network. The first observation is increasing the average transaction latency with increasing sending rate, it seems that at some point it grows faster, but this is because the scale on the x-axis has changed. It seems that the average latency depends linearly on the transaction sending rate. The second observation is that networks with fewer peers have smaller average latency than bigger networks at the same transaction sending rate. The third observation is that for networks with 30 peers and more there is no difference in average transaction latency. It might create an assumption that the performance of fabric networks depends only on the number of endorsed peers.

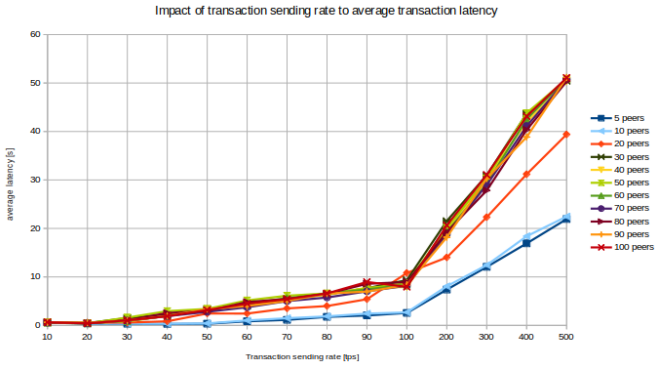

Fig. 2. Impact of transaction sending rate to average latency of Hyperledger fabric platform

Analysis of transaction sending rate impact to network throughput was performed just as its impact on latency. The maximum number of transactions in a single block is 10. Network traffic is distributed equally to all peers. The maximum number of endorsing peers in the network is 30.

The result shows how the network throughput depends on the transaction sending rate. Results show increasing network throughput with transaction sending rate to some point at which with larger transaction sending rate throughput is constant. To this point throughput is only a bit smaller than the transaction sending rate. The second observation is that smaller networks reach their saturation point later and achieve bigger network latency. As the maximum network endorsing peers number is limited to 30, we can see larger networks with no more endorsing peers have maximal throughput the same as the network with 30 peers only.

To analyze how block size impacts average latency fabric networks up to 100 peers were tested. Transactions were distributed equally between all peers. Tests were performed for two block size values: 10 and 50.

Figure 3 shows impact of block size to transaction average latency. Every network was tested against two-block size values. To detect the relations between maximal transactions in a block and transaction latency, network with 5, 20, and 40 peers were chosen to be depicted in the figure. Figure 3 shows there is no difference in transaction latency for different block sizes. It might suggest that the bottleneck of the transaction processing is not in the ordering service.

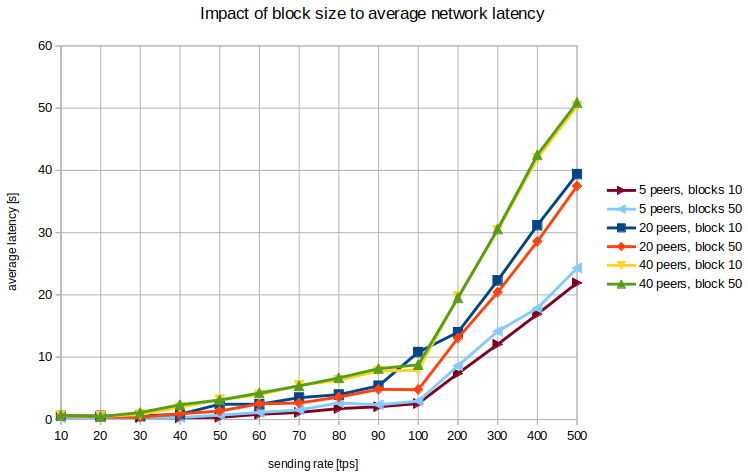

Fig. 3. Impact of block size to transaction latency for Hyperledger fabric platform

To analyze how block size impacts network throughput, fabric networks up to 100 peers were tested with a sending rate up to 500 transactions per second. Transactions were distributed equally between all peers. Tests were performed for two block size values: 10 and 50.

Results shows the impact of block size to network throughput. Every network was tested against two-block size values. To detect relations between network throughput and maximal transactions in a block fabric network with 5, 10, 20, and 40 peers were chosen to be depicted in the figure. Results shows that networks with 20 peers had bigger throughput

with larger block sizes while for 5 and 10 peers it was opposite, smaller block size caused better throughput. No strict correlation between block size and throughput were noticed.

To analyze the difference between traffic distribution modes to transaction latency and network throughput all network sizes were tested against with the transaction sending rate starting with 50 transactions per second and finishing with 500 transactions per second. The network block size was 50 for all networks. All transactions in the network are being handled by a single peer which is the only endorsing peer in the network.

Figure 4 shows the relation between transaction sending rate and average transaction latency for fabric networks up to 100 nodes with all network traffic sent to a single node. Average latency, as seen in the figure, is constant until a saturation point for a sending rate equal to 100 tps, after that the saturation point latency starts to grow linearly. In comparison with figure 2 from subsection "Analysis of transaction sending rate to transaction latency" it is easy to notice that for traffic divided between multiple nodes the latency was bigger.

Figure 5 shows the relation between transaction sending rate and network throughput for networks up to 100 nodes in which all transactions are sent to one peer. Network throughput, as seen in the figure, grows linearly with the transaction sending rate until it reaches a saturation point for the transaction sending rate between 100 and 200 tps. Regardless of the network size, all networks do not exceed the throughput of 150 tps in this examination.

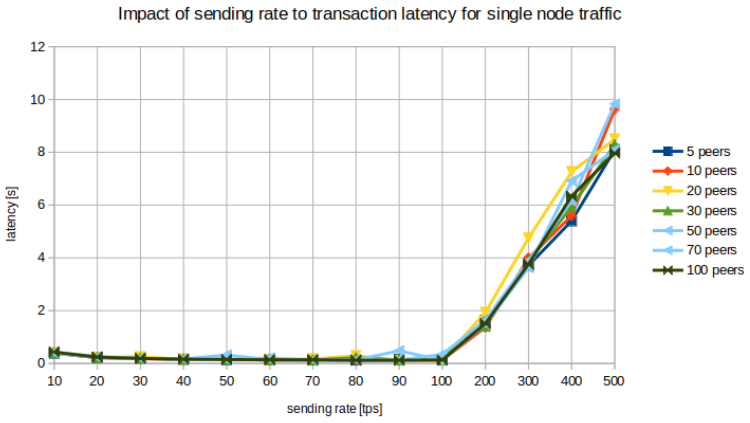

Fig. 4. Impact of transaction sending rate to transaction latency of Hyperledger Fabric platform for traffic handled by a single node

To analyze the relationship between network size and network average transaction latency fabric networks with peers quantity up to one hundred were tested. Each network was tested against a few sending transaction rates. All transactions sent to the network were distributed equally among all peers in the network. Maximal transactions in the block were equal to 10. The maximal number of endorsing peers was 30.

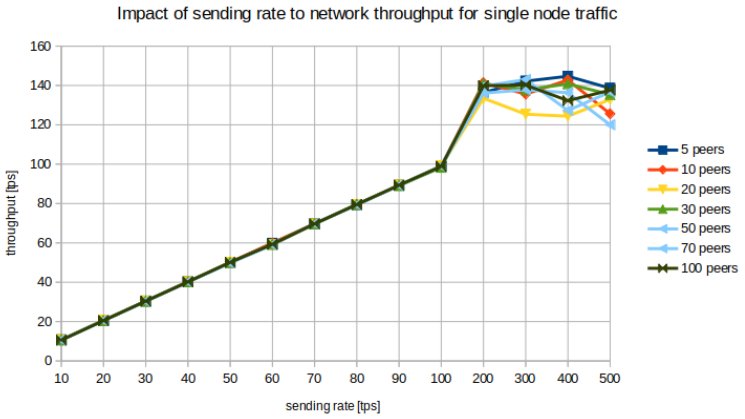

Fig. 5. Impact of transaction sending rate to network throughput for Hyperledger Fabric platform for traffic handled by a single node

Figure 6 shows impact of fabric network size to average transaction latency in the network. To show the relation between network size and its average transaction latency results are shown for different transaction sending rates: 50, 100, 300, and 500 transactions per second. From figure 6 it is easy to notice that the transaction latency grows with the network size up to the maximum number of endorsing peers in the network, which is 30. For more peers than 30 latency becomes a constant value, depending on sending transaction rate.

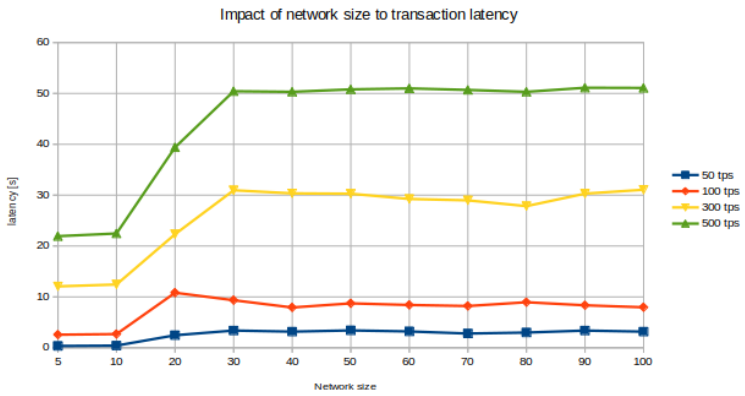

Fig. 6. Impact of network size to average transaction latency for Hyperledger Fabric

To analyze the relationship between network size and network throughput fabric networks with peers quantity up to one hundred were tested. Each network test consisted of sending transactions at a different rate. All transactions sent to the network were dis-

tributed evenly between all network nodes. Maximal transactions in the block were equal to 10. A maximal number of endorsing peers was 30.

Figure 7 shows impact of fabric network size to fabric network throughput. To show the relation between network size and network throughput each network was tested against different transaction sending rates: 50, 100, 300, and 500 transactions per second. Figure 7 shows that for bigger fabric network throughput is getting smaller. Peers who are not endorsing peers do not impact network throughput.

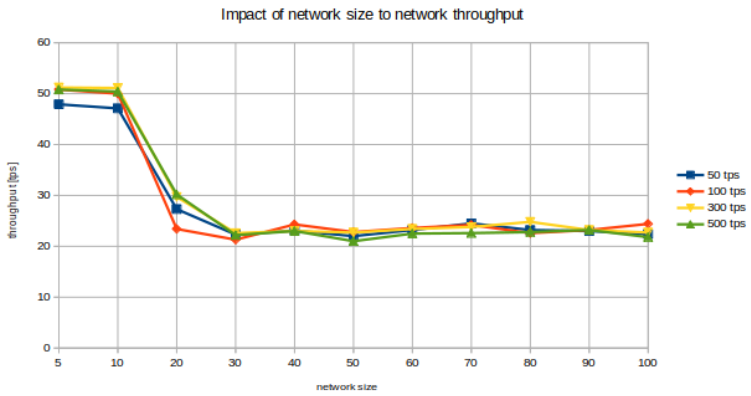

Fig. 7. Impact of network size to network throughput for Hyperledger Fabric

4.2. Hyperledger Iroha performance evaluation

This section contains test results and analysis of Hyperledger Iroha. This network was tested up to 50 nodes, as for more nodes the network did not work correctly. The network was tested for two maximal transactions in a single block value: 10 and 50. Maximal sending rate for which this platform was tested was 200 transactions per second.

Analysis of transaction sending rate to transaction latency To analyze the transaction sending rate impact to transaction latency each Iroha blockchain network was tested against different transaction sending rates. The whole network used a maximal block size of 50 transactions. Network traffic was distributed equally between all peers.

Figure 8 shows minimal transaction latency for different transaction sending rates. To illustrate these four network sizes were tested: 5, 10, 20, and 30 peers. Block size was equal to 50 transactions. Figure 8 shows that there is no strict correlation with transaction sending rate and minimal latency. For most transaction sending rates the minimal latency is around 4 seconds.

Results shows the average transaction latency for different transaction sending rates. To illustrate these five network sizes were tested consisting of 5, 10, 20, 30, and 40 peers. Block size was equal to 50 transactions. Result shows that generally the average transaction latency is constant and at the same value for all transaction sending rates, therefore

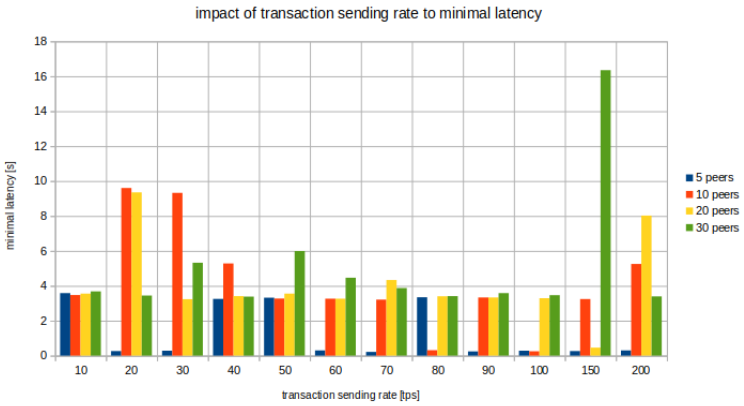

Fig. 8. Impact of transaction sending rate to minimal transaction latency for Hyperledger Iroha

does not depend on it. Network size seems to affect the value of average transaction latency, with larger networks having bigger average transaction latency.

Analysis of transaction sending rate impact to network throughput was preceded with testing Iroha networks up to 50 nodes against increasing the transaction rate up to 200 transactions per second. All networks used block size of 50 transactions and all sent transactions were distributed between all nodes in the network. Results shows the impact of sending transaction rate to Iroha network throughput for networks consisting of 5, 10, 20, 30, and 40 peers. The network of 40 peers has a constant throughput below 5 transactions per second for every transaction rate, but as seen in the previous section such a large network does not work correctly and most of the transactions fail. For smaller networks generally, the throughput is increasing with transaction sending rate, but all networks had throughput below 40 transactions per second, which is less than a maximal block size.

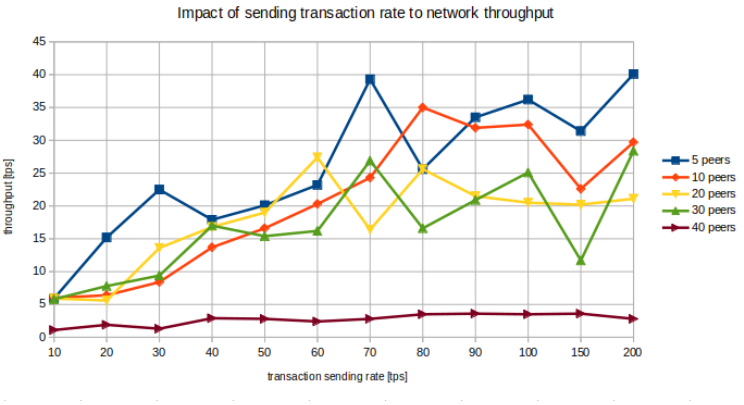

Fig. 9. Impact of transaction sending rate to network throughput for Hyperledger Iroha

Analysis of block size impact to average transaction latency was done for two block sizes: 10 and 50 transactions. The test was performed for three different network sizes: 5, 10, and 20 nodes. Each of the networks was tested against increasing the transaction sending rate up to 200 transactions per second. Transactions were distributed equally to all nodes in the networks.

Figure 10 shows impact of block size to average transaction latency in Iroha networks. To find the relation between block size and average transaction latency each network size had tested for different value of block size. After comparing appropriate pairs it is easy to see that for most transaction sending rates average latency was significantly lower for larger block size. Network with 20 nodes and block size equal to 50 has an average latency below 5 seconds for most transaction sending rates, while the same network with 10 transactions in a block can have an average latency around 43 seconds. This is 9 times bigger latency for 5 times smaller block.

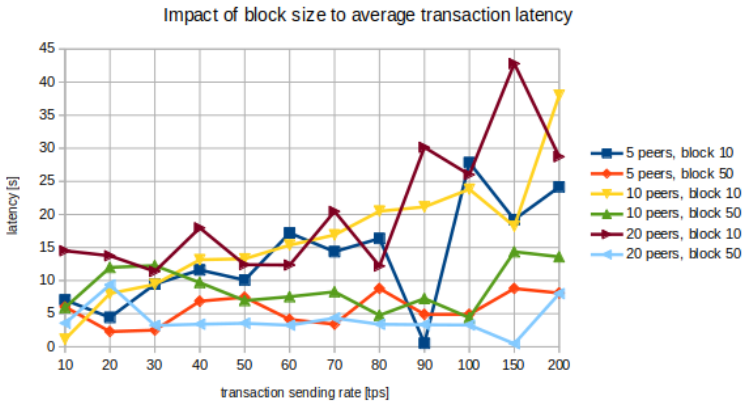

Fig. 10. Impact of block size to average transaction latency for Hyperledger Iroha

To analyze the impact of block size on network throughput tests with 10 and 50 maximal number of transactions in a block were performed. The test was performed for three different network sizes: 5, 10, and 20 nodes. Every network was tested against increasing the transaction sending rate up to 200 transactions per second. Transactions were distributed equally to all nodes in the networks.

Figure 11 shows block size impact to network throughput in Iroha networks. To find the relations between block size and network throughput networks must be compared in pairs: The same network sizes with different block sizes. For all networks depicted in figure 11 it is easy to notice that smaller block size causes the network throughput decrease. It is noticeable especially for bigger transaction sending rates.

This subsection describes the difference in the average transaction latency of Iroha networks for two cases:

- Network traffic distributed equally. Transactions are sent to every node in a network.
- Network traffic distributed to a single node that processes all transactions sent to the network.

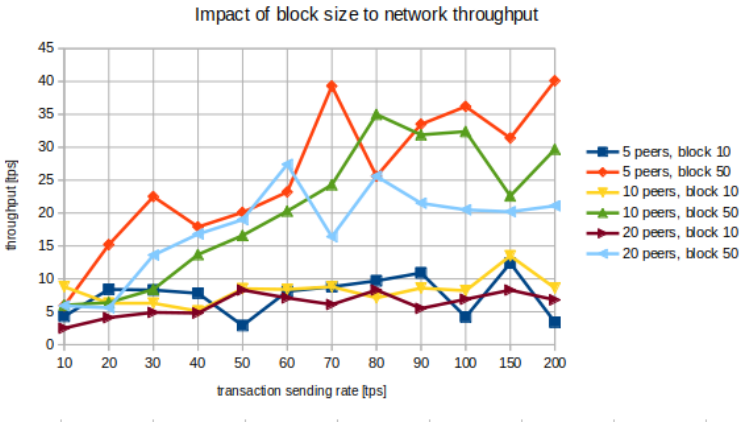

Fig. 11. Impact of block size to network throughput for Hyperledger Iroha

Figure 12 shows the impact of network traffic distribution to Iroha average transaction latency. All tests were performed by increasing transaction sending rate up to 200 transactions per second. The maximal transaction number in a block was 50. Each network size is tested with transactions sent to one node and distributed among all nodes in the network. For a network with 20 nodes traffic directing to a single node increases the average transaction latency for all transaction sending rates. For smaller networks such behavior is similar, but noticeable for higher transaction sending rates.

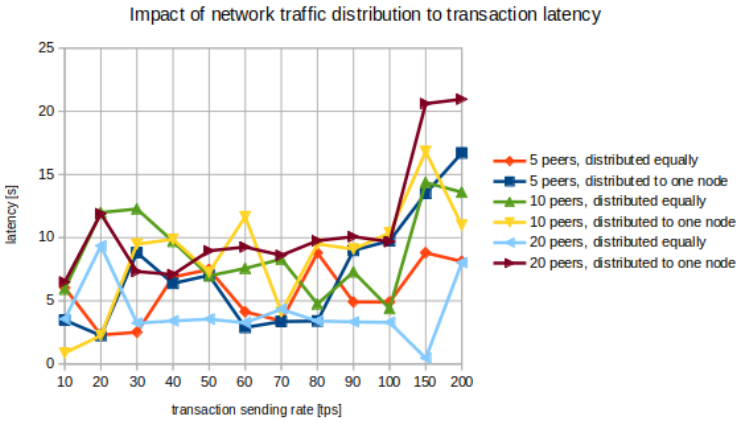

Fig. 12. Impact of network traffic distribution to average transaction latency for Hyperledger Iroha

Analysis of network traffic distribution to network throughput This subsection describes the difference in network throughput of Iroha networks for two cases:

– Network traffic distributed equally. Transactions are sent to every node in a network.
– Network traffic is distributed to a single node that processes all transactions sent to the network.

Figure 13 shows the impact of network traffic distribution on the Iroha network throughput. All tests were performed by increasing the transaction sending rate up to 200 transactions per second. Maximal block size is 50. Each network size is tested with transactions sent to one node and distributed among all nodes in the network. In figure 13 there is no strict correlation between the impact of transaction distribution to network throughput, but for transaction sending rate over 80 tps the throughput is higher for equal distribution of transactions between peers.

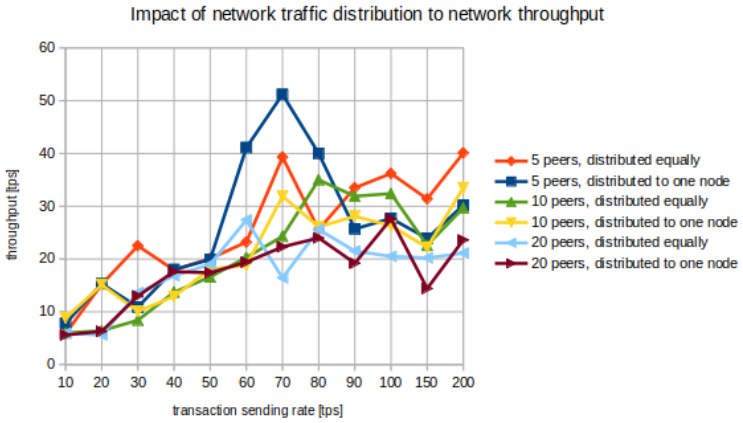

Fig. 13. Impact of network traffic distribution to network throughput for Hyperledger Iroha

Analysis of network size to transaction latency To analyze the relationship between network size and network average transaction latency of Iroha networks, networks with nodes number up to forty were tested. Each network was tested for increasing sending transaction rates up to 200 transactions per second. All transactions sent to the network were distributed equally against all peers in the network. Maximal transaction quantity in the block was equal to 50.

Figure 14 shows the impact of Iroha network size to the average transaction latency in the network. To show the relation between network size and its average transaction latency, results are shown for different transaction sending rates: 20, 60, 100 transactions per second. Figure 14 shows that average latency increases after network size exceeds 30 nodes. Between 10 and 30 nodes average transaction latency is generally constant and below 10 seconds. For a small Iroha network consisting of 5 nodes average transaction latency is the smallest.

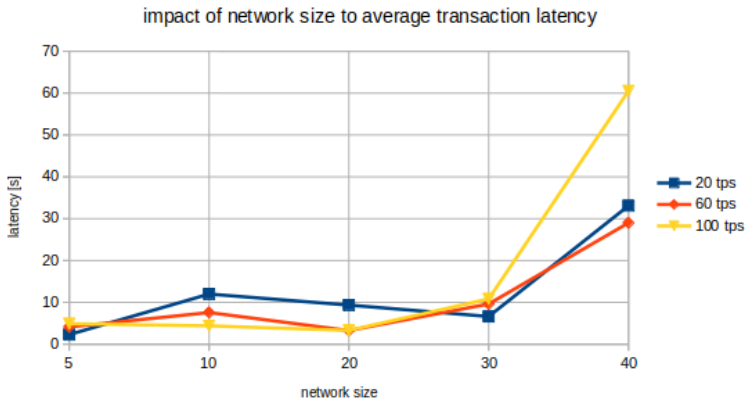

Fig. 14. Impact of network size to average transaction latency for Hyperledger Iroha

Analysis of network size to transaction throughput To analyze the relationship between network size and network throughput of Iroha networks, networks with nodes quantity up to forty were tested. Each network was tested for increasing sending transaction rates up to 200 transactions per second. All transactions sent to the network were distributed equally against all peers in the network. Maximal transaction quantity in the block was equal to 50.

Figure 15 shows that networks under small transactions sending rate(20 tps) have generally constant throughput when their size is between ten and thirty nodes. For high transaction sending rate(100 tps) throughput gets lower with network size increase. For network size exceeding 30 nodes throughput is very small(around 2-3 tps) regardless of transaction sending rate.

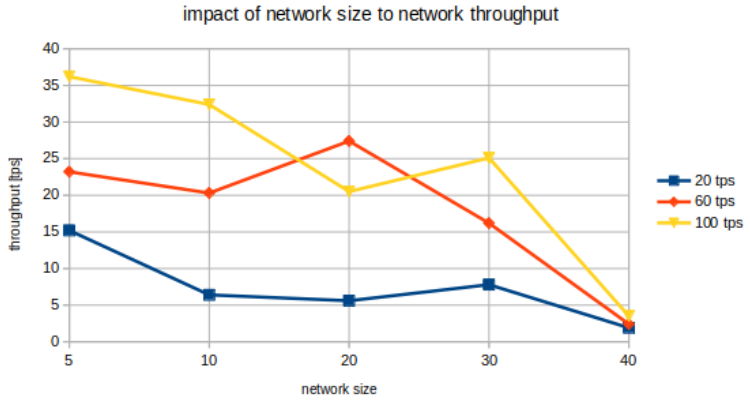

Fig. 15. Impact of network size to network throughput for Hyperledger Iroha

4.3. Hyperledger Sawtooth performance evaluation

This section contains test results and analysis of Hyperledger Sawtooth. Table 1 contains the parameters values for which benchmarking was done. This network was tested up to 30 nodes, as for more nodes network did not work at all. The network was tested for a block consisting of at most 50 transactions. The platform was tested for transaction sending rate increasing with every round starting with 10 and finishing with 100 transactions per second.

Table 1. Sawtooth performance evaluation results for network of 5 nodes and network traffic distributed among all nodes

sending rate[tps]	max latency[s]	min latency[s]	avg latency[s]	throughput[tps]	Failed [%]
10	4.81	0.41	2.5	7.3	0
20	5.22	0.42	2.99	13.9	0
30	2.82	0.41	1.74	24	0
40	2.81	0.42	2.34	27.4	0
50	6.23	1.61	3.91	26.5	0
60	8.26	0.61	4.9	25.1	0
70	9.02	0.62	7.37	2	67
80	NaN	NaN	NaN	NaN	100
90	6.23	4.62	5.35	1.4	67
100	8.02	5.42	6.7	2.4	49

Result shows that even small Sawtooth networks consisting of 5 peers only have a problem with successful transaction processing when network traffic is distributed among all participants. Networks with 20 nodes and more fail to process almost all transactions when the transaction sending rate exceeds 40 transactions per second. Similar results are in the scenario with a single node handling all transactions sent to network small networks (5 and 10 nodes) process almost all transactions successfully. For larger networks, transaction failures happens for smaller transaction sending rate. For example with a network consisting of 20 nodes shows that after rounds with 100% transaction failures(for 80 and 90 transactions per second) there is still a chance that in the next round network will process all transactions correctly.

There are multiple possible explanations why benchmarking of Hyperledger Sawtooth platform results in so many transactions failing including benchmark and platform problems:

- Hyperledger Caliper lets defining multiple nodes to which transactions are being sent, but only one validator which is used to check whether the transaction is added to the ledger. Due to the probabilistic implications of the PoET consensus algorithm, transactions can be included in other nodes, which are not yet in sync with the defined validator, therefore the transactions fail after some time.
- It was observed that right after sending transactions to a ledger there was an error message while checking the transaction state. There might be a bug in the Sawtooth adapter of the Hyperledger Caliper project. In the logs produced by peers, the network seemed to be in sync(as no attempts to synchronize with other peers were seen).

Maybe due to that error, Caliper is not able to verify the positive rest of the transactions and mark them invalid, although they are added to the ledger. In the next round of transaction sending, the rate network could add a new block.
- In some cases where transaction failures were 100% peer logs were constantly showing attempts to synchronize ledgers state for multiple peers. For some reason the network could not reach agreement on blocks ordering and remain unsynchronized, therefore the validator could not have added blocks in his private ledger. It might be a bug in fork resolver of Hyperledger Sawtooth v1.0.5 or message exchange protocol.
- Transaction processor responsible for processing transactions might be non-deterministic.
- Block size has a big impact on the performance for Hyperledger Iroha, bigger block size can decrease average latency and increase throughput.
- Analysis of network traffic distributions shows that it is better to equally distribute transactions in the Iroha network.

5. Conclusions

This paper shows the performance evaluation of three different consensus algorithms used in private blockchain networks and implemented in existing solutions. Each chosen consensus algorithm is based on a different principle. Hyperledger Iroha implements the YAC consensus, which is a voting-based Byzantine Fault Tolerant algorithm. Hyperledger Sawtooth implements the PoET algorithm which is proof-based consensus with probabilistic finality. Hyperledger Fabric v1+ abandons the standard state-machine replication protocol and introduces a new execute-order-validate architecture with Kafka Ordering service to improve performance of the platform compared with standard vote-based protocols.

Performance evaluation and scalability assessment were done by varying different sets of parameters such as block size, transaction sending rate, network traffic distribution, and network size. Performance evaluation was done based on average transaction latency, network throughput, and transaction failure rate. Scalability was assessed based on changes in transaction latency and throughput with increasing network size. Test results let to study the impact of a particular parameter on the blockchain network performance and show how they can be adjusted to improve performance. Performance measurements were gathered by Hyperledger Caliper, which is a dedicated benchmark platform for Hyperledger solutions. Hyperledger Caliper does not support the newest Hyperledger platform versions. The main conclusions are as follows: Hyperledger fabric was found to be the most deterministic regarding its performance results. Minimal transaction latency was in all cases between 0.2-0.4 seconds. The average latency of fabric platforms grows linearly with increasing sending rate. Throughput grows to a saturation point after which it is constant. Smaller fabric networks have smaller average latency and higher throughput than bigger networks with the same transaction sending rate. Tests show that adding endorsing peers to the channel decreases throughput and increases the average latency, while adding additional peers without an endorsing role does not have an impact on latency and throughput changes. Network size depends on the number of endorsing peers. Scalability analysis shows that adding new endorsing peers decreases the performance of the blockchain network. Test cases did not show any significant impact of block size on latency or throughput for Hyperledger fabric. The failing transaction might be caused by Caliper timeouts rather than network problems which can not commit the transaction fast

due to high overload. Minimal latency in Hyperledger Iroha according to experiments not depend on network size and transaction sending rate. For all sending rates it was around 4 seconds. Both the minimal and average latency is almost constant and does not depend on the transaction sending rate. Scalability analysis has shown that Hyperledger Iroha network performance worsens for networks which size exceeds thirty nodes. Hyperledger Sawtooth fails in many tests. Possible failures might be the result of: a bug in Caliper SawtoothAdapter used, a bug in PoET CFT implementation of Sawtooth v1.0.5, or bug in transaction processor handling 'Simple' transactions. Due to those failures further investigation must be done. Maximal used CPU power during the test was 15%, all network components were started and managed by a single docker container. It might suggest that the docker could not use all available power to properly manage nodes. Hyperledger Fabric is the most adult platform among all Hyperledger platforms and is used already in many production systems.

References

1. Bamakan, S.M.H., Motavali, A., Bondarti, A.B.: A survey of blockchain consensus algorithms performance evaluation criteria. Expert Systems with Applications 154, 113385 (2020)
2. Caliper, H.: Hyperledger caliper architecture. Electronic Article. url: https://hyperledger. github. io/caliper/docs/2_Architecture. html (visited on 03/10/2019) (2019)
3. Cruz, Z.B., Fernández-Alemán, J.L., Toval, A.: Security in cloud computing: A mapping study. Computer Science and Information Systems 12(1), 161–184 (2015)
4. De Angelis, S., Aniello, L., Lombardi, F., Margheri, A., Sassone, V.: Pbft vs proof-of-authority: applying the cap theorem to permissioned blockchain. In: Italian Conference on Cyber Security. p. 11 pp. (01 2017)
5. Dhillon, V., Metcalf, D., Hooper, M.: The hyperledger project. In: Blockchain enabled applications, pp. 139–149. Springer (2017)
6. Dinh, T.T.A., Wang, J., Chen, G., Liu, R., Ooi, B.C., Tan, K.L.: Blockbench: A framework for analyzing private blockchains. In: Proceedings of the 2017 ACM International Conference on Management of Data. pp. 1085–1100. ACM (2017)
7. Feng, L., Zhang, H., Chen, Y., Lou, L.: Scalable dynamic multi-agent practical byzantine fault-tolerant consensus in permissioned blockchain. Applied Sciences 8(10), 1919 (2018)
8. Gorenflo, C., Lee, S., Golab, L., Keshav, S.: Fastfabric: Scaling hyperledger fabric to 20,000 transactions per second. arXiv preprint arXiv:1901.00910 (2019)
9. Group, H.A.W., et al.: Hyperledger architecture volume 1: Introduction to hyperledger business blockchain design philosophy and consensus (2017)
10. Huang, K., Chen, Y., Jia, H., Lan, J., Yan, X., Wang, Z.: Fast multicast scheme with secure network coding in cloud data centers. Computer Science and Information Systems 13(2), 531–545 (2016)
11. Jovanović, B., Milenković, I., Bogićević-Sretenović, M., Simić, D.: Extending identity management system with multimodal biometric authentication. Computer Science and Information Systems 13(2), 313–334 (2016)
12. Kedziora, M., Kozlowski, P., Szczepanik, M., Jozwiak, P.: Analysis of blockchain selfish mining attacks. In: International Conference on Information Systems Architecture and Technology. pp. 231–240. Springer (2019)
13. Li, W., Sforzin, A., Fedorov, S., Karame, G.O.: Towards scalable and private industrial blockchains. In: Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts. pp. 9–14. ACM (2017)

14. Moser, M., Eyal, I., Sirer, E.G.: Bitcoin covenants. In: International Conference on Financial Cryptography and Data Security. pp. 126–141. Springer (2016)
15. Nasir, Q., Qasse, I.A., Abu Talib, M., Nassif, A.B.: Performance analysis of hyperledger fabric platforms. Security and Communication Networks 2018 (2018)
16. Nguyen, G.T., Kim, K.: A survey about consensus algorithms used in blockchain. Journal of Information processing systems Vol. 14, No. 1, pp. 101-128, Jan. 2018 (2018)
17. Performance, H., Group, S.: Hyperledger blockchain performance metrics, https://www.hyperledger.org/HLWhitepaperMetricsPDFV1.01.pdf
18. Rüsch, S.: High-performance consensus mechanisms for blockchains (2018)
19. Sousa, J., Bessani, A., Vukolic, M.: A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform. In: 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). pp. 51–58 (June 2018)
20. Sukhwani, H.: Performance modeling & analysis of hyperledger fabric (permissioned blockchain network). Duke University (2018)
21. Sukhwani, H., Martínez, J.M., Chang, X., Trivedi, K.S., Rindos, A.: Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric). In: 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS). pp. 253–255. IEEE (2017)
22. Sukhwani, H., Wang, N., Trivedi, K.S., Rindos, A.: Performance modeling of hyperledger fabric (permissioned blockchain network). In: 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA). pp. 1–8. IEEE (2018)
23. Thakkar, P., Nathan, S., Viswanathan, B.: Performance benchmarking and optimizing hyperledger fabric blockchain platform. In: 2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS). pp. 264–276. IEEE (2018)
24. Turki, H., Salgado, F., Camacho, J.M.: Honeyledgerbft: Enabling byzantine fault tolerance for the hyperledger platform
25. Vukolić, M.: The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In: Camenisch, J., Kesdoğan, D. (eds.) Open Problems in Network Security. pp. 112–125. Springer International Publishing, Cham (2016)
26. Wang, R., Ye, K., Meng, T., Xu, C.Z.: Performance evaluation on blockchain systems: A case study on ethereum, fabric, sawtooth and fisco-bcos. In: International Conference on Services Computing. pp. 120–134. Springer (2020)

Arnold Woznica received his M.Sc. degree in Computer Science from University of Science and Technology, Wroclaw, Poland in 2019. His research area of interests encompass blockchain systems and software engineering.

Michal Kedziora received the Ph.D. degree in Computer Science from the Wroclaw University of Science and Technology, Wroclaw, Poland, in August 2014, and the M.S. and Engineer degree in Computer Security from the University of Technology in Wroclaw, Poland, in December 2006. In 2017 he finished postdoc at University of Wollongong, Australia. He was working as a Visiting Researcher at University of Technology Sydney, Australia (2019) and Embry-Riddle Aeronautical University, Daytona Beach, FL, USA (2020).

Received: May 07, 2021; Accepted: January 25, 2022.