# Consortium Blockchains: Overview, Applications and Challenges

Omar Dib, Kei-Leo Brousmiche, Antoine Durand, Eric Thea, Elyes Ben Hamida
IRT SystemX, Paris-Saclay, France
Email: {first.lastname}@irt-systemx.fr

***Abstract*—The Blockchain technology has recently attracted increasing interests worldwide because of its potential to disrupt existing businesses and to revolutionize the way applications will be built, operated, consumed and marketed in the near future. While this technology was initially designed as an immutable and distributed ledger for preventing the double spending of cryptocurrencies, it is now foreseen as the core backbone of enterprises by enabling the interoperability and collaboration between organizations. In this context, consortium blockchains emerged as an interesting architecture concept that benefits from the transactions' efficiency and privacy of private blockchains, while leveraging the decentralized governance of public blockchains. Although many studies have been made on the blockchain technology in general, the concept of consortium blockchains has been very little addressed in the literature. To bridge this gap, this article provides a detailed analysis of consortium blockchains, in terms of architectures, technological components and applications. In particular, the underlying consensus algorithms are analyzed in details, and a general taxonomy is discussed. Then, a practical case study that focuses on the consortium blockchain technology Ethermint is performed in order to highlight its main advantages and limitations. Finally, various research challenges and opportunities are discussed.**

***Keywords–Consortium Blockchain; Distributed Ledger Technology; Smart Contract; Consensus Algorithm; Data Privacy and Security; Scalability; Benchmark.***

## I. INTRODUCTION

The blockchain technology has attracted increased interests worldwide in recent years, with emerging applications in key domains, including finance, energy, insurance, logistics and mobility. Indeed, this technology is expected to disrupt businesses and markets on a global scale.

A blockchain is essentially a trustless, peer-to-peer and continuously growing database (or ledger) of records, *aka.* blocks, that have been verified and shared among the participating entities [1]. Each block typically contains a timestamp, a cryptographic hash value of the previous block, and a set of transactions data. Once a new block is validated by consensus and written to the ledger, transactions cannot be altered retroactively without the collusion of the network majority. This technology was initially designed as a public transaction ledger to solve the double spending problem in the Bitcoin digital currency system [2], without the need for any third party or trust authority.

This technology per se is not novel, but is rather a combination of well-known building blocks, including peer-to-peer protocols, cryptographic primitives, distributed consensus algorithms and economic incentives mechanisms. A blockchain is more a paradigm shift in the way applications and solutions will be built, deployed, operated, consumed and marketed in the near future, than just a technology. Blockchain is secure by design and relies on well-known cryptographic tools and distributed consensus mechanisms to provide key characteristics, such as persistence, anonymity, fault-tolerance, auditability and resilience.

More recently, smart contracts [3] have emerged as a new usage for blockchains to digitize and automate the execution of business workflows (*i.e.,* self-executing contracts or agreements), and whose proper execution is enforced by the consensus mechanism. This makes the blockchain technology particularly suitable for the management of medical records [4], notary services [5], users' identities [6] and reputations [7], data traceability [8], vehicles' history [9], *etc.*

In this context, the blockchain technology is foreseen as the core backbone of future smart cities and enterprises by enabling the interoperability and collaboration between organizations, while enhancing their security, data and process management. Although many studies have been made on the blockchain technology in general [10], the application of this technology to business environments, beyond digital currencies, has been very little addressed in the literature. Indeed, several challenges will need to be addressed to unlock the tremendous potential of blockchains especially before this paradigm shift becomes technically, economically and legally viable in enterprises environments.

These challenges concern the technical aspects of blockchains, including its architecture, deployment, governance, scalability and data privacy. Private and consortium blockchains emerged from this perspective as appropriate architecture concepts for business environments, where restrictions are applied on who is allowed to participate to the network. While the former approach assumes that the network is operated by a single entity, a consortium blockchain operates under the leadership of a group of entities, thus enabling collaborative business transformation among organizations and innovative business models. Last but not least, the legal aspects of blockchains represent a challenge, especially in Europe, where this technology should be analyzed in the light of upcoming new regulations, such as the *General Data Protection Regulation* (GDPR) (Regulation (EU) 2016/679 [11]), and whose objective is to strengthen users' data privacy and protection within the European Union.

This article aims at bridging the gap between blockchain technologies and their potential benefits to business environments, by providing a detailed analysis of consortium blockchains, in terms of architectures, technologies and applications. In particular, the underlying consensus algorithms are analyzed in details, and a general taxonomy is discussed. Then, a practical and experimental use case is discussed in order to assess the performance of the consortium blockchain technology Ethermint. The performance indicators analyzed are the time required to validate a transaction, the number of transactions validated per second, as well as the average inter

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

blocks delay and the size of the blockchain data folder. Parameters that have been varied are the number of validators, the number of transactions submitted per second and the topology of the network. The results highlighted some limitations in terms of transactions validation time and storage requirements that may hinder the usage of Ethermint to deal with some real world use cases.

The remainder of this article is organized as follows. Section II discusses the technical aspects of blockchains in terms of taxonomy, system architecture, and building blocks. Section III provides a detailed analysis of consensus algorithms that are used today in private and consortium blockchains. Section IV discusses a general classification of blockchains applications and highlights typical use cases in the finance, energy, mobility and logistics sectors. Section V presents a practical case study based on the Ethermint consortium blockchain technology in order to highlight its main advantages and limitations. Section VI draws and discusses various research challenges and opportunities. Finally, Section VII concludes the article.

## II. Blockchain Technology Overview

While public blockchains enable parties to make transactions in a secured manner, in trust-less environments, they show certain limitations when applied to industrial use cases. Indeed, we believe that aspects, such as controlled data reversibility (**i**), data privacy (**ii**), transactions volume scalability (**iii**), system responsiveness (**iv**) and ease of protocol updatability (**v**) that are crucial for the majority of corporate applications are not covered by public blockchain implementations. These shortcomings led industrials to develop alternative blockchain technologies tackling the aforementioned aspects and intended for restricted audience. These technologies can generally be classified into two categories: private and consortium blockchains [12]. The distinction between them comes down to the governance scheme along with the infrastructure (see Table I). In private blockchains, one entity rules the whole system whereas members of consortium blockchains share the authority among them. Accordingly, the infrastructure is centralized in case of private blockchains but, similarly to common distributed databases, data is replicated on multiple nodes that belong to the single owner. In contrast, consortium blockchains are deployed in a decentralized manner on multiple hardwares managed by different owners (or companies). Moreover, data is not necessary homogeneous among consortium nodes since some blockchains allow private transactions leading to knowledge fragmentation (*i.e.,* private transactions are shared by subsets of participants).

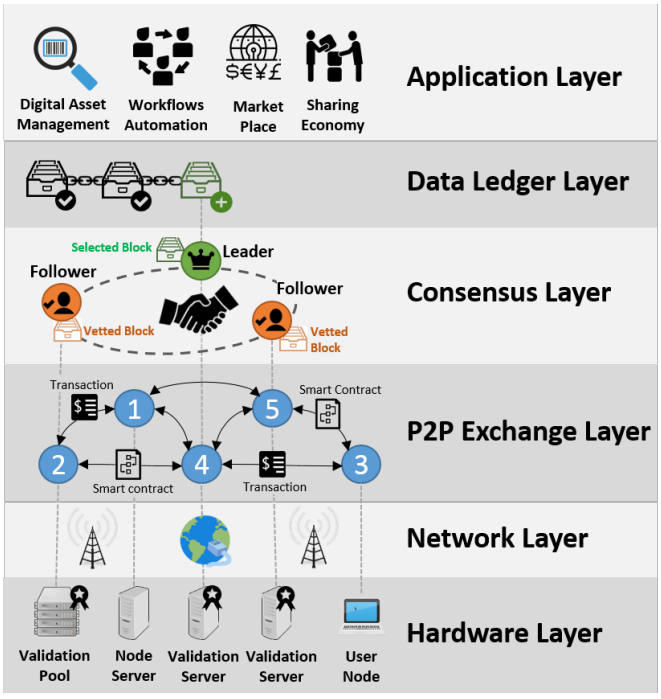


Figure 1. Blockchain High Level Architecture

Figure 1 shows the typical high level architecture of blockchain and its main layers. However, it should be noted that blockchain architecture is not yet standardized, and other representations exist in the literature, such as [13].

In the following, the term consortium blockchains will encompass both private and consortium blockchains. In the next sections, we overview the architecture of both public and consortium blockchain and highlight their inherent differences.

### A. Data structure (Data Ledger Layer)

The data structure of a blockchain, whether public or consortium, corresponds to a linked list of blocks containing transactions also referred to as the "ledger". Each element of the list, has a pointer to the previous block and embodies its hash value as illustrated in Figure 2.

This hash value is the key element of the blockchain integrity, and is computed thanks to a one-way hash function that maps data of arbitrary size to a non-invertible hash value of fixed size. Indeed, if an adversary tries to modify the content of a block, anyone can detect it by computing its hash and comparing it to the one stored in the next block to see the inconsistency. In order to avoid this detection, the adversary could try to change all the hashes from the tampered block to the latest block. However, this is not feasible without the consent of a significant proportion of validator nodes, depending on the consensus algorithm (see Section III). Hence, the "immutability" characteristic of blockchain: data are tamper-proof.

On the other hand, consortium chains members can come to an agreement and alter previous blocks (**i**). In order to prove that data were not tampered and preserve the auditability of the ledger, it is common to periodically publish the hash of a block onto a public blockchain. By doing so, one can be assured that blocks in the interval of two published hashes have not been modified.

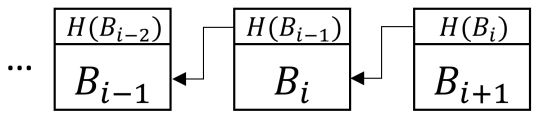


Figure 2. List of linked blocks with hash pointers

### B. Network and privacy (Network/P2P Exchange Layer)

Along with its data structure, a blockchain is based on a peer-to-peer network that links its members. Members participate to the network through their blockchain client node. Each

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE I. Blockchain Classification

| Property | Blockchain Governance | | |
|---|---|---|---|
| | Public | Consortium | Private |
| Governance Type | Consensus is public | Consensus is managed by a set of participants | Consensus is managed by a single owner |
| Transactions Validation | Anynode (or miner) | A list of authorized nodes (or validators) | |
| Consensus Algorithm | Without permission<br>(PoW, PoS, PoET, etc.) | With permission<br>(PBFT, Tendermint, PoA, etc.) | |
| Transactions Reading | Any node | Any node (without permission) or<br>A list of predefined nodes (with permission) | |
| Data Immutability | Yes, blockchain rollback is almost impossible | Yes, but blockchain rollback is possible | |
| Transactions Throughput | Low (a few dozen of transactions validated per second) | High (a few hundred/thousand transactions validated per second) | |
| Network scalability | High | Low to medium (a few dozen/hundred of nodes) | |
| Infrastructure | Highly-Decentralized | Decentralized | Distributed |
| Features | Censorship resistance<br>Unregulated and cross-borders<br>Support of native assets<br>Anonymous identities<br>Scalable network architecture | Applicable to highly regulated business (known identities, legal standards, etc.)<br>Efficient transactions throughput<br>Transactions without fees<br>Infrastructure rules are easier to manage<br>Better protection against external disturbances | |
| Examples of technologies | Bitcoin, Ethereum, Ripple, etc. | MultiChain, Quorum, HyperLedger, Ethermint, Tendermint, etc. | |

node has a local copy of the whole linked list (or the most recent part of it in case of *light nodes* [14]). When retrieving the list for the first time, a node verifies the integrity of the blocks by computing all the hashes and keeps verifying each new block.

Depending on the implementation of the blockchain, the network can be either public (*i.e.,* anyone can access it) or permissioned (*i.e.,* only accounts that are allowed can participate). This restricted access to the network in consortium blockchains ensures data privacy (**ii**). Moreover, some blockchains allows to control data visibility at a more finer grain by enabling data encryption at transaction level (*e.g.,* [15]).

The identity of a participant is defined by his cryptographic asymmetrical key pair. The public key is derived to obtain his unique address, which serves as his public identity. The private key is used to sign transactions and guarantee their authenticity (*i.e.,* other participants can verify the signature using the associated public key).

In order to add data to the blockchain, a node sends a transaction request to the network. The prime data fields of a transaction in most technologies are the addresses of both sender and receiver, data values that are being communicated and the signature of the sender. These transactions' requests are then picked by some special nodes called *miners* also referred as to block generators or validators on consortium blockchains.

### C. Security vs scalability (Consensus Layer)

On public blockchains, miners are nodes that are willing to share their computational power to add blocks to the blockchain in return of some reward. The way they are rewarded depends on the implementation of the blockchain protocol, however, it often involves a fee (*i.e.,* the node who was asking to add data pays the miner to do it) and/or creation of value (*e.g.,* in case of Bitcoin, the blockchain mints value and gives it to the miner who has added a block). Thus, miners are in competition: they all want to add the next block but only one or few of them will achieve it in a random way for each new block.

The process for selecting the actual node that will add the next block among all the miners is referred as a consensus protocol. For instance, Bitcoin and Ethereum use the Proof of Work (PoW) consensus [2], where miners have solve a computationally demanding cryptographic puzzle to prove their commitment. This protocol enables the random selection of the next miner and prevent adversary nodes from adding fraudulent blocks since their probability to be retained is too low compared to the procured energy and time to solve the puzzle. This leads to the 51% attack: if more than half of the power of the network is allied with the adversary, then his version of the ledger will always end up being the main one.

In a trust-less public configuration where miners are anonymous, this consensus is crucial for the integrity and security of the data. On the other hand, in consortium chains, validators are preliminary known according to the governance scheme and are trusted to some degree. Indeed, in majority of consortium protocols, validators are defined at the genesis of the blockchain. Some technologies enable the dynamic addition and retrieval of validators but these actions are always under the control of the current validators. Therefore, the security and required computational power can be lowered by using less demanding consensus algorithms. This reduction of complexity in the consensus protocol leads directly to an increased scalability in terms of transactions throughput (**iii**). Indeed, consensus such as PoW delay data transmission, *e.g.,* 10 minutes for each 1Mb block for Bitcoin, whereas protocols for consortium blockchains can make optimal use of the network and reach throughput on the order of thousands transactions per second. An overview of the major consensus algorithm for consortium blockchains is proposed in Section III.

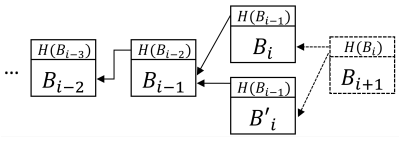


Figure 3. Fork

### D. Forks and responsiveness

Once a miner's block has been published, it is added to the blockchain and the information is broadcast. Due to network effects, there are cases where multiple miners blocks are published at the same time. It is also possible that some miners propose other block versions when they do not agree

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

with the content of the most recent one. These cases are called a *fork:* the blockchain splits into multiple branches. However, the nodes goal is to converge towards acknowledging a unique and same version of the blockchain. In practice, the Proof of Work consensus achieves this result by requiring miners to work on the longest branch, with the intuition that eventually the longest branch will be the same for all nodes.

Thus, even if a transaction has been validated, we cannot be sure that it will remain on the main chain. In Bitcoin, users usually wait six blocks of confirmation before considering a transaction (*i.e.,* its block) as valid, that being $6 \times 10min = 1h$. In that sense, transactions are never definitely accepted, there is only the probability of reversal that decrease exponentially as the chain grows.

Hence, there is a correlation between the probability of fork occurrence and the *responsiveness* of the blockchain (*i.e.,* time to wait for a transaction to be validated and adopted by the rest of the network). On consortium blockchains, the use of adapted consensus algorithm allows for "block finality": once a block has been validated, it remains on the main chain and forks are not allowed. This increases the system responsiveness by shrinking waiting time for confirmations (**iv**).

### E. Forks and updates

Miners software is sometimes updated to fix bugs or add functionalities. This also can create forks, as different nodes might handle transactions differently depending on their software versions. We usually distinguish:
- "soft forks" where the transactions considered valid by the new version are also valid for the old version.
- "hard forks" where the transactions considered invalid by the old version might be valid for the new version.

While it is complicated to synchronize the software of public blockchains due to the huge amount of anonymous participants and potential disagreements among them, it is easily feasible on consortium chains where members know each other and can quickly come to a mutual agreement (**v**).

### F. Smart Contracts

The concept of smart contract has been introduced by Ethereum and consists in computer programs, which executions results are verified by miners. Their deployments and executions are triggered by users through transactions. Each node participating to the blockchain has a local virtual machine (*e.g.,* EVM in Ethereum [16]) in addition to the linked list and executes smart contracts, according to the transactions that have been validated and maintain a globally shared state. By inheriting the security level offered by transactions, blockchain's smart contracts protocol benefit from the properties of the blockchain: security, integrity, no intermediary, transparency and availability. Some consortium blockchains offer the possibility to restrict the visibility of these contracts to a subset of members in the same way as private transactions (*e.g.,* [17]).

This overview of blockchain architecture and components highlighted the main differences between public and permissioned implementations. In the next section, we present major consensus algorithms used in consortium blockchains that enables high transaction throughput compared to the regular algorithm such as the Proof of Work or the Proof of Stake.

## III. Consensus Algorithms

As we saw in the previous section, the process that adds blocks to the linked list is a major factor to security and scalability. This is done with a *State Machine Replication* (SMR) algorithm, which makes the network agrees on a unique, constantly growing, ordered set of transactions. The *consensus* algorithm, is the part that makes the nodes agree on a single piece of data [26], *i.e.,* the block that is going to be added. However, since they are both very closely related, the terms are often used interchangeably.

To solve consensus, an algorithm has to provide two properties: Safety and Liveness. Informally, Safety means that once nodes confirmed a new entry to the transaction log, it will not be changed later. Liveness means that if a honest user has sufficient connectivity and tries to submit a transaction, then it will eventually get accepted.

The reason why blockchains are deemed to be *trustless* is because those algorithms are *Byzantine Fault Tolerant* (BFT), that is, other nodes may behave arbitrarily, including in malicious ways. Therefore, blockchains users do not need to trust others. However, it has been proved that no consensus algorithm can tolerate more than a third of the nodes being malicious [27]. Note that blockchains may tolerate more byzantine faults, but will not satisfy the traditional definition of consensus. This is most often the case with public blockchains that do not have transaction finality (or block finality, as mentioned earlier), *i.e.,* transactions are guaranteed to be irreversible only with a sufficiently high probability.

### A. Algorithms

In this section, we describe prominent consensus algorithms that are considered today in consortium blockchains where miners are designated nodes (*i.e.,* validators). In Table II, we synthesize their characteristics: *Fault tolerance* indicates whether the algorithm tolerates arbitrary (*e.g.,* malicious) behavior, or only crash failure. The *Threshold* further quantifies the amount of faults tolerated. *Confirmation time*, also known as latency, is the time elapsed between the submission of a transaction, and its definitive acceptance. *Throughput* is the number of transactions per second that the system can sustain as a whole, and *Scalability* is the number of nodes that can participate in the consensus. We warn the reader that the figures for the last three columns are not based on rigorous experimental studies, but they were collected from online blogs and websites. Indeed, these performances are not expected to be reachable simultaneously for the three metrics, and depends largely on the algorithm configuration and network settings.

*a) Practical Byzantine Fault Tolerance (PBFT):* PBFT is the first high performance consensus algorithm with an optimal byzantine fault tolerance [28]. As such, it has been widely implemented and tested, and served as a basis for a wide range of newer algorithms.

This method works with a leader that does the ordering of transactions. Then consensus is reached in three phases: First, the leader broadcasts the new request, *e.g.,* transactions or block (pre-prepare). Then, each validator signs and broadcasts a *prepare* message for the request. If enough prepare messages have been received, then validators broadcast *commit* messages, and when enough commits have been received, the request is finally accepted.

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE II. BENCHMARKING OF CONSENSUS ALGORITHMS

| Consensus algorithm | Fault Tolerance | Threshold | Confirmation time | Throughput | scalability |
|---|---|---|---|---|---|
| Tendermint Core | Byzantine | 33% | 5s [18] | 10k tx/s [18] | 100 nodes [18] |
| PBFT | Byzantine | 33% | 1s [19] | 50k tx/s [19] | 30 nodes [19] |
| Hashgraph | Byzantine | 33% | n/a (claimed 1s) [20] | n/a (claimed 100k tx/s) [20] | n/a |
| SCP | Byzantine | partitioning | up to 15s [21] | 1-10k tx/s [22] | - |
| PoET | Byzantine | TEE failure | n/a | n/a | very high [23] |
| DiversityMining | Crash | tunable, $\leq$ 50% | n/a | 1k tx/s [24] | n/a |
| Raft | Crash | 50% | 1s [25] | up to 30k tx/s [25] | n/a |

If at some point the leader does not behave as expected (*e.g.,* if there was no answer after some timeout), the next leader in the round robin order takes over through a complex procedure named *view change*. Informally, view change can be seen as a weaker variant of consensus, because nodes need to reach agreement on the fact that the leader changed.

This algorithm is able to tolerate a maximum of a third of the nodes being malicious, which is the maximum possible in this setting. It also makes the assumption that the network is partially synchronized since nodes need to be able to skip faulty leaders, *i.e.,* after a timeout. A PBFT implementation is currently in development for Hyperledger Fabric. Moreover, IstanbulBFT has been developed as a blockchain-friendly variant of this algorithm, and is implemented in Quorum [29].

*b) Tendermint:* Tendermint is a consensus algorithm that has been designed with applications for consortium blockchain in mind [30]. It is somewhat similar to PBFT, but additionally provide safety even when more that one third of the nodes get corrupted [31]. For each new block, a leader node is selected in a round-robin manner until one is able to commit the block. For a block to be committed, it has to gather more than two third of votes of validators within a given time period, through a procedure similar to the three-phases commit from PBFT.

Tendermint-core is provided as a stand-alone consensus engine, and has been implemented to function with the Ethereum Virtual Machine (EVM) in Hyperledger Burrow and Ethermint. Other blockchain frameworks based on Tendermint can be found on [32].

*c) Proof of Elapsed Time:* Proof of elapsed time is an initiative that aims at removing the computational cost induced by the usual proof of work approach, by leveraging Trusted Execution Environment (TEE) compatible hardware, such as Software Guard Extensions (Intel SGX).

For each block, miners wait for a given random time. The first miner for which the waiting time has elapsed is selected to validate a block before repeating this process. In other words, the miner with the shortest waiting time is elected as the leader. To prove that the node actually waited the required time, the waiting procedure is executed within the TEE, which produce a proof of its execution [33]. In essence, this is the same mechanism as in PoW-based consensus, except that the cryptographic puzzle is replaced by a hardware-enforced wait procedure. Note that the trusted hardware can also enforce that all nodes follow the protocol in its entirety (*i.e.,* are honest), thus making any crash-tolerant protocol withstand malicious faults as well. However, the precise construct of PoET allows to keep the advantages of PoW-based consensus, namely, the great scalability in the number of nodes. This does come with the same performance cost than public blockchains, since the waiting delay must be large enough to have a low probability of a (network-induced) fork.

The downside is that the security is guaranteed only if the TEE platform vendor is trusted. Moreover, if one is willing to assume that nodes may still be compromised (*e.g.,* due to implementation bugs), then only a few number of them would be sufficient to compromise the whole system [23]. PoET is primarily used within the Hyperledger Sawtooth platform [34].

*d) Stellar Consensus Protocol (SCP):* The Stellar Consensus Protocol [35] is the algorithm that was developed to power the Stellar network. It breaks the usual prerequisite of a unanimously accepted membership list by letting participants independently choose the nodes they trust. Each participant knows some nodes that are considered as trustful, *i.e.,* its *neighborhood*, and waits that the majority of them agree on a new transaction before considering it as valid. Consensus within a neighborhood is reached by first iteratively nominating candidate values, then by committing one. Those procedure are based around a primitive that allows nodes to ratify statements, which is done with two round of voting.

This approach allows each node to only be aware of a limited number of neighbors, while enabling an efficient network-wide consensus. On the other hand, security relies on each user good configuration, and requires that each neighborhood sufficiently intersects with each other. Additionally SCP may be less suited to consortium blockchains, due to its federated architecture, where one neighbor alone will not be trusted. Therefore, to actually benefit from SCP, a given blockchain platform should either anchors to the Stellar network, or try to spawn a new independent network.

*e) Hashgraph:* Hashgraph is an asynchronous protocol, which means that it makes no assumptions on the network delays [36]. This allows the nodes to be connected through an unreliable network such as the Internet. Its core idea is to piggyback the underlying broadcast protocol (*i.e.,* gossip) to reach consensus. By explicitly recording the network-layer execution of the broadcast into a graph similar to a blockchain, it is able to track events that has been sufficiently spread in the network to reach consensus. The Hashgraph authors show that if nodes are constantly synchronizing, then a given event will end up sufficiently deep in the graph so that one can be assured that a majority of nodes are able to tell that this event is valid.

Swirlds [37], the company that developed this algorithm, made available an implementation through a closed source Java SDK in alpha version. However, at the time of writing there was no independent benchmark of Hashgraph, which makes assessing its performances difficult.

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE III. BENCHMARKING OF ENTERPRISE BLOCKCHAIN PLATFORMS AND TECHNOLOGIES.

| Company | Platform | Consensus Algorithm | Smart Contracts | Private Transactions | Popularity ♠ | Activity (Github) ♠ |
|---|---|---|---|---|---|---|
| Coin Sciences Ltd | MultiChain | DiversityMining | No | No | Low | Medium |
| Quorum | Quorum | pluggable: IstanbulBFT ①, Raft | EVM | Yes | High | Medium |
| IBM | Hyperledger Fabric | pluggable ②: Kafka | Chaincode | Yes ③ | High | High |
| R3 | Corda | pluggable: Raft, BFT-SMaRt | JVM | Yes | Medium | High |
| SWIRLDS | Hashgraph | Hashgraph | No ④ | No | Low | N/A |
| Stellar | Stellar | SCP | No | No | High | Medium |
| ParityTech | Parity | pluggable: Ethereum, Aura ⑤, Tendermint | EVM | No | High | High |
| Intel | Hyperledger Sawtooth | PoET | EVM | No | Low | High |
| Monax, Intel | Hyperledger Burrow | Tendermint core | EVM | No | Medium | Medium |
| All In Bits Inc. | Ethermint | Tendermint core | EVM | No | Medium | Medium |

♠ Rough estimation based on GitHub metrics and online presence (*e.g.,* download count, community hubs activity, *etc.*)
① IstanbulBFT is an adaption of PBFT for blockchains
② Other consensus to be added, or unofficial: PBFT, BFT-SMaRT, HoneyBadgerBFT
③ Stand alone private transactions are not possible, but participants can set up private channels
④ Hashgraph let the transactions semantics be implemented by the application, but they will not be checked during consensus
⑤ Aura is a simple consensus engine developed by ParityTech, but it is not well specified and there is no assessment of its soundness

*f) Diversity Mining consensus:* DiversityMining is a consensus algorithm developed by MultiChain [38]. It is based on leader election, but where PBFT uses timeouts to handle leader failure, DiversityMining allows a fraction (subject to a parameter named *diversity*) of them to claim leadership independently, directly by broadcasting a block. Thus, the amount of tolerable faults is directly set by this parameter. Forks resulting from this process are handled the same way as in Bitcoin, with the longest-chain rule. As a result this algorithm does not provide transaction finality and does not fit the usual definition of consensus in the permissioned model.

Moreover, this algorithm only tolerates crash faults (*i.e.,* not malicious nodes) [39] and thus should be compared with other crash tolerant systems, like Raft [40] or Apache Kafka. At the time of writing, there was no available assessment of DiversityMining performances with a high number of nodes. However, since it exhibits a communication pattern similar to public blockchains, it can be expected to have viable performance with large sized network, at least under optimal conditions.

*g) Raft:* Raft [40] is a consensus algorithm that has been designed to be understandable and modular. It tolerates up to 50% of crashed nodes, which is the maximal threshold for this kind of faults. It is based on a leader that does the ordering of transactions. Leader election is triggered by a timeout, but contrary to PBFT, the timeout is randomized for each server. As a result, the elected leader (if successful) will be the first to timeout. Once there is an available leader, it can simply broadcast transactions to impose its version of the transaction log. Raft has been widely adopted and has a great number of implementations, so its robustness and practical performances are well known [41].

### B. Benchmarking Existing Technologies

Blockchain is currently under extensive research and development, leading to a high market fragmentation, with more than 20 different technologies and frameworks, which have been released by companies, open-source communities and universities. Table III compares the key characteristics of some popular blockchain technologies, especially for the context of enterprise and consortium based case studies. The column consensus algorithm lists the consensus engine that have a compatible implementation. In column Smart Contract we give the mechanism that execute the smart contract (*e.g.,* virtual machine, specification), if this feature is available. In Private Transactions, we specify whether the platform allows client to send transaction whose contents are only available to the recipients, possibly including their identities. Then we give a rough estimation of the platform popularity and activity based on github metrics and online presence (*e.g.,* download count, community hubs activity, *etc.*)

Two points should be noted regarding private transactions: First, if the blockchain allows arbitrary data in transactions, it is always possible to add encrypted data using the recipients public key, but then the validators cannot verify the semantics of the transaction (*e.g.,* double spending), although some solution exists to this issue [42][43]. Second, in the particular consortium setting, it is always possible to instantiate new blockchains for a specific subset of users, which is conceptually what Hyperledger Fabric does with its *channels*. However, this is limited in the sense that anything that happens within a channel cannot have an influence on something external to it, *e.g.,* a monetary transaction cannot be redeemed outside the channel where it has been made, only the members can vouch for it. In the corresponding column, we state "Yes" when the platform has a working implementation of a feature that allows for some level of privacy, including the mentioned techniques, "No" otherwise.

As it can be seen from Table II, consensus algorithms in the classical setting, *i.e.,* that tolerate 33% byzantine nodes, have similar characteristics: they show good performance but do not scale well when the number of nodes increase. There are proposals to address this issue such as SCP or PoET, but they imply alternative models that are subject to caveats, especially regarding trust assumptions.

Therefore, an application that requires cooperation from a small set of distrusting entities will be able to use a more classical consensus algorithm such as Tendermint or PBFT, and consequently will have a wide range of choices for a blockchain framework. Then, the determining factor is related to the additional features that are specific to each blockchain platform, such as the ones listed in Table III.

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

On the other hand, applications that require a larger network, *e.g.,* in a scenario where each registered user of the decentralized application needs to participate to the consensus, the choice will have to involve some performance trade-off. For instance, PoET may meet the scalability requirements, but not provide a low enough confirmation time. As a result, it may be difficult to find a fitting blockchain platform that also have a consensus algorithm meeting those requirements. Finally, applications that does not actually need a trustless platform can rely on crash tolerant systems where scalability is less an issue. Therefore, the choice for a blockchain framework will be driven by the specific characteristics of each platform, similarly to the first case.

## IV. Applications and Case Studies

The advent of the blockchain technology has enabled a wide range of new applications. In this section, we introduce a general classification of these applications, followed by examples of case studies in key domains.

### A. Classification

In a centralized world, an ecosystem is organized around one predominant actor. This architecture has advantages: it is easy to manage, stable and the responsibilities are clear. But it also has drawbacks: a monopolistic approach creates barriers to entry and hinders innovation, it can also result in an unbalanced value distribution among participants.

With the blockchain, we move towards a distributed approach with 4 categories of benefits: data traceability for a new trust paradigm, systems interoperability for process automation, flexibility for services enhancement, token-based ecosystems for a shared governance.

*1) Data traceability for a new trust paradigm:* The blockchain can be seen as an immutable distributed ledger where transactions are timestamped by block. The derived properties (transparency, integrity, immutability) create the trust conditions that enable a new framework for asset transactions. Indeed, from a vision perspective, the blockchain is to value exchange what the Internet is to information exchange: an interconnected network developed, maintained and updated by the participants for their benefits, a common ground that is safe, neutral, disintermediated and universal. Based on this framework, it is now possible to better trace, manage and monetize data. This offers new perspectives from machine to machine transactions, to identity attributes sharing, to user-centric data marketplaces.

*2) Systems interoperability for process automation:* The blockchain can also help bringing down domains silos. As an example, the transport infrastructure will more easily interoperate with the energy infrastructure. How so? As a shared ledger, the blockchain creates the conditions for process standardization, via shared on-chain data models, smart contracts and rules, between any actors that would like to work together. These can be actors within a domain, say a supplier and a customer, or actors from different domains, as in the electro-mobility example. Once standardized, processes can then easily be automated by using smart contracts (also known as chaincodes in the Hyperledger vocabulary).

*3) Flexibility for services enhancement:* Based on these interoperable systems, it is now possible to develop new features or services. Having access to more shared data and more standardized processes will definitely help. But the new operational organization will help too. Because there is no going through a third party, everyone on the blockchain is free to implement new features or services directly, at its own pace (and its own risk). More freedom also means more responsibilities. But this flexibility is good for services evolution and improvement. For example, it becomes easy to create ad-hoc temporary offers. It is also becomes easy to provide personalized services, by leveraging the shared data mentioned above.

*4) Tokens-based ecosystems for a shared governance:* Beyond the traceability, interoperability and flexibility benefits, the blockchain can help fundamentally transform ecosystems by leveraging digital assets, also known as tokens. Indeed, tokens can be used to track and monetize transactions, but they can also be considered as a great tool to materialize the governance rules and maintain an equilibrium among actors. The minting protocol and the trading rules put in place will reflect the consortium view on how to steer behaviors and share the value created. As an example, a green token could be created and used only in environmental-friendly scenarios. Going a step further, a decentralized autonomous organization (DAO) is another way to create new ecosystems. A DAO is an organization that relies on the blockchain to manage interactions between participants. Rules are implemented in smart contracts executed on the blockchain, so the governance is executed automatically with transparent rules and immutable actions.

### B. Case Studies

While we classified above the blockchain use cases in several categories, we can now look at concrete examples in different application domains.

*1) Finance and Insurance:* The first blockchain application was the cryptocurrency Bitcoin. But many use cases have followed since. As an example, it can be used at the ”data level” to issue and trade assets, such as bonds, in a decentralized market place (see the proof-of-concept from Caisse Des Depots in France). At the ”infrastructure level” Chaincore implements the distributed ledger technology for clearing and settlement, as a way to lower costs and improve efficiency. The blockchain can also help with processes such as KYC (Know Your Customer), by sharing the proof of identity and not the data itself between banks (see KYC-chain as an implementation example). At the ”service level”, we can imagine new offers such as personalized short term insurances [44], created on the spot and taking into account diverse pieces of information about the beneficiary. Finally, crowd-sharing an insurance deductible can be a good DAO application in the insurance sector.

*2) Energy:* With the rise of solar panels and other green sources of energy, the energy production is becoming more decentralized and offers a promising field for blockchain applications. As an example, the distributed ledger technology can be used to certify the data, the source of energy production, therefore, guaranteeing that it is environment-friendly. It can also be used to trade energy at the local grid level, between individual producers and consumers (see the proof-of-concept

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

from LO3 Energy in Brooklyn or [45]). We can imagine further benefits in the home where devices can schedule their energy charging to optimize costs and exchange data autonomously between them. This can be a totally new ecosystem model, where tokens are directly exchanged between parties to incentivize appropriate (green) behaviors.

*3) Mobility:* In this sector, the distributed ledger technology can be used to safely store the car data (for example, its mileage [9] or certificates [46]). We can also look at electro-mobility use cases, where the mobility infrastructure (say electric vehicles) interoperates more easily with the energy infrastructure (say the charging points). Another application example is chasyr, which is a blockchain-based ride-sharing platform that matches passengers and drivers. So, this is basically an uber-like service, in a decentralized architecture. One last example would be a decentralized transportation ecosystem, where people can use a same token to ride on a bus, rent a bike or carpool, without any central authority to organize its operation.

*4) Logistics:* In this sector, the distributed ledger technology can be used to track an asset. For example, Everledger tracks diamonds to ensure their authenticity, Provenance can track food origin to guarantee its sanitary safety. Another example would be using a blockchain to create a collaborative IT system, which matches transporters and customers timetable for efficient delivery.

## V. A Practical Case Study - Analyzing the EtherMint Technology

In this section, we provide a detailed study related to the performance evaluation of the Ethermint consortium blockchain technology. Being based on Ethereum, this technology inherits all the capabilities including the EVM and smart contracts. Moreover, the consensus relies on the Tendermint protocol. By combining those two characteristics, a versatile protocol and a lightweight consensus that remains secure, we believe that Ethermint is a great candidate for various use cases.

Although extensive studies have been conducted to assess the performance of the Tendermint consensus protocol such as [31] or other blockchain technologies such as [22], to the best of our knowledge, the literature has not provided yet any detailed study related to the technical performance of Ethermint. That is the first reason behind using this technology for benchmarking. The second reason lies in the fact that many industrials, such as in [9], are looking for the usage of this technology for their own use cases. However, they do not have yet any detailed assessment of the performance of this technology. Therefore, for the sake of helping them testing this technology, this study is conducted.

The remainder of this section is organized as follows. First we give a detailed description of the Tendermint consensus protocol before presenting Ethermint itself (Sections V-A and V-B). Then, we specify the experimental setup and the results regarding the transaction validation speed depending on multiple factors such as the transaction load and the network topology (Sections V-C and V-D).

### A. Tendermint: Overview and Architecture

Tendermint [30] can be seen as a software for replicating an application on many machines in a secure and consistent manner. Tendermint protocol guarantees that machines compute the same state and it can tolerate up to 1/3 of malicious or failing machines.

From a functional architecture point of view, Tendermint consists of two main components: a blockchain consensus engine called Tendermint Core, which is used to ensure that the same transactions are recorded on every machine in the same order, and a generic application interface called the Application blockchain Interface (ABCI) that is used to enable the transactions to be processed in any programming language.

In contrast to most current blockchain solutions (such as Bitcoin) that come pre-packaged with built in state machines, Tendermint can be used for replicating state machines related to applications written in any programming language or development environment. Thanks to these features, Tendermint has been widely used in a variety of distributed applications including blockchain platforms such as Ethermint [47] and Hyperledger Burrow [48].

From a consensus point of view, Tendermint belongs to the family of BFT (Byzantine Fault Tolerance) consensus protocols [49]. More precisely, participants in this protocol are called validators; their main role is to propose blocks of transactions and vote on them. Blocks come in the form of a chain and only one block is committed at each height.

As in most consensus protocols, a block may fail to be committed due to many reasons such as network connectivity or malicious behaviors. In such a case, the Tendermint protocol moves to the next round, and a new validator is designed to propose a block at this height level. The selection of a proposer is proportional to its validation power. Considering the voting phase, two stages are required to successfully commit a block; a pre-vote and a pre-commit. For any block to be committed, more than 2/3 of validators must pre-commit for it in the same round. We show in Figure 4 the optimal validation scheme of transactions using Tendermint.

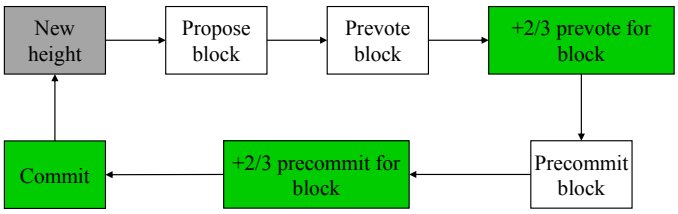


Figure 4. Tendermint consensus: optimal workflow

Finally, it should be noticed that Tendermint is overall qualified as a weakly synchronous protocol since validators continue to do their work only after hearing from more than 2/3 of the validators set. Furthermore, validators have also to wait for a small amount of time in order to receive a complete proposal block from the proposer before voting to move to the next round.

### B. Ethermint: Ethereum + Tendermint

As previously mentioned, Tendermint has a generic application interface that enables the transactions to be processed in any programming language. Indeed, to use Ethereum with a Tendermint consensus protocol, Ethermint has been developed. Using Ethermint was first introduced in May 2017, as part of

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Tendermint's goal to launch the COSMOS hub [50], the first blockchain in the Cosmos network, which is a decentralized network of independent parallel blockchains.

The key idea of Ethermint is to enable Ethereum to run on top of Tendermint. This allows developers to have all the nice features of Ethereum, while at the same time benefit from Tendermints consensus protocol implementation. Tendermint combined with Ethereum is supposed to result in fast block times, transaction finality while also getting the goodies of smart contracts.

In the next sections, the performance of the blockchain technology Ethermint is considered. This will encompass explaining the experimental setup, describing the assessment workflow, as well as, analyzing several indicators related to the performance of this technology.

### C. Experimental Setup

To assess the performance of Ethermint, several parameters are studied and many performance indicators are considered. The evaluation process consists of dynamically deploying a blockchain network on an Openstack virtual machine [51] having the following properties (20 GB of RAM and 6 Virtual Central Processing Unit). The used Tendermint version is $0.12.0$ and $0.5.3$ for Ethermint.

Parameters used for building the blockchain are: the number of nodes $n$, the number of validators $v$ and the network topology $t$. By this latter, we mean how nodes are connected to each other. In this work, the network may be ***complete*** (each node $i$ is directly connected to all other nodes), in the form of ***line*** (each node $i$ is connected to node $i+1$ except for the last node $n$), ***cycle chain*** (line and the last node is connected to the first node), ***enhanced chain*** (a node $i$ is connected to nodes $i+1$ and $i+2$; the $i+2$ node of the node $n-1$ is the first node; the $i+1$ node of the last node is the first node and the $i+2$ node of the last node is the second node).

The above explained topologies are represented in Figure 5. The main reason why the network topology is considered is due to the fact that the Tendermint consensus protocol lies in a very important gossiping phase between all nodes in the network in order to accomplish the validation steps (look at Figure 4 for more information). Therefore, any topology will certainly affect the speed at which the information circulates in the network.

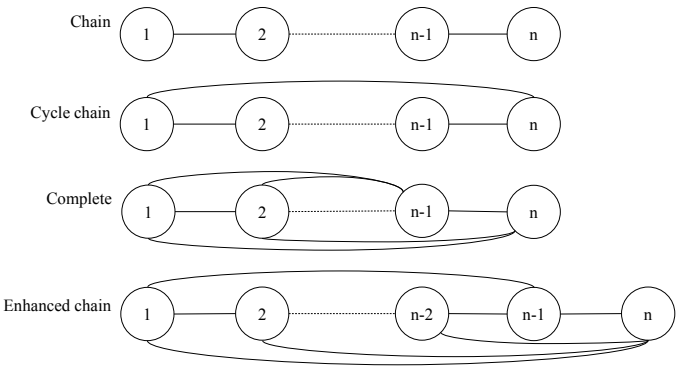


Figure 5. Network topology

To perform the deployment process, a NodeJS web service has been developed. The aforementioned parameters are used as an input for this service. A ready to use blockchain respecting those parameters is generated as an output. More precisely, the service works as follows: A NodeJS server always listens on a specific port; once a correct request is received, $n$ containers are built ($n$ corresponds to the number of nodes in the blockchain). Each container will hold a Tendermint node associated with an Ethermint node.

To deal with the validators for the Tendermint consensus, the first $v$ Tendermint nodes will be selected. Besides, each node in the validators set will ask for the public key of all other validator nodes in order to build a common genesis file. This latter contains the list of validators and the power associated with each of them. We assume in this work that validators have the same validating power.

Once the genesis file is ready and successfully shared, Tendermint nodes will start and so will do Ethermint nodes. Moreover, the web service will check that all containers are successfully started and the blockchain is ready to use by asking for the first block information; a success flag is returned if and only if the blockchain already starts validating blocks (in this case empty blocks are generated). Moreover, another service has been developed in order to destroy the blockchain. By destroying, we mean removing all containers and data related to a blockchain.

Besides, a separate Java program that is run on a 8 GB of RAM is used for interacting with the blockchain, as well as, sending transactions and computing performance indicators. As for all Ethereum based blockchains, instances of web3j client have been used. To assess the blockchain performance, this program will dynamically create and remove a blockchain by calling the corresponding services.

Moreover, this program will be in charge of sending transactions in asynchronous mode (*i.e.,* we do not wait for the transaction to be validated). The number of transactions to be sent in second as well as, the sent interval duration and the dynamic blockchain parameters are considered as input for this program. The sent transaction consists of adding an element to a map in the smart contract that is written in Solidity. The map assigns a random value to the account calling the smart contract.

At the end of each scenario (*i.e.,* after the assessment duration), a validator blockchain node is contacted in order to fetch all validated transactions and map them with the sent transactions. By doing so, several metrics can be computed such as the duration between the sending time of a transaction and the time at which the transaction has been written on the blockchain (the validation time of a transaction). Furthermore, the number of transactions that each block contains is computed, as well as, the time between two blocks. We show in Algorithm 1 the assessment workflow from a high level point of view.

### D. Experimental Results

We start this section by analyzing the impact of both the number of transactions sent per second, and the number of validators on the blockchain performance. By this latter, we mean the number of transactions validated per second. The network topology in this scenario is fixed to be complete. The results are compared with the ideal case line where all sent transactions are validated in one second or less.

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

**Algorithm 1:** Assessment workflow

**input :** Number of nodes $n$,
Number of validators $v$,
Topology of the network $t$,
Assessment duration $d$
Number of transactions per second (frequency $f$)

**output:** Performance indicators (average transaction validation time, average block size, number of transactions validated per second, etc)

```
1  begin
       // BlockChain Deployment
2      buildBlockChain(n,v,t)
3      waitForBlockChainToBeReady()
       // Send Transactions
4      while duration < d do
5          for i = 0; i < f; i++ do
6              tx ← PrepareTransaction()
7              send(tx) // Send Asynchronously
8
9          end
10         sleep(1s)
11     end
       // Extract performance indicators
12     ComputeAverageValidationTime()
13     ComputeAverageBlockSize()
14     ComputeAverageTransactionsPerSecond()
15 end
```

As can be seen from Figure 6, the results show that more the number of validators increases, more the number of transactions validated per second decreases. This can be explained by the fact that more communications are required to pre-vote and pre-commit a block when the number of validators becomes important. Besides, it has been noticed that the number of transactions sent per second has an impact on the output of the blockchain.

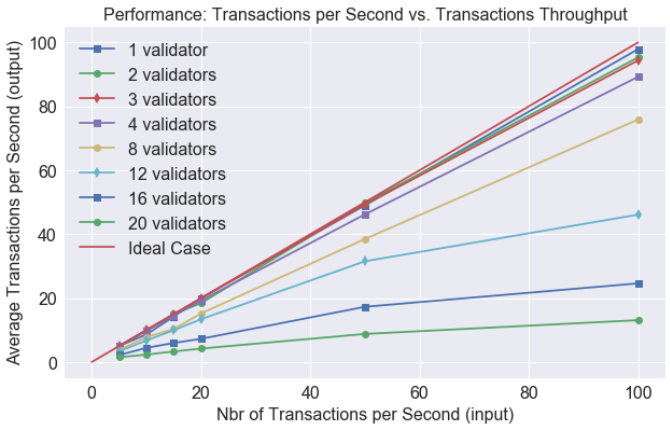


Figure 6. Average transactions per second

More precisely, more transactions sent per second, more the network accumulates some delays to validate transactions. For instance, in the worst case when 100 transactions are sent per second, a network of 1, 2, 3 or 4 validators can almost validate them in one second or less. However, for a network containing 8, 12, 16 or 20 validators, the blockchain requires several seconds to validate all the input.

When it comes to assessing the impact of the size of validators set on the average transaction's validation time, the results in Figure 7 show that more the number of validators is important, more the time required to valid a transaction increases.

For instance, in a network containing 1 validator, the average validation time is very low (less than 1 second), however, that increases to approximately two minutes when 20 validators are considered. The confidence interval is also given in this figure.

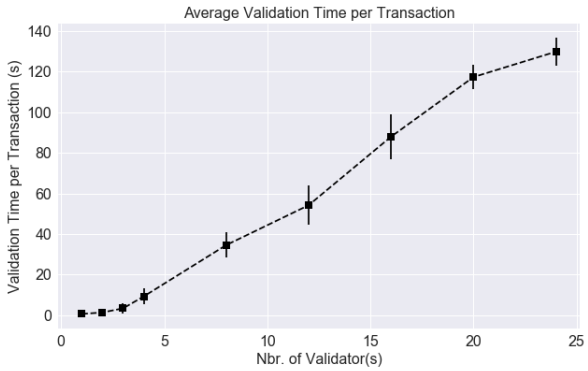


Figure 7. Validation time

From both Figures 6 and 7 it can be said that a compromise has to be made between the number of validators considered, the number of transactions sent per second and the desired blockchain performance. More precisely, an additional effort has to be made in order to select the appropriate number of validators in the network, and to fix the adequate number of transactions that the blockchain can deal with.

This usually depends on the use case where the blockchain is used. For instance, within a blockchain based energy market place, that requires validating 100 of energy exchange transactions every 20 seconds, it is obvious that it wont be possible to achieve that using more than 4 validators.

In the following, the impact of the network topology is studied. As previously mentioned, four topologies are considered: complete, chain, cycle chain and enhanced chain. To do the assessment, the number of validators has been fixed at 20 and the input transactions number is varying.

The results in Figure 8 indicate that the topologies are in the following order (from best to worse) enhanced chain, cycle, chain, complete regarding their performance in terms of the number of transactions that have been validated. This can be explained by the fact that in a complete network, the validator nodes spend more time in order to synchronize their state with the rest of the network. As a result, such nodes have less time in order to accomplish the validation work.

In a chain topology, the time required to spread an information (*i.e.,* send or receive votes) is quite high in comparison with other topologies. In contrast to those latter, the enhanced chain topology, represents a good compromise between the spread information time and the synchronization effort. More precisely, each node is to be synchronized with only four nodes

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

(see Figure 5) and the time required to spread information is more optimized in comparison with other topologies.

In the following the time between two blocks is studied. The results in Figure 9 show that more the number of validators increases, more the inter-blocks delay increases. This is coherent with what have been previously obtained. Indeed, in a network containing an important number of validators, a validator node requires additional time in order to inform/get informed about the state of other nodes. That will certainly delay the validation/creation of new blocks in the network.

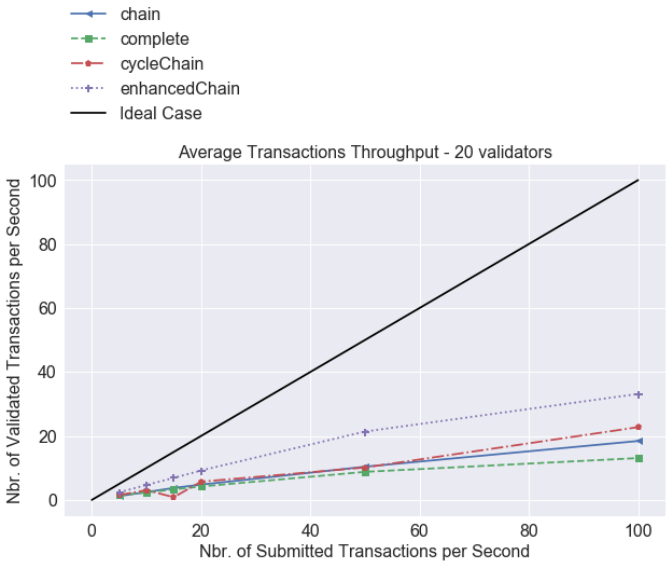


Figure 8. Network topology effect

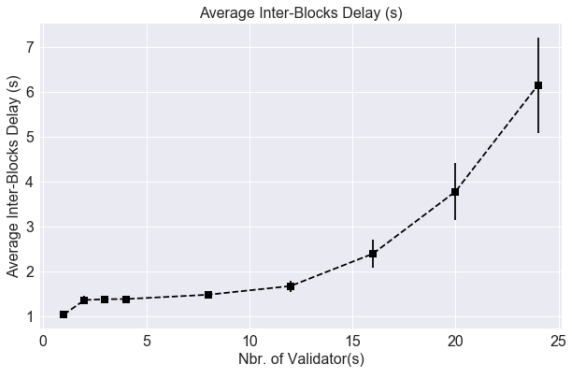


Figure 9. Average inter blocks delays

The last parameter studied in this paper is the size of the blockchain. By this we mean the actual size of the folder containing the blockchain information. For this purpose, four scenarios have been considered. The first one consists of sending 20 transactions per second during 2 hours to a network containing 1 validator.

The results in Figure 10 show that the blockchain size increases with time. After two hours, the final size increases to approximately 90 Megabytes. When 20 validators are considered, the size is not so important as in the previous scenario since few transactions will be validated. As a result, it can obviously be said that the size of a blockchain is proportional to the transactions that the blockchain writes. That is, more validated transactions will obviously result in higher blockchain size.

A final note is related to the sudden decrease in the size. This is actually due to a compression mechanism used in Ethermint. More precisely, at each increase of approximately 35 megabytes, a compress mechanism is applied by which approximately 15 megabytes are gained.

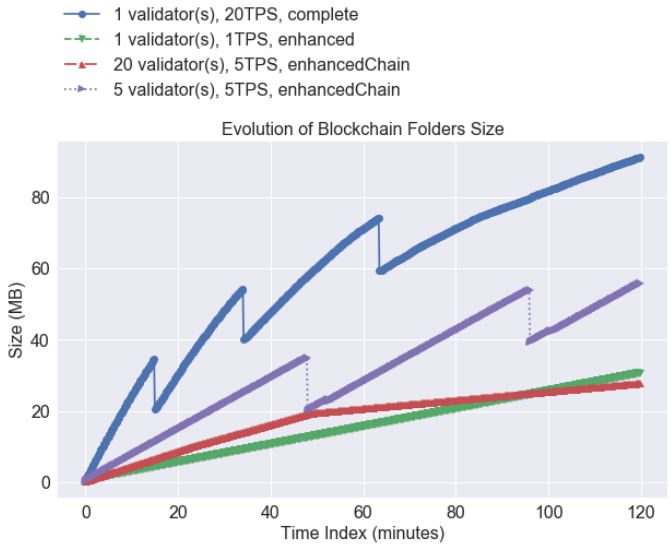


Figure 10. Blockchain folder data evolution

From the above results, it can be concluded that the performance of the Ethermint technology will certainly limit its usage in practice to specific use cases where the time required to valid a transaction is very high, and the number of validators joining the network is not very important. Besides, the storage space can also constitute a bottleneck for this technology to be used for real world applications. For instance, running the blockchain for several weeks will certainly be a problematic for certain use cases.

## VI. Research Directions and Opportunities

Blockchain is currently under extensive research and development from both the academia and the industry, however, there are still major challenges to be overcome before mass market penetration and adoption. In this section, we highlight major research directions and opportunities that we believe are important to investigate.

### A. Data Analysis and Visualization

A blockchain being no more than a ledger of transactions between accounts, data from a blockchain can be seen as no more than nodes connected by occasionally existing multi-property edges. Under which structural form should they be tackled depends on the aimed analysis. From a blockchain *network supervision* point of view, crucial in a private companies consortium, the relevant data aggregation level is the

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

block, with a time-series scheme. From the point of view of auditing the *quality of the user activity*, transactions should be considered the atomic level to investigate, under a graph scheme, and more specifically under a time-varying graph (TVG) scheme [52].

The aim of efficiently auditing a blockchain brings several challenges:

*1) Real-time analysis:* Because of the possibility of forks, there is no such thing as absolute reliability of the data retrieved from the blockchain. It is decreasingly high toward the most recent blocks data, as one only get the version of the ledger stored on a node at a given time, so that a blockchain-specific time-dependent reliability weight has to be determined. This procedure must be highly dependent on the chosen consortium governance scheme.

*2) Exploitable visual representation of TVG:* From a graph point of view, each edge (transaction) represents a unique and directed communication bridge between nodes, having an infinitesimally narrow time-width. To be able to graphically analyze a blockchain networks, or to compute common graph indicators such as centrality or community borders, systematic smart ways to define edges weight based on non-Dirac delta function in time have to be conceived.

*3) Smart contract internal transactions unraveling:* Unless explicitly coded as so, the transactions from and to smart contracts, or from smart contracts to users, are no written down in the ledger, and this can be used for transaction obfuscation allowing token laundering [53], Ponzi scheme [54] or other uses where the blockchain only serves itself. In order to determine whether or not blockchain transactions are related to real-world event, or more generally what it is used for, studies on specific key quality indicators related to smart contract have to be conducted.

### B. Blockchain Audit

Data immutability is generally put forward when referring to blockchain technologies. However, as already discussed in Section II.A, the written data could still be tampered and the blockchain rebuilt as long as the majority of the participants (or miners) have reached a consensus. This is especially true in consortium and private blockchains where the number of miners is generally limited in comparison with public blockchains.

In this context, it becomes extremely difficult for a regulation authority to audit consortium based blockchains and to check whether the data and transactions have been tampered with or not. A commonly adopted solution consists in piggybacking data hashes from the consortium blockchain into the Bitcoin network, by embedding those hashes inside the OP_RETURN field of Bitcoin transactions. However, this contribute in polluting and increasing the size of the Bitcoin network with nonsense and non-financial data.

More recently, alternatives solutions have been proposed to reduce the impact of piggybacking on public blockchains, including the concepts of side-chains and notary chains whose main objective is to make it extremely hard for malicious users and/or the network participants to alter the blockchain data.

### C. Governance

The governance in a private blockchain assigns authority and responsibility among the consortium members. It determines nodes that will be able to create blocks (*i.e.,* miners), to read/write data, to contribute in the consensus mechanism (*e.g.,* voting for a miner) and/or to participate in decisions for the system evolution (*e.g.,* operations management [55], software updates, allow new nodes to join the system etc.). This power distribution has an impact not only within the system but also on the business model of the use case.

Costs linked with the system activity such as the system set-up, its execution or maintenance are shared within the consortium according to the governance scheme. It also affects future incomes or losses at a business level since the governing nodes decide the rules of the system. For example, the majority of governing nodes can agree to allow the membership of a new entity within the consortium, and which is competitor of another member who has no power over this decision. Hence, this could jeopardize the viability of the system.

The viability of the system can also be affected by the governance definition. In many cases, to be durable, the consortium has to be able to grow by allowing new members to integrate the system. It is the case for example of new services over blockchain like dematerialized car service books. More companies join the consortium such as car manufacturers, car repair shops or insurance companies, the more durable and available is the system. On the other hand, the power is dissolved with the growth of consortium.

One should also take into account the impacts on the business model when building the governance scheme as it will be discussed in the next section.

### D. Incentives and Business Models

Blockchain solves the issues of trust between actors in situations of exchange where the temptation of cheating is high by *removing* this need of trust. Any business model based on a solution that would not claim to solve a trust issue would inevitably fail, as its solution could be replaced by a less constraining and probably already existing centralized system.

In a blockchain whose users are exclusively individuals, the pecuniary incentives must ensure that, because members either receive additional incomes or just lessen their expenses, they find a financial interest in participating to the process.

However, in a consortium of commercial entities, it should be pointed out that the simple fact *not to* be part of the consortium might represent a handicap that could lead to loss of turnover or customers attrition, because of the latter attraction to blockchain promises and interest in financial incentives.

### E. Data Privacy

Data privacy is an imperative for enterprise blockchains. But lets first distinguish anonymity and privacy. A transaction is considered anonymous if we cannot identify its owner, whereas a transaction is called private if the object and the amount of transaction are unknown.

We have seen many schemes on public blockchains to improve privacy: Stealth Addresses, Pedersen Commitments, Ring signature, Homomorphic encryption, Zero-knowledge-proof. No scheme can hide the sender, the receiver and the

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

amount at the same time, so we see actual implementations mixing these techniques in order to achieve the desired level of privacy. In addition, there are some known drawbacks such as computational time, so further research is needed. But we can expect that these initiatives on public blockchains will drive improvement on enterprise blockchains privacy as well. Interested readers may refer to [56] for more detailed information on privacy issues in blockchain.

### F. Security

Guaranteeing End-to-End security means identifying vulnerabilities and mitigating risks at each element level and at the system level. This goes beyond looking at the blockchain building blocks (consensus, distributed network, cryptographic tools) and includes evaluating the virtual machine, the Smart Contracts, the Oracle, the user client, the hardware component, the keys management and PKI, etc. Some areas of research are the following: Formal verification of smart contracts, Usage of trusted platform modules for key storage, Identification of the different types of attack vectors and their counter strategies (sybil attacks, double spending attacks, distributed denial of service attacks, botnet attacks, storage specific attacks, censorship, *etc.*), Audit (detect issues a priori or a posteriori), Supervision (detect issues during run time). Interested readers may refer to [57] for more detailed information on security challenges in blockchain.

### G. Scalability

As usual, there is always a trade-off between costs, security and performance. Because participants are known in enterprise blockchains, the scalability issue is therefore easier to solve, as compared to public blockchains. Yet, in order to achieve scalability, we first need to keep in mind the usage context and the performance metrics we want to optimize: transactions throughput, validation latency, number of participant nodes, number of validating nodes, energy costs, computation costs, storage costs or other criteria? As always, remember the trade-off principle: A round robin consensus algorithm will scale well, but the participants need to be honest. A PBFT algorithm can recover from malicious behaviors (up to 1/3) but the validating nodes should not be too many (tens of nodes at most) if the system is to work [58]. All in all, scalability is an active area of research and we can mention some initiatives such as: fragmenting the global ledger into smaller sub-ledgers run by sub-groups of nodes, removing old transactions in order to optimize the storage, using a hierarchy of blockchains (transactions are done at a higher level and settled optionally afterwards in the blockchain), and so on.

## VII. Conclusions

The blockchain technology represents a major paradigm shift in the way business applications will be designed, operated, consumed and marketed in the near future. In this paper, we discussed in details the concept of consortium blockchains, in terms of architecture, technological components and applications. In particular, we analyzed and compared various consensus algorithms. An experimental study was then proposed in order to assess the performance of the Ethermint technology. The performance indicators analyzed are the time required to validate a transaction, the number of transactions validated per second, as well as the average inter blocks delay and the size of the blockchain data folder. Parameters that have been varied are the number of validators, the number of transactions submitted per second and the topology of the network. The results highlighted some limitations in terms of transactions validation time and storage requirements that may hinder the usage of Ethermint to deal with some real world use cases. Finally, we highlighted some major research challenges that need to be addressed before achieving mass market penetration, including the issues related to governance, audit, scalability, incentives, data privacy and security.

## Acknowledgment

This research work has been carried out under the leadership of the Institute for Technological Research SystemX, and therefore granted with public funds within the scope of the French Program Investissements d'Avenir.

## References

[1] E. Ben Hamida, K. L. Brousmiche, H. Levard, and E. Thea, "Blockchain for Enterprise: Overview, Opportunities and Challenges," in The Thirteenth International Conference on Wireless and Mobile Communications (ICWMC 2017), Nice, France, Jul. 2017.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, accessed: 2017-05-25. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[3] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," IEEE Access, vol. 4, 2016, pp. 2292–2303.

[4] M. Mettler, "Blockchain technology in healthcare: The revolution starts here," in 2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom), Sept. 2016, pp. 1–3.

[5] "stampd.io: A document blockchain stamping notary app," accessed: 2018-05-25. [Online]. Available: https://stampd.io/

[6] A. Yasin and L. Liu, "An online identity and smart contract management system," in 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), vol. 2, June. 2016, pp. 192–198.

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

[8] F. Tian, "An agri-food supply chain traceability system for china based on rfid blockchain technology," in 2016 13th International Conference on Service Systems and Service Management (ICSSSM), June. 2016, pp. 1–6.

[9] K.-L. Brousmiche, T. Heno, C. Poulain, A. Dalmieres, and E. Ben Hamida, "Digitizing, securing and sharing vehicles life-cycle over a consortium blockchain: Lessons learned," in Proceedings of 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 1st International Workshop on Blockchains and Smart Contracts (BSC), Feb. 2018.

[10] Z. Zheng, S. Xie, H.-N. Dai, and H. Wang, "Blockchain challenges and opportunities: A survey," International Journal of Web and Grid Services, 2017, pp. 1–23.

[11] "Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016," accessed: 2018-05-25. [Online]. Available: http://eur-lex.europa.eu/eli/reg/2016/679/oj

[12] V. Buterin, "On Public and Private Blockchains," 2015, accessed: 2018-05-25. [Online]. Available: https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/

[13] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in 2017 IEEE International Congress on Big Data (BigData Congress), June. 2017, pp. 557–564.

[14] "wiki: The Ethereum Wiki -," Feb. 2018, accessed: 2018-05-25. [Online]. Available: https://github.com/ethereum/wiki

[15] "Quorum | J.P. Morgan," accessed: 2017-07-01. [Online]. Available: https://www.jpmorgan.com/country/US/EN/Quorum

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[16] "Ethereum," Feb. 2017, accessed: 2018-05-25. [Online]. Available: https://fr.wikipedia.org/w/index.php?title=Ethereum&oldid=134391073

[17] "Hyperledger Project," accessed: 2018-05-25. [Online]. Available: https://github.com/hyperledger/

[18] E. Buchman. Tendermint: Byzantine fault tolerance in the age of blockchains. Accessed: 2018-05-25. [Online]. Available: https://allquantor.at/blockchainbib/pdf/buchman2016tendermint.pdf (2018)

[19] N. Knezevic. A high-throughput byzantine fault-tolerant protocol. Accessed: 2018-05-25. [Online]. Available: https://infoscience.epfl.ch/record/169619/files/EPFL_TH5242.pdf (2018)

[20] Hedera hashgraph platform. Accessed: 2018-05-25. [Online]. Available: https://www.hederahashgraph.com/platform#speed (2018)

[21] O. Band. Stellar load testing results for the kin ecosystem. Accessed: 2018-05-25. [Online]. Available: https://medium.com/kin-contributors/stellar-load-testing-results-for-the-kin-ecosystem-64c4d8676e69 (2018)

[22] Get started with the basics of the stellar network. Accessed: 2018-05-25. [Online]. Available: https://www.stellar.org/how-it-works/stellar-basics/ (2018)

[23] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poet)," in Stabilization, Safety, and Security of Distributed Systems, P. Spirakis and P. Tsigas, Eds. Cham: Springer International Publishing, 2017, pp. 282–297.

[24] Multichain developers q&a: About throughput performance. Accessed: 2018-05-25. [Online]. Available: https://www.multichain.com/qa/5556/about-throughput-performance (2018)

[25] D. Ongaro. Consensus: Bridging theory and practice. Accessed: 2018-05-25. [Online]. Available: https://ramcloud.stanford.edu/~ongaro/thesis.pdf (2014)

[26] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," ACM Transactions on Programming Languages and Systems (TOPLAS), vol. 4, no. 3, 1982, pp. 382–401.

[27] M. Pease, R. Shostak, and L. Lamport, "Reaching agreement in the presence of faults," Journal of the ACM (JACM), vol. 27, no. 2, 1980, pp. 228–234.

[28] M. Castro and B. a. Liskov, "Practical Byzantine fault tolerance," in Third Symposium on Operating Systems Design and Implementation (OSDI), vol. 99, 1999, pp. 173–186.

[29] J. P. Morgan. Quorum description. Accessed: 2018-05-25. [Online]. Available: https://github.com/jpmorganchase/quorum/wiki (2018)

[30] J. Kwon, "Tendermint: Consensus without mining," 2014, accessed: 2018-05-25. [Online]. Available: https://tendermint.com/static/docs/tendermint.pdf

[31] E. Buchman, "Tendermint: Byzantine fault tolerance in the age of blockchains," Ph.D. dissertation, 2016, accessed: 2018-05-25. [Online]. Available: https://allquantor.at/blockchainbib/pdf/buchman2016tendermint.pdf

[32] Tendermint's software ecosystem. Accessed: 2018-05-25. [Online]. Available: https://tendermint.com/ecosystem (2018)

[33] V. Costan and S. Devadas, "Intel sgx explained." IACR Cryptology ePrint Archive, 2016, p. 86, accessed: 2018-05-25. [Online]. Available: https://eprint.iacr.org/2016/086.pdf

[34] Intel. Hyperledger Sawtooth description. Accessed: 2018-05-25. [Online]. Available: https://sawtooth.hyperledger.org/docs/core/releases/latest/introduction.html (2018)

[35] D. Mazieres, "The stellar consensus protocol: A federated model for internet-level consensus," 2015, accessed: 2018-05-25. [Online]. Available: https://www.stellar.org/papers/stellar-consensus-protocol.pdf

[36] L. Baird, "Hashgraph consensus: fair, fast, byzantine fault tolerance," Swirlds Tech Report, Tech. Rep., 2016, accessed: 2018-05-25. [Online]. Available: http://www.swirlds.com/wp-content/uploads/2016/06/2016-05-31-Swirlds-Consensus-Algorithm-TR-2016-01.pdf

[37] I. Swirlds. Swirlds website. Accessed: 2018-05-25. [Online]. Available: https://www.swirlds.com/ (2018)

[38] "MultiChain | Open source private blockchain platform," accessed: 2018-05-25. [Online]. Available: http://www.multichain.com/

[39] C. Cachin and M. Vukolić, "Blockchains consensus protocols in the wild," arXiv preprint arXiv:1707.01873, 2017, accessed: 2018-05-25. [Online]. Available: https://arxiv.org/pdf/1707.01873.pdf

[40] D. Ongaro and J. K. Ousterhout, "In search of an understandable consensus algorithm." in USENIX Annual Technical Conference, 2014, pp. 305–319.

[41] "The raft consensus algorithm," accessed: 2018-05-25. [Online]. Available: https://raft.github.io/

[42] B. Parno, C. Gentry, J. Howell, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," Cryptology ePrint Archive, Report 2013/279, 2013, accessed: 2018-05-25. [Online]. Available: https://eprint.iacr.org/2013/279

[43] R. Mercer, "Privacy on the blockchain: Unique ring signatures," arXiv preprint arXiv:1612.01188, 2016, accessed: 2018-05-25. [Online]. Available: https://arxiv.org/pdf/1612.01188.pdf

[44] M. Raikwar, S. Mazumdar, S. Ruj, S. S. Gupta, A. Chattopadhyay, and K.-Y. Lam, "A blockchain framework for insurance processes," in Proceedings of 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 1st International Workshop on Blockchains and Smart Contracts (BSC), Feb. 2018.

[45] J. Horta, D. Kofman, D. Menga, and A. Silva, "Novel market approach for locally balancing renewable energy production and flexible demand," Dresden, Germany, Oct. 2017.

[46] N. Lasla, M. Younis, W. Znaidi, and D. Ben Arbia, "Efficient distributed admission and revocation using blockchain for cooperative its," in Proceedings of 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 1st International Workshop on Blockchains and Smart Contracts (BSC), Feb. 2018.

[47] Ethermint. Ethermint project description. Accessed: 2018-05-25. [Online]. Available: http://ethermint.readthedocs.io/en/develop/ (2017)

[48] Hyperledger. Hyperledger project description. Accessed: 2018-05-25. [Online]. Available: https://www.hyperledger.org/projects/hyperledger-burrow (2017)

[49] K. Driscoll, B. Hall, H. Sivencrona, and P. Zumsteg, "Byzantine fault tolerance, from theory to reality," in International Conference on Computer Safety, Reliability, and Security. Springer, 2003, pp. 235–248.

[50] Cosmos. Cosmos white paper. Accessed: 2018-05-25. [Online]. Available: https://cosmos.network/about/whitepaper (2017)

[51] OpenStack. OpenStack description. Accessed: 2018-05-25. [Online]. Available: https://www.openstack.org/ (2018)

[52] A. Casteigts, P. Flocchini, N. Santoro, and W. Quattrociocchi, "Time-varying graphs ans dynamic networks," International Journal of Parallel, Emergent and Distributed Systems, vol. 27, no. 5, 2012, pp. 387–408.

[53] M. Moser, R. Bohme, and D. Breuker, "An inquiry into money laundering tools in the bitcoin ecosystem," in eCrime Researchers Summit (eCRS), 2013. IEEE, 2013, pp. 1–14.

[54] M. Bartoletti, S. Carta, T. Cimoli, and R. Saia, "Dissecting ponzi schemes on ethereum: identification, analysis, and impact," CoRR, vol. abs/1703.03779, 2017.

[55] T. Sato and Y. Himura, "Smart-contract based system operations for permissioned blockchain," in Proceedings of 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 1st International Workshop on Blockchains and Smart Contracts (BSC), Feb. 2018.

[56] M. Conti, S. K. E, C. Lal, and S. Ruj, "A survey on security and privacy issues of bitcoin," CoRR, vol. abs/1706.00916, 2017.

[57] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," CoRR, vol. abs/1802.06993, 2018.

[58] M. Vukolic, "The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication," Open Problems in Network Security (iNetSec), 2015, pp. 112–125.

2018, © Copyright by authors, Published under agreement with IARIA - www.iaria.org