IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 22, 2020, accepted July 2, 2020. Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2020.3007251

Scaling Blockchains: A Comprehensive Survey

ABDELATIF HAFID[1,2], (Member, IEEE), ABDELHAKIM SENHAJI HAFID[2], (Member, IEEE), AND MUSTAPHA SAMIH[1]

[1]Team of EDA–Mathematical Laboratory and their Applications, Department of Mathematics, Faculty of Sciences, University of Moulay Ismail, Meknes 50050, Morocco
[2]Montreal Blockchain Laboratory, Department of Computer Science and Operational Research, University of Montreal, Montreal, QC H3T 1J4, Canada

Corresponding author: Abdelatif Hafid (a.hafid@edu.umi.ac.ma)

This work was supported in part by Mohammed VI Polytechnic University - UM6P.

ABSTRACT Blockchain (e.g., Bitcoin and Ethereum) has drawn much attention and has been widely-deployed in recent years. However, blockchain scalability is emerging as a challenging issue. This paper outlines the existing solutions to blockchain scalability, which can be classified into two categories: *first layer* and *second layer solutions*. First layer solutions propose modifications to the blockchain (i.e., changing the blockchain structure, such as block size) while second layer solutions propose mechanisms that are implemented outside of the blockchain. In particular, we focus on sharding as a promising first layer solution to the scalability issue; the basic idea behind sharding is to divide the blockchain network into multiple committees, each processing a separate set of transactions. More specifically, (a) we propose a taxonomy based on committee formation and intra-committee consensus; and (b) we compare the main existing sharding-based blockchain protocols. We also present a performance-based comparative analysis (i.e., throughput and latency), of the advantages, and disadvantages in existing scalability solutions.

INDEX TERMS Blockchain, scalability, sharding, first layer solutions, second layer solutions.

I. INTRODUCTION

In recent years, Blockchain technology has been widely used in almost all industry segments, including the healthcare sector [24], [25], cryptocurrencies [1], [17], artificial intelligence [22], [23], the government sector [28], [29], internet of things [26], [27], and others [30]. Every one of these segments benefit from Blockchain's transparent, decentralized, immutable, and fully distributed peer-to-peer architecture that records digital assets (e.g., transactions). Despite these attractive characteristics, one of the key limitations of blockchain is scalability; indeed, the number of transactions that can be processed per second is small and insufficient (e.g., up to 7 for Bitcoin [1] and 15 for Ethereum [17]). This is unacceptable for most traditional centralized payment systems that require 1000s of transactions per second (tx/s); as a comparison, Visa handles an average of 1700 tx/s [32]. Generally, scalability is not well-defined in the literature. However, the scalability trilemma is well-known in blockchain; it was first described by Vitalik Buterin, the co-founder of Ethereum [38]. Vitalik states that trade-offs are inevitable between three important properties: decentralization, scalability, and security (see Figure 1). Decentralization is the core and the nature of blockchain, security is an essential propriety, whereas scalability is the main challenge. In other words, the scalability trilemma states that we can only have two out of either decentralization, scalability or security (i.e., we can pick just one side of the triangle shown in Figure 1; thus, trade-offs are almost inevitable). The key challenge is to achieve scalability, security, and decentralization at the same time. Several solutions to the scalability issue have been proposed in the literature, such as sharding [20], Directed Acyclic Graph [84], and Lightning Network [36].

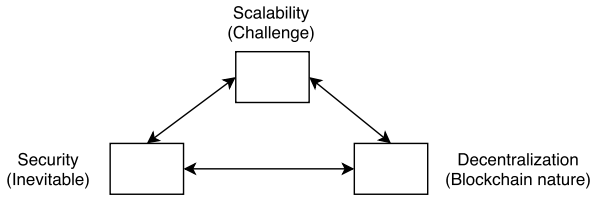

FIGURE 1. The scalability trilemma.

In this paper, we survey existing blockchain scalability solutions. These solutions can be classified into two categories: First layer solutions (e.g., sharding [20], bigger blocks [56], and DAG [80], [85], [86]) and second layer solutions (e.g., payment channels [36], [40], and side chains [35], [72]). In particular, we focus on solutions that use

The associate editor coordinating the review of this manuscript and approving it for publication was Roberto Nardone.

FIGURE 2. Taxonomy and comparison of blockchain scalability solutions.

the concept of sharding, given that it is one of the most promising solutions to the scalability problem. The basic idea behind sharding is to divide the network into subsets, called shards/committees. Throughout the paper, we will use the terms shard and committee interchangeably. Each committee will be working on a different set of transactions, rather than the entire network processing the same transactions. We present a taxonomy in which we classify sharding-based blockchain protocols based on committee formation and intra-committee consensus; we also analyze the security of these protocols by computing the failure probability for one committee and for each epoch (i.e., one sharding round). Furthermore, the paper presents a detailed comparison of existing scalability solutions, incorporating each of their advantages and disadvantages. Figure 2 shows a summary of our taxonomy and comparison of blockchain scalability solutions.

The contributions of this paper can be summarized as follows:

- We overview blockchain scalability and we provide a general taxonomy to classify existing blockchain scalability solutions;
- We survey and compare sharding-based blockchain protocols (more than the ones reported in [19]–[21]);

- We analyze the security of sharding-based blockchain protocols and we investigate the trade-off between security and performance (throughput) based on the failure probability and years to fail;
- We present a detailed comparison of existing solutions of scalability based on their performances (i.e., throughput, latency), advantages, and disadvantages.

The rest of this paper is organized as follows. Section II introduces blockchain scalability and presents definitions and notations used in the subsequent sections. Section III presents sharding first layer solution. Section IV presents other first layer scalability solutions. Section V presents second layer scalability solutions. Section VI presents a comparative analysis of scalability solutions. Section VII compares our paper with existing surveys. Section VIII concludes the paper.

II. SCALABILITY & NOTATIONS

In this section, we introduce blockchain scalability. Then, we present definitions and notations that are used in the rest of the paper.

A. SCALABILITY

The objective of existing blockchain scalability solutions is to process a high number of transactions per second (i.e., throughput) without sacrificing security and decentralization [45], [46]. Indeed, we can considerably increase the throughput, but we will lose in terms of decentralization (which is a key characteristic of blockchain) [45], [46]. To be more specific, several factors impact blockchain scalability; these factors include, in addition to the throughput, storage, cost, and latency:

- *Throughput:* It is the number of confirmed transactions per second. In public blockchains, the throughput is too small (e.g., up to 7 tx/s for bitcoin and up to 15 tx/s for Ethereum [17]) compared to other systems (e.g., up to 1700 tx/s for Visa and up to 193 tx/s for Paypal).
- *Storage:* If all transactions are recorded in blockchain, its size will considerably increase. Indeed, Figures 3, 4, and 5 show that as the total number of confirmed transactions per day for Bitcoin from 2019 to 2020 increases (Figure 3), the average block size in megabytes (MB) increases (Figure 4), yielding in an increase in the total size of the blockchain (Figure 5). The blockchain size of Bitcoin (BTC), Ethereum (ETH), and Litecoin (LTC) are 305.23 GB, 222.58 GB, and 28.45 GB, respectively [47]. This will create a great demand on storage and increase the time needed to download the blockchain.
- *Cost:* Once a transaction is confirmed, the user pays transaction fees to the miner (i.e., node that created the block where the transaction is included). Thus, it will be much cheaper for the user to conduct as many transactions as possible outside of the blockchain and then later record them as one transaction.
- *Latency (Aka Confirmation Time):* The time between submitting a transaction to the blockchain and the first confirmation of acceptance by the blockchain. As more users thus generate a greater amount of transactions, the verification time increases; each transaction requires peer-to-peer verification. For instance, Bitcoin is currently mining one block every 10 minutes, ensuring that the confirmation time is higher.

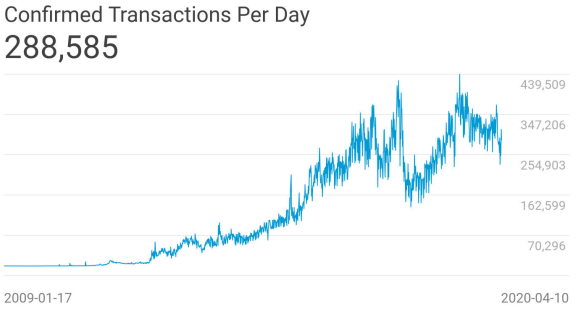

FIGURE 3. The total number of confirmed transactions per day for Bitcoin from 2009 to 2020 [48].

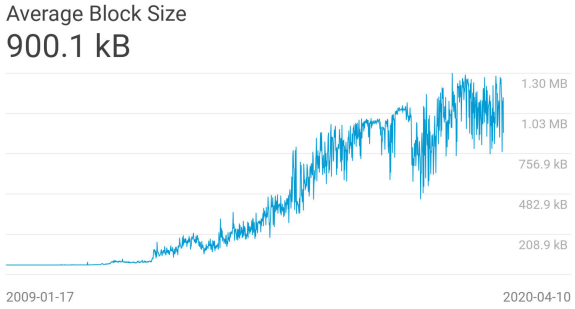

FIGURE 4. The average block size in MB for Bitcoin from 2009 to 2020 [48].

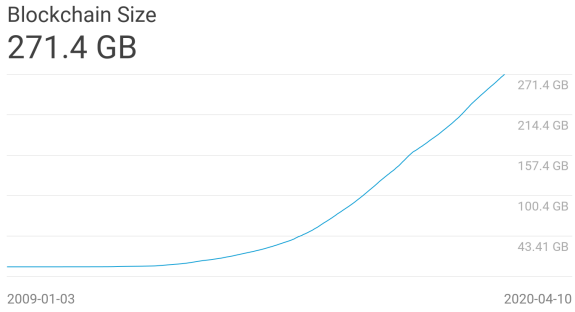

FIGURE 5. The total blockchain size minus the database indexes in megabytes (MB) for Bitcoin from 2009 to 2020 [48].

B. DEFINITIONS AND NOTATIONS

Table 1 shows the list of symbols and variables that are used to describe the security analysis of sharding protocols.

1) DEFINITIONS

Definition 1 (Failure Probability): The probability of the malicious nodes exceeding their limit in the netwok/commitee (i.e., maximum percentage of nodes/validators that can act in a malicious manner, such as, in the case of Elastico [4], where the limit is 25% of the nodes in the network).

TABLE 1. Notations.

Notation	Description
N	Total number of nodes
n	Committee size
M	Total number of malicious nodes
r	Committee resiliency
R	Total resiliency
η	Number of committees
p_c	Committee failure probability
$h(M;N;n;m)$	Hypergeometric distribution with parameters M, N and n
$H(M;N;n;m)$	Cumulative hypergeometric distribution with parameters M, N and n
$B(n;p;m)$	Cumulative binomial distribution with parameters n and p
B_w	Block weight
B_s	Block size in bytes
Ba_s	Block size minus witness data size in bytes

Definition 2 (Committee Resiliency): The maximum number of malicious nodes that the committee is able to contain while still being secure.

Definition 3 (Total Resiliency): The maximum number of malicious nodes that the whole network is able to contain while still being secure.

Definition 4 (Merkle Tree): A mechanism for hashing a large piece of data, which operates as shown below:

- Use a partition function to split the data, denoted by D, into chunks $D_1, \ldots, D_n$, such that D_i for $i \in \{1, 2, \ldots, n\}$ has a very smaller size compared with D where $D = \cup_{i=1}^{n} D_i$ and for all $i, j \in \{1, 2, \ldots, n\}$; such as $i \neq j$, we have $D_i \cap D_j = \varnothing$. Let us define this collection as $L_0(D)$;
- To determine $L_{i+1}(D)$ (starting from $i = 0$), split $L_i(D)$ into collections of fixed size using some deterministic algorithm, and return $L_{i+1}(D)$ the set of the hashes of the collections. Because there are fewer collections than elements (each collection has more than one element), the size of $L_{i+1}(D)$ will be smaller than the size of $L_i(D)$;
- Eventually, this leads to some $L_k(D)$, which contains only one element. We call this the root of the tree.

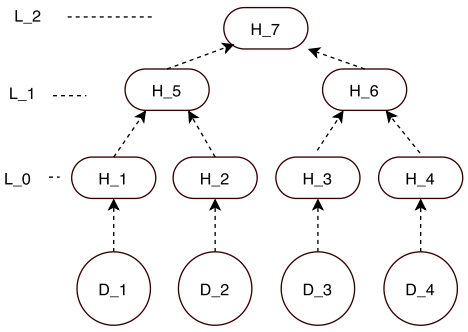

FIGURE 6. An example of a Merkle Tree.

First, let $H(D_1) = H_1$ (i.e., hash of the set of data D_1), $H(D_2) = H_2$, etc. Figure 6 shows an example of Merkle Tree; in this case, $L_0(D)$ includes H_1, H_2, H_3, and H_4, $L_1(D)$ includes $H_5 = H(H_1 + H_2)$ and $H_6 = H(H_3 + H_4)$, and $L_2(D)$ includes $H_7 = H(H_5 + H_6)$. In this case, $L_2(D)$ represents the root of tree.

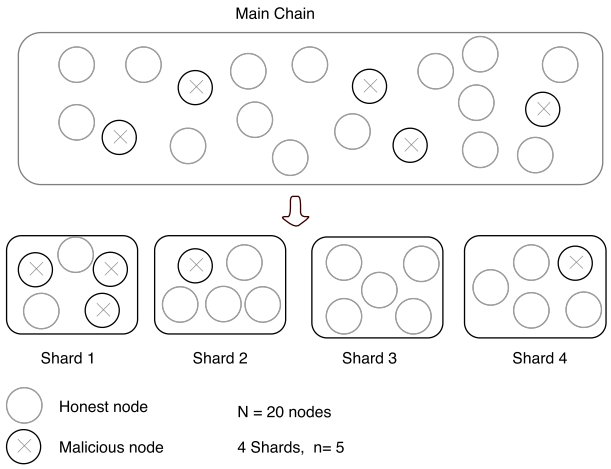

FIGURE 7. A simple example of a single shard takeover attack (shard 1 in this case).

III. FIRST LAYER SOLUTIONS: SHARDING

The leading solution currently discussed in the blockchain literature makes use of the concept of sharding [4]–[10], [12]–[14], [16]. The key idea behind sharding is to divide or split the network into subsets, called shards; each shard will be working on a different set of transactions, rather than the entire network processing the same transactions. This allows the network to scale with the numbers of shards, allowing the throughput and the storage to achieve high efficiency, yet potentially compromising the security. Indeed, for the blockchain to be secure, all shards need to satisfy the byzantine validator limit (aka, committee resiliency), which is the maximum percentage of malicious validators/nodes. In most networks, this limit is 33% average (e.g., Elastico [4], and OmniLedger [5]) of the validators; beyond that limit, a consensus instance is fundamentally insecure. The critical issue at hand is the fact that even if the whole network falls well under that limit, a single shard could be compromised. For instance, if we assume that a network with 25% malicious nodes is split evenly into 4 shards and more than 33% of malicious nodes in the network end up in one shard, it will be insecure. This is known as a *single shard takeover attack* (e.g., see Figure 7).

Figure 7 shows a scenario where there is a single shard takeover attack. The network contains 20 nodes with 25% of them malicious; there are 5 malicious nodes in the network and 3 of them are in shard 1. Therefore, the existence of 60% malicious nodes in shard 1 is larger than the byzantine limit (33%).

In the remainder of this section, we introduce a taxonomy in which we classify sharding-based blockchain protocols into three categories. We also analyze their security, and eventually discuss and compare them based on factors that include transaction model and consensus.

A. TAXONOMY OF SHARDING-BASED BLOCKCHAIN PROTOCOLS

Most sharding-based blockchain protocols in the literature use either *PoW* for committee formation and *PFT* for intra-committee consensus or *PoS* for committee formation and *PFT* for intra-committee consensus. The others use specific consensus algorithms. Thus, we classify sharding-based blockchain protocols into three categories: sharding-based Proof-of-Work and *PFT*, sharding-based Proof-of-Stake and *PFT*, and sharding-based on other consensus algorithms.

The following will present certain representative sharding-based blockchain protocols in each category.

► **Sharding-based Proof-of-Work and PFT**

Elastico: In 2016, Luu *et al.* [4] proposed the first sharding-based blockchain protocol, called Elastico, as a candidate for a public blockchain that tolerates byzantine adversaries. Elastico divides the network into multiple committees where each handles a separate set of transactions, called shard. Elastico uses *PoW* for committee formation and *PFT* for intra-committee consensus. The number of shards grows almost linearly with the size of the network. When the network reaches up to 1600 nodes, Elastico succeeds at increasing the throughput (e.g., up to 40 tx/s). However, it does have shortcomings. The first one is the division of each epoch (i.e., one sharding round; e.g., once a day) of Elastico can be influenced by malicious nodes. Moreover, the network can only tolerate up to 25% of malicious/faulty nodes (total resiliency) and 33% of malicious nodes in each committee (committee resiliency).

OmniLedger: Kokoris-Kogias *et al.* [5] proposed a sharding-based protocol called OmniLedger in order to correct some of Elastico's shortcomings. The protocol makes use of a bias resistant randomness protocol to ensure security. Just as Elastico, OmniLedger uses *PoW* for committee formation and *PFT* for intra-committee consensus. In addition, OmniLedger uses a Byzantine shard atomic commit (Atomix) to deal with cross-shard transactions. The OmniLedger consensus protocol uses a variant of ByzCoin [42] to handle and achieve faster transactions (e.g., up 500 tx/s when the network grows up to 1800 nodes). OmniLedger claims the same resiliency as Elastico for both total resiliency and committee resiliency.

RapidChain: In a more recent work, Zamani *et al.* [6] proposed a sharding-based public blockchain protocol by the name of RapidChain. It outperforms existing sharding algorithms (e.g., [4], [5]) in terms of throughput and security [6]. Cross-shard transactions in RapidChain rely on an inter-committee routing scheme which is based on the routing-algorithm of Kademlia [78]. Indeed, RapidChain can tolerate up to 33% of malicious/faulty nodes in the network, and 50% of malicious nodes in each committee. In addition, RapidChain claims a much higher throughput (e.g., up to 4220 tx/s when the network grows up to 1800 nodes).

► **Sharding-based Proof-of-stake and PFT**

Zilliqa: The team, at Zilliqa proposed a solution that would considerably increase the throughput [7]. Zilliqa's sharding design allows the network to process transactions in parallel and reach a high throughput. Zilliqa is expected to process about a thousand times the throughput of Ethereum [7]. However, this solution does possess shortcomings, including the fact that it does not divide the storage of blockchain data (i.e., state sharding; moreover, Zilliqa's sharding process is susceptible to a single-shard takeover attack [7]. Zilliqa claims the same local and global resiliency as Elastico and OmniLedger [7].

Harmony: Harmony [8] has been proposed as a way to fix some of Zilliqa's shortcomings, claiming that it is fully scalable; Harmony shards not only the network communication and transaction validation, like Zilliqa, but also shards the blockchain state. In addition, Harmony was able to prove that its sharding process ensures a high security, thanks to its distributed randomness generation process [8]. Harmony claims the same local and global resiliency as Zilliqa, Elastico, and OmniLedger [8].

Ethereum sharding 2.0: Ethereum sharding 2.0 is one of the most popular sharding-based blockchain protocols, consisting of three essential phases. The first is Beacon Chain, which manages all shards in the network; more specifically, it applies consensus rules, rewards and penalties to validators, and manages validators and their stakes. The second phase is Shard Chains, which enables parallel transactions. The final phase is State Execution, where the operations of the entire system are executed; it introduces the concept of "Execution Environments (EEs)", which provides a smart contract similar to Ethereum 1.0 [11], [12]. Ethereum decided to choose receipt paradigm in order to reckon cross-shard communication. In *receipt paradigm*, every transaction generates a receipt. These receipts will be stored on the beacon chain via *distributed shared memory*; this means that receipts can be seen by other shards, yet remain unable to be modified. Ethereum also uses the so-called *Casper*; a new *PoS* consensus that will replace the current *PoW* to implement Ethereum sharding 2.0 [12], [79].

► **Sharding-based on other consensus**

Logos: Logos has succeeded in increasing throughput with a low latency using a novel structure [10]. Technically, Logos uses *Axios* consensus, which is a delegated version of the *PBFT* algorithm inspired by ByzCoin [42], and OmniLedger [5]; the Axios consensus has been adapted to Logos's structure [10]. Each account on the Logos's network has an individual chain that tracks all of its transactions, allowing independent transactions to be processed in parallel. A main settlement chain provides universal synchronization of nodes as well as dynamic validator sets. Sharding adds a second dimension of parallelism to the network and mitigates the scalability issue. Transactions are validated and approved by a small number of delegates elected via a representative system, which reduces redundant operations

without compromising security. Furthermore, Logos's unique hybrid data structure allows shards to efficiently sync on summary data and greatly reduces the overhead imposed by cross-shard transactions; this is the key improvement of Logos over alternative sharding-based protocols [10].

Monoxide: Monoxide introduces a specific asynchronous consensus, called *Zones* (i.e., shards), which linearly scales-out blockchain with the number of zones, without compromising or weakening security, or risking decentralization. Monoxide uses the eventual atomicity and ensures the correctness of cross-shards/cross-zones transactions. It also proposes *Chu-ko-nu mining*, a novel proof-of-work scheme that ensures effective mining power in each zone at the same level as the entire network, rendering an attack on any individual zone as difficult as an attack on the full network. Indeed, Monoxide is the only sharding mechanism that supports the Nakamoto consensus protocol with PoW for intra-consensus. Finally, Monoxide claims that it is the only scalable blockchain system capable of implementing full sharding by splitting/partitioning the workload of all components of a blockchain system including transaction broadcasting, mining competition, chain storage, transaction execution and state representation [13].

Monoxide proposes *Chu-ko-nu mining*, a novel proof-of-work scheme. It ensures that effective mining power in each zone is at the same level as the entire network, making an attack on any individual zone as hard as that on the full network. Indeed, Monoxide is the only sharding mechanism that supports Nakamoto consensus protocol with *PoW* for intra-consensus. Finally, Monoxide claims that it is the only scalable blockchain system implementing full sharding by splitting/partitioning the workload of all components of a blockchain system including transaction broadcasting, mining competition, chain storage, transaction execution and state representation [13].

Table 2 summarizes and compares common characteristics of existing sharding-based blockchain protocols.

B. SECURITY ANALYSIS

This section is devoted to analyzing the security of sharding-based blockchain protocols; more specifically, we compute the number of years until failure, and investigate the trade-off between security and performance.

In sharding-based blockchain protocols, the process of assigning nodes to shards can be modeled by sampling without replacement since committees are disjointed, and thus do not overlap. In this case, hypergeometric distribution yields better estimates compared to that of binomial distribution [3], [49]. Thus, the formation of committees/shards (or the partition of the network into shards/committees) will be modeled by using hypergeometric distribution. Generally, we use binomial distribution when the sample is drawn with replacement [49]. However, there are many sharding-based blockchain protocols in the literature that use the binomial distribution such as Ethereum-sharding 2.0 [12], OmniLedger [5], and Dfinity [50], where the assumption is that $X \sim B(n; p)$ (i.e., X follows the binomial distribution with parameters n and p) where $p = \frac{K}{N}$ is the probability that a node is malicious. Thus, the failure probability of one committee with resiliency r using the cumulative binomial distribution can be expressed as follows:

$$P(X \geq nr) = \sum_{m=\lfloor nr \rfloor}^{n} \binom{n}{m} p^m (1-p)^{n-m}. \tag{1}$$

After calculating the failure probability of the first committee, they conclude the epoch failure probability by multiplying the failure probability of the first committee by the number of committees. However, there are recent contributions [2], [3], that propose the use of the hypergeometric distribution in order to overcome the limitation of the previous contributions.

The failure probability for a committee with resiliency r by using the cumulative hypergeometric distribution is

TABLE 2. Comparison of sharding-based blockchain protocols.

Protocol	Transaction Model	Consensus	Committee Resiliency	Total Resiliency	Network Synchrony	Throughput	Latency (second)	Smart Contract	Public Blockchain
Elastico [4]	UTXO	PoW, PBFT	33%	25%	Partial Sync.	40 tx/s [1]	800 s	×	✓
OmniLedger [5]	UTXO	Hybrid [a] (BFT, PoW)	33%	25%	Partial Sync.	3,500 tx/s [2]	800 s	×	✓
RapidChain [6]	UTXO	BFT, PoW	50%	33%	Partial Sync.	7,380 tx/s [3]	8.7 s	×	✓
Ostraka [9]	UTXO	PoW	50%	33%	N/A	400k tx/s [4]	N/A	×	✓
Zilliqa [7]	Account [b]	PoW, PBFT	33%	25%	Async.	N/A	N/A	✓	✓
Harmony [8]	Account	PoS, BFT	33%	25%	Sync.	N/A	N/A	✓	✓
Ethereum-sharding 2.0 [12]	Account	PoS, BFT	33%	33%	Partial Sync.	N/A	N/A	✓	✓
Dang et al. [14]	Account	BFT, TEE [c]	50%	30%	Partial Sync.	3,000 tx/s	N/A	✓	✓
Logos [10]	Account	Axios [d]	33%	33%	Partial Sync.	2,500 tx/s	< 3	✓	✓
Stegos [15]	UTXO	gPoS	N/A	N/A	N/A	N/A	N/A	✓	✓
Chainspace [16]	CSCoin	BFT	33%	25 %	Async.	350 tx/s [5]	N/A	✓	×
Monoxide [13]	Account	Zones	50%	50 %	Async.	11,694 tx/s [6]	N/A	×	✓
SSChain [98]	UTXO	PoW	N/A	25 %	N/A	6500 tx/s [7]	N/A	×	✓

✓ : has property; ×: does not have property; [1]: 100 nodes/committee. 16 committees; [2]: 600 nodes/committee. 3 committees; [3]: 250 nodes/committee. 16 committees; [4]: 64 committees, [5] 15 committees; [6]: 24 nodes/zone. 2,048 zones; [7]: 1800 nodes; [a]: Details on hybrid consensus can be found in [53]; [b]: Also known as balance model or account-based data model; [c]: Example of TEE is Intel SGX [54], that is used in [14]; [d]: A delegated Practical Byzantine Fault Tolerance (dPBFT) algorithm.

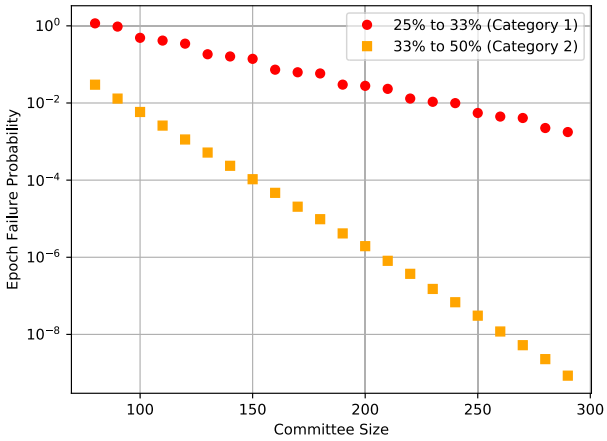

FIGURE 8. Log-scale plot of the probability of failure of one epoch (sharding round) versus the size of the committee for both categories 1 and 2.

expressed as follows:

$$H(M,N,n,n\,r) = \sum_{m=\lfloor nr \rfloor}^{n} \frac{\binom{M}{m}\binom{N-M}{n-m}}{\binom{N}{n}} \tag{2}$$

Similarly, they compute the epoch failure probability by multiplying the failure probability of the first committee by the number of committees.

It is possible to ignore the bootstrap probability (i.e., the probability that the committee election fails in the first epoch for each sharding-based blockchain protocol). The epoch failure probability can thus be expressed as follows:

$$\eta \times H(M,N,n,n\,r) = \eta \times \sum_{m=\lfloor nr \rfloor}^{n} \frac{\binom{M}{m}\binom{N-M}{n-m}}{\binom{N}{n}} \tag{3}$$

We classify sharding-based blockchain protocols, based on their resiliency, into two categories: (1) Category 1: It includes protocols with committee resiliency 33% and with a total resiliency of 25% (e.g., [5], [7]); and (2) Category 2: It includes protocols with a committee resiliency of 50% and with a total resiliency of 33% (e.g., [6], [9]).

Figure 8 illustrates the failure probability for one epoch when varying the size of the committee (80–300 nodes) in a network of 2000 nodes for both categories 1 and 2. We observe that the failure probability decreases when the size of the committee increases for both categories. We also observe that the epoch failure probability for category 2 decreases rapidly compared to category 1; this can be explained by the fact that category 2 has a higher resiliency than category 1.

Figure 9 shows the years to fail when varying the size of the committee (80–300 nodes) in a network of 2000 nodes for both categories 1 and 2. We observe that the years to fail increases when the size of the committee gets larger for both categories. We also observe that the years to fail for category 2 increases rapidly compared to category 1 thanks to its high resiliency.

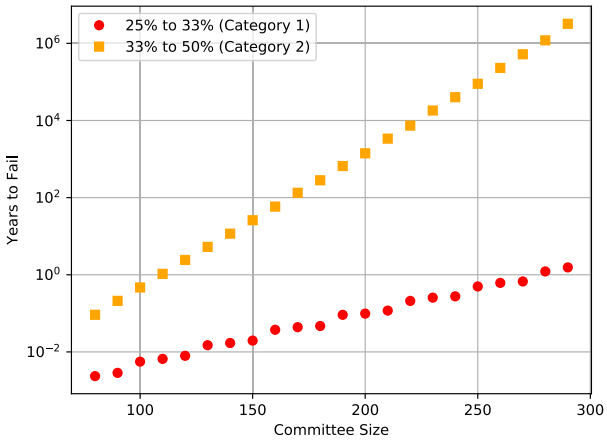

FIGURE 9. Log-scale plot of time to failure (in years) versus the size of the committee for both categories 1 and 2.

Now, based-on Figures 8 and 9, we observe that when the size of the committee gets larger, the epoch failure probability decreases and the years to fail increases. Thus, it is possible to conclude that there is a clear trade-off between security (i.e., years to fail) and performance (i.e., throughput; the smaller the committees' size, the larger the number of committees in the network, hence the bigger the throughout and vice versa). Indeed, the smaller committees' size leads to better throughput but can compromise security.

C. COMPARISON AND FUTURE DIRECTIONS

In this section, we provide a comparison among sharding-based blockchain protocols and we discuss future research directions and open issues.

1) SHARDING-BASED BLOCKCHAIN PROTOCOLS: A COMPARISON

Table 2 summarizes and compares key components of existing sharding-based blockchain protocols in the literature. More specifically, we compare these protocols in terms of performance (throughput and latency), resiliency, type of blockchain, consensus algorithms, the use of smart contract, and network synchrony (the level of coordination of all network components). Specifically, there are three levels of synchrony: synchronous, partially synchronous, and asynchronous [117] as well as transaction model. Numerous protocols use the "*Unspent Transaction Output*" (*UTXO*) transaction model (see Table 2). UTXO represents an unspent transaction output that can be spent as an input in a new transaction [51]. Other sharding-based protocols use *Account Model* transaction model; in this model, the balances of accounts are stored as part of the blockchain state. Both transaction models are used to maintain the balance and validate new generated transactions, thus keeping track of account balances in a single blockchain).

Each model transaction has its own advantages; generally, *UTXO* is more suitable in the context of sharding,

whereas *Account Model* achieves efficient storage usage economy, and functions better regarding smart contracts. Most sharding-based blockchain protocols have 25% as global/total resiliency and 33% as a committee resiliency, save for a few (e.g., the well-known RapidChain [6]), which has 33% as a total resiliency and 50% as a committee resiliency. As shown in Section III-B, high resiliency ensures high security; thus, RapidChain achieves a high level of security compared to Elastico and OmniLedger. Generally, sharding-based blockchain protocols achieve an acceptable number of transactions per second with a reasonable latency, not including, for example, Chainspace and Elastico. Table 2 shows that all sharding-based blockchain protocols are permissionless (aka, public blockchains) except Chainspace, which is a permissioned blockchain. We can conclude that sharding-based solutions are more suitable for public blockchains where the network size is large, such as in the thousands. Ostraka [9] achieves a high number of transactions. Unlike sharding-based blockchain protocols that split the network into shards, Ostraka [9] shards the nodes themselves and scales linearly with available resources. Ostraka [9] is not vulnerable to Denial-of-Service (DoS) attacks such as other sharding-based blockchain protocols. Indeed, in these systems, transactions are usually assigned to shards according to their *Tx-Hashs* (i.e., each transaction is uniquely identified by a cryptographic hash, called *Tx-Hash*). An attacker can more easily overwhelm a single shard with many transactions, rather than the entire system. Moreover, since each transaction may affect multiple shards, transactions in other shards can also be affected.

Table 2 shows also that sharding-based blockchain protocols use different consensus protocols. For example, OmniLedger uses an hybrid consensus that combines PoW and PBFT. Elastico and Chainspace use PBFT for intra-committee consensus whereas Monoxide uses PoW-based Chu-Ko-nu mining [13]. More details on the intra-committee consensus comparison of sharding-based blockchain protocols can be found in [19], [20].

In addition to the sharding-based blockchain protocols shown in Table 2, there exist lesser-known sharding solutions such as PolyShard [105], Aspen [99], RSCoin [100], SMChain [104], and Channels [103]. Aspen [99], is a sharding-based blockchain designed to scale with an increasing number of services without compromising the security. The permissioned blockchain, RSCoin [100], uses the sharding technique for the purpose of making cryptocurrencies under the authority of banks. PolyShard [105] is a polynomially coded sharding scheme that achieves efficiency of the storage, system throughput, as well as trust, thus enabling a truly scalable system. Channels [103] scales permissioned blockchains horizontally and with confidentiality. In Industrial Internet of Things (IIoT), platforms use sharding; for example, SMChain [104] is a hierarchical, secure, and scalable blockchain protocol in distributed industrial plants.

2) FUTURE DIRECTIONS AND OPEN ISSUES

Sharding-based blockchain protocols can face many types of attacks, such as *Sybil attacks* (when the attacker creates multiple fake identities) [101], and *double spending attacks* (when the attacker tries to spend a specific coin multiple times) [102]. Each sharding protocol has its own strategy to prevent these attacks (e.g., Zilliqa [7], detects double spending attacks by using the *nonce*, which counts the number of transactions sent from the sender's account).

The key challenge in each current sharding protocols is the dilemma of increasing performance without compromising security or decentralization. One of the ideas that requiring more investigation and research, in our opinion, is heterogeneous sharding; indeed, most existing solutions to this problem assume that validators/miners are homogeneous, save for the distinction of honest or malicious. However, validators are different in terms of communication bandwidth, computing power, and past behaviors. Thus, under random sharding protocols (e.g., Elastico [4] and OmniLedger [5]), less-competent validators (e.g., validators that have less computing power) hamper the performance (e.g., throughput) of the system. A solution that uses heterogeneous sharding is RepChain [106] which makes use of reputation to explicitly characterize heterogeneity among validators. RepChain achieves a high throughput, up to 15421 tx/s when the network size grows up to 1800 nodes and the size of the shards grows up to 100 nodes. Thus, this new kind of sharding design is very promising in terms of improving the throughput of sharding-based blockchain protocols.

While sharding-based solutions were able to demonstrate that they can considerably improve blockchain scalability, we believe that more formal studies and investigations are needed regarding a number of the operations of these protocols. The first operation is committee formation (how to split the network into shards); the second is intra-committee consensus (how to select the most suitable consensus that ensures security inside a committee); the third is cross-shard transactions (how to cross-verify transactions between shards).

Figures 8 and 9 show that the committee size impacts the network security (i.e., when the size of the committee gets larger, the failure probability decreases and the years to fail increases). Given the fact that in every sharding round, new participants may join or leave the network, it is important to change the size of the committee dynamically to preserve a predefined level of security (i.e., a predefined number of years to fail). An interesting research direction to investigate is the use of Machine Learning (ML) algorithms to dynamically determine the size of the committee. Furthermore, we believe that ML algorithms can be used to analyze and learn, using blockchain network data, to improve performance and security. For example, these algorithms can be used to detect the malicious nodes and punish them (e.g., by freezing their accounts) or at least split these nodes *uniformly* between shards to increase security.

Moreover, one of the critical open issues is the definition of a strong model to analyze the security of sharding-based blockchain protocols. One of the first attempts to define such a model was reported in [2], [3]. The authors [2], [3] used hypergeometric distribution to measure the security of sharding protocols by counting the number of years to fail; however, they assumed that the failure probability in the first committee was indicative of the failure probability in any other committee, since they calculated the failure probability of one epoch as the failure probability of the first committee times the number of committees (see Equation (3)). The problem with this assumption is that the samples are not independent when the sampling is done without replacement. This means that when we sample the first committee, the parametrization of the model changes (i.e., the number of network nodes and the number of malicious nodes). Thus, the failure probability of the second committee will be different from the first, the third will be different from the first and the second, and so on.

In addition, the inaccuracy of the calculations [2], [3] increases with the number of committees. We conclude that building a strong mathematical model, that takes into consideration the failure probability of each committee, is an interesting open research problem.

IV. FIRST LAYER SOLUTIONS: OTHERS

In this section, we present first layer solutions (aka, on-chain solutions) that do not include sharding protocols. Generally, on-chain solutions propose modifications to the blockchain protocols (typically requiring a hard fork of the blockchain), such as block size increase (e.g., [56]) and Directed Acyclic Graph (DAG) (e.g., Spectre [80], DLattice [85]).

A. BIGGER BLOCKS

The big block is a method that increases the maximum block size [56]. In blockchain networks, the blocks are created periodically; each block contains a list of transactions. The number of transactions is limited by the block size; thus, if the block size increases, the number of transactions that can be included in a block will increase leading to an increase in the throughput. However, a larger block size leads to higher block transmission delays; this may lead to unacceptable propagation delays of blocks. In 2016, Croman *et al.* [94], reported that given the current 10-minute block interval of Bitcoin, the maximum block size should not exceed 4 MB, which yields a maximum throughput of 27 tx/s. Numerous blockchain protocols implement this method, including Bitcoin-NG [95], Bitcoin Cash [96], Bitcoin Classic [97], MAST [59], and SegWit [58]. Big block solutions are very limited since big blocks cause higher block transmission delays of blocks. Thus, we can not significantly increase the throughput by using this method.

In the following, we briefly cover Bitcoin Cash [96], MAST [59], and SegWit [58].

1) BITCOIN CASH (BTH)

Bitcoin Cash [96], is a peer-to-peer electronic cash system. It is a permissionless, decentralized cryptocurrency. Due to the inherent scalability of Bitcoin, the Bitcoin community decided to split the bitcoin system into two protocols in 2017. These protocols were Bitcoin Cash (BCH) and the original Bitcoin (BTC). BCH increases the block size to 8 MB, in order to process more transactions than BTC [111]. The big block increases throughput and reduces transactions fees.

2) MERKELIZED ABSTRACT SYNTAX TREES (MAST)

MASTs combine Merkle Trees (Figure 6) and Abstract Syntax Trees (ASTs) [59], [108]. The Merkle tree is a data structure that can be used to efficiently verify the integrity of the stored data [108]. In Bitcoin, Merkle Trees are currently used to efficiently store the transaction history of the blockchain [59]. ASTs represent the syntactic structure of programs [59]. Figure 10 shows a simple structure of MAST; the root of the tree represents the entirety of the program, while all of the other nodes represent subprograms. Each path in the tree is a different execution branch that the program can take. Jeremy *et al.* [59] claim that by using MASTs in Bitcoin protocol [1], for a program of length n, a compression to O(log n) is to be expected.

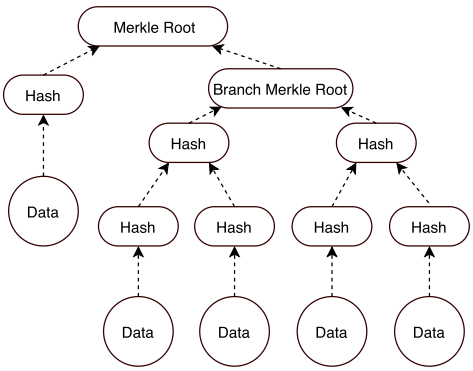

FIGURE 10. A simple structure of MAST.

3) SEGWIT WITNESS SOFT-FORK (SegWit)

SegWit [58] is a Bitcoin network upgrade that aims to solve Bitcoin's scalability and malleability. Malleability is the process of creating a new transaction's identifier (*id*) for an existing transaction. This network upgrade is a proposed change to how blocks are structured. Non-SegWit blocks (aka legacy blocks) have a total size of 1 MB whereas SegWit have large 4MB blocks. Indeed, SegWit is a block size increase. SegWit blocks consist of a base transaction block with a size of 1 Mb and an extended block with 3 MB (see Figure 11). While legacy blocks are measured in size, SegWit blocks are measured in weight. Block weight is a new concept introduced in SegWit, which is defined as follows:

$$B_w = 3 \times Ba_s + T_s, \tag{4}$$

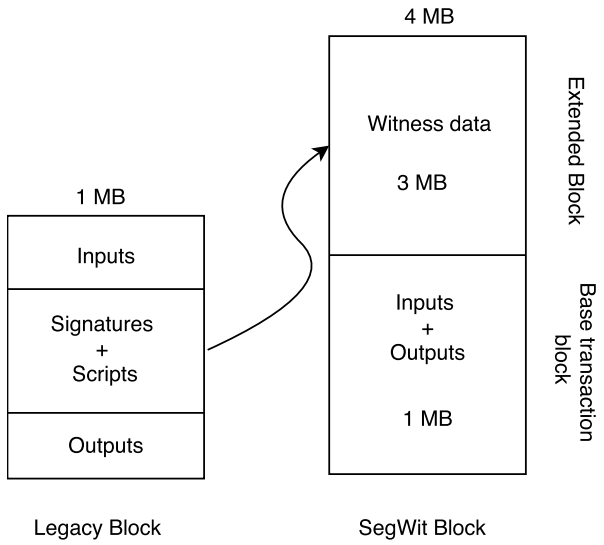

FIGURE 11. Legacy block vs. SegWit block.

Technically, SegWit blocks move the digital signature and other data known as "the witness" outside of the base transaction block and place them in the extended block (see Figure 11). Thus, the base transaction block includes only the information about the sender and the receiver; this optimizes the space of the base transaction block, since the witness data takes up to 65 % of the transaction size. Indeed, it allows for more transactions to fit inside the 1 MB base transaction block. Finally, SegWit achieves two major goals: (1) It moves the digital signature outside of the transaction block; thus, if someone changes the transaction *id* on the transaction, it will not affect the transaction *id*. This solves the transaction malleability issue; and (2) It reduces the size of the base transaction data, which allows more transactions to fit inside the 1 MB block. This allows an increase to the throughput. Even with all of these advantages, SegWit does possess some drawbacks: (1) Since SegWit is a block size increase, the SegWit protocol does not go far enough to solve the scalability issue; and (2) SegWit has caused several hard forks. The most well-known of these forks is Bitcoin Cash.

B. ALTERNATIVES TO PROOF-OF-WORK

Proof-of-Work is widely criticized for its high energy consumption. For instance, a single Bitcoin transaction consumes about 729 KWh of electricity, which can power 24 U.S. households for a day [91]. Figure 12 illustrates Bitcoin energy consumption relative to several countries. More specifically, Figure 12 shows that Bitcoin consumes more than the Czech Republic. Another comparative study, by the University of Cambridge, claims that Bitcoin consumes about 80.04 TWh of electricity per year, more than the total electricity consumption of several countries, such as Chile (73.22 TWh per year) and the Philippines (78.30 TWh per year) [92]. In addition, PoW-based blockchain protocols suffer from its low throughput (e.g., Bitcoin processes up to 7 tx/s). In response, the blockchain community has proposed various consensus alternative to *PoW*, in order to mitigate these issues, including Proof-of-Stake (PoS), Delegated Proof-of-Stake (DPoS), Stellar Consensus Protocol (SCP), Proof of Authority (PoA), Proof of Elapsed Time (PoET), and Proof of Retrievability (PoR). Interested readers are referred to [89] and [93] for comprehensive surveys of distributed consensus for Blockchain Networks. In this section, we present two solutions to the scalability issue, which use alternatives consensus protocols to Proof-of-Work: (1) EOS.IO, which uses the Delegated Proof-of-Stake (DPoS) consensus; and (2) Stellar, which uses a specific consensus, called Stellar Consensus Protocol (SCP). The reasoning behind choosing these two protocols is that EOS can support a high number of transactions up to millions and Stellar can help in the conversion of fiat currency into cryptocurrency. These two features are rare in other blockchain protocols.

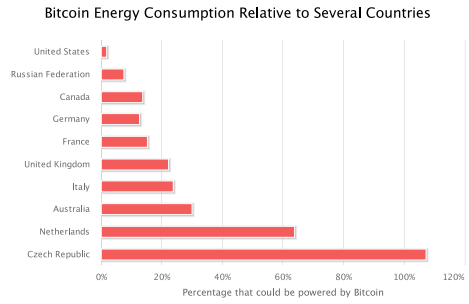

FIGURE 12. Bitcoin energy consumption relative to Several countries. Source: BitcoinEnergyConsumption.com.

1) EOS.IO

EOS.IO is an open-source blockchain architecture designed to enable vertical scaling (e.g., explore advanced database technologies to improve flexibility and throughput) and horizontal scaling (e.g., explore smart contract parallel execution) of decentralized applications [62]. Technically, EOS.IO uses Byzantine Fault Tolerant (BFT) and Delegated Proof-of-Stake consensus algorithms. More specifically, EOS.IO adds asynchronous Byzantine Fault Tolerance (aBFT) for the purpose of achieving a fast irreversibility. Indeed, the aBFT algorithm provides 100% confirmation of irreversibility within just one second [55].

Finally, EOS.IO claims that it can support millions of users, with no transaction fees, a high throughput (it may scale to millions of transactions per second), and a very low latency. However, in EOS.IO, only 15 or more producers out of the 21 producers can validate a block [55], [62]. This limited number of producers in EOS.IO results in losses in terms of decentralization. Furthermore, EOS.IO suffers from bot accounts. Yuheng *et al.* [109], reported that about 30 % of the accounts in the EOS.IO platform are bot accounts and are responsible for a number of real-world attacks (up to 301 attack accounts). Yuheng *et al.* [109] identified that about 80 attack accounts have been confirmed by DApp teams, causing the loss of 828,824 EOS tokens, which is about 2.6 million USD.

2) STELLAR

Stellar is an open-source, distributed, blockchain-based payments infrastructure [64]. Stellar helps in the optimum conversion of fiat currency into cryptocurrency, XLM, in order to enable fast cross-border payments between different currencies at extremely reduced rates between people, payment systems and financial organizations. Stellar uses a consensus algorithm, called *Stellar Consensus Protocol*, a construction for Federated Byzantine Agreement (FBA) consensus [63], [64]. Stellar is a global payment system; anyone in the world can become a Stellar user. On Stellar, users can interact directly with the world market [64]. The main feature of this infrastructure is the tethering token to a traditional asset like the US dollar, unlike many distributed blockchain systems. Stellar can handle and deal with every currency in the world (e.g., dollars and euros), not just cryptocurrencies [64].

C. DIRECTED ACYCLIC GRAPH (DAG)

DAG is another blockchain structure that differs from traditional blockchains. It is a network of individual transactions linked to multiple other transactions. While blockchain is a linked list of blocks, *DAG* is a tree, branching out from one transaction to another, and so on. Figure 13 illustrates the difference in terms of structure between *DAG*-based blockchains and traditional blockchains. In the literature, there are many blockchain protocols that use *DAG*, such as Spectre [80], IOTA [81], Phanton [83], DLattice [85], CoDAG [86], Nano [87], and XDAG [88]. In this paper, we give an overview of IOTA [81], Spectre [80], DLattice [85], and CoDAG [86]. Further details and a comparative analysis of *DAG*-based blockchains can be found in [84].

1) IOTA

IOTA is the most popular *DAG*-based blockchain protocol [81]. IOTA is a scalable, public distributed ledger, designed specifically for Internet of Things (IoT). The core feature of IOTA is the Tangle technology, which is *DAG* adapted for decentralized information storage (i.e., adapted for storing immutable and transparent data/transactions in a decentralized network) [82]. Tangle also offers the required features to establish machine-to-machine micro-payment protocols [82]. All nodes in an IOTA network store a copy of the Tangle and reach a consensus on its content. Furthermore, IOTA uses quantum-robust, one-time signatures to stop attackers that use quantum computers, and thus ensures security [81].

2) SPECTRE

In 2017, Sompolinsky *et al.* [80], proposed Spectre, a fast and scalable DAG-based public blockchain protocol. Spectre relies on a data structure that generalizes Nakamoto's blockchain into a *DAG* or a *block DAG*. Spectre is a *PoW*-based protocol that can process a high throughput of transactions and maintain fast confirmation times while remaining secure from attackers. Spectre claims that it is resilient to attackers with up to 50% of the computational power at any throughput [80]. Spectre is designed only for payments, and is not suitable for smart contracts [80].

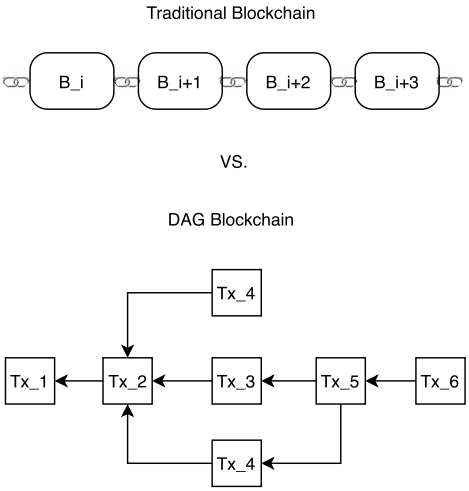

FIGURE 13. Traditional blockchain versus DAG-based blockchain.

3) DLattice

In a recent work, Zhou *et al.* [85] proposed DLattice, a public blockchain protocol with a novel double-*DAG* architecture. DLattice uses a new protocol, called *DPoS-BA-DAG* (PANDA), to reach a consensus among users. In DLattice, each account has its own Account-*DAG* and all accounts make up a greater Node-DAG structure. DLattice also parallelizes the growth of each account's Account-DAG; an account is not influenced by other accounts' irrelevant transactions. In addition, DLattice introduces a process of data tokenization based on its own structure, including data assembling, data anchoring, and data authorization.

4) CoDAG-BASED IIoT

In the Industrial Internet of Things, Cui *et al.* [86] proposed an efficient and secure blockchain protocol called CoDAG. It is based on compacted Directed Acyclic Graph, and claims that it is resistant against two attack strategies: (1) Adversary builds an alternate channel; and (2) Adversary competes in the original channel [86]. CoDAG was able to prove that it is resistant to these two attacks. Moreover, CoDAG proposes algorithms to maintain and secure the network. As a result, CoDAG scales with width and blocksize; indeed, when the blocksize is 2 MB and width is 15, CoDAG achieves a throughput up to 1151 tx/s, which is about 164 × Bitcoin's throughput and 77 × Ethererum's throughput. Table 3 shows a comparison between Bitcoin and DAG-based blockchains in terms of throughput, latency, blockchain structure, consensus, the ability to support smart contracts, application (i.e., designed only for payment or can have multiple uses), and decentralization.

TABLE 3. Bitcoin vs. DAG-based blockchains.

Protocol	Throughput	Latency	Structure	Consensus	Decentralized	Smart Contract	Application	Transaction Cost
Bitcoin [1]	7 tx/s	10 minutes	List of blocks	PoW	✓	○	Arbitrary	Yes
IOTA [81]	N/A	1-60 s	Tx DAG	PoW	×	●	Payment	No
Spectre [80]	N/A	N/A	Block DAG	PoW	✓	○	Payment	No
DLattice [85]	∼ 1200 tx/s	10 s	Double-DAG	PANDA	✓	○	Arbitrary	No
CoDAG [86]	1151 tx/s	N/A	Block DAG	N/A	✓	○	Payment	No

✓: has property; ×: does not have property; ●: supports smart contract; ○: does not support smart contracts.

V. SECOND LAYER

In this section, we present well-known second layer solutions (aka, off-chain solutions), which propose mechanisms that are implemented outside of the blockchain. They process certain transactions (e.g., micro-payment transactions) outside of the blockchain, and only record important transactions (e.g., final balances) on the blockchain. Second layer scalability solutions can be classified into two classes: sidechains (e.g., Plasma [35], RootStock [72]) and payment channels (e.g., Lightning Network [36], Raiden Network [40]).

A. SIDECHAINS OR CHILDCHAINS

A sidechain is a separate blockchain. However, it is not a standalone blockchain, as it is pegged in some way to the main chain. The main chain and the sidechain are interoperable; indeed, assets can move freely from main chain to sidechain and vice versa. We present two well-known sidechains: Bitcoin RootStock [72] and Ethereum Plasma [35].

1) BITCOIN RootStock

RootStock (aka, RSK) is an open-source smart contract platform pegged to Bitcoin with a two-way peg [72]. It allows Bitcoin miners to participate in the smart contract revolution by rewarding them via merge-mining. RSK is an innovative design (i.e., hybrid federated sidechain and a merge-mining model) that enables higher scalability and reduced transaction costs, while also increasing transaction throughput by porting decentralized applications (DApps) to it. Furthermore, RSK is the first Bitcoin sidechain that provides smart contract, compatible with Ethereum's; hence, it provides Ethereum users and companies a new compatible platform to deploy their solutions using Bitcoin as the native currency, relying on the Bitcoin mining infrastructure for its security. Finally, RSK enables developers around the world to create personal and corporate decentralized applications with a high level of security and low transaction cost [72]. However, RSK has some drawbacks: (1) It requires users to deposit some Bitcoins before performing their transactions; and (2) Since RSK is based on PoW, by supporting SHA-256D merged mining, it consumes high energy.

2) PLASMA

Plasma [35], is designed specifically to address scaling off-chain protocols. It is a technique to conduct off-chain transactions in a highly scalable way. The basic idea behind Plasma is to have many chains, called child chains, pointing back to the parent chain. Child chains can further spawn into more sub-chains, creating blockchains within blockchains (see Fig. 9). These chains can run as independent blockchains; intermittent updates are performed to the parent chains (if needed). They can have their own consensus mechanisms. There are numerous implementations of Plasma, including: Minimal Viable Plasma (MVP) [65], More Viable Plasma (MoreVP) [66], Plasma Cash [67], and Plasma Debit [68], which is an extension to Plasma Cash. More specifically, Plasma is a framework for building scalable, decentralized applications based on the concept of MapReduce, a well-known framework used to process big data sets [73].

Technically, the Plasma structure is built through the use of smart contracts [74], and Merkle Trees [75], [76], enabling the creation of an unlimited number of child chains. In Plasma, more chains can be created, thus building a tree-like structure. Essentially, each Plasma child chain is a customizable smart contract that can be designed to work in a singular way, serving different needs. This means that the chains can coexist and operate independently. Eventually, Plasma will make it possible for businesses and companies to implement scalable solutions in various ways, according to their specific context and needs. The communication between the parent chain and the child chains are secured by fraud proofs [77]; the parent chain is responsible for keeping the network secure and for punishing malicious actors/miners. Finally, Plasma succeeds in increasing scalability to reach transactions in the potential range of 1000. Nevertheless, it still contains many limitations. These include (1) Visualizing Plasma chains with full EVM (Ethereum Virtual Machine) capabilities, which is not the correct approach due to the nature of the ownership in Ethereum smart contracts (solidity smart contracts); (2) The confirmation signatures outlined in [35], are also limiting. They require that both parties send an acknowledgement transaction in order to ensure finality; and (3) Long waiting periods (7-14 days) for users who wish to withdraw their funds.

B. PAYMENT CHANNELS

Payment channels allow two participants to send a nearly-unlimited number of payments off-chain. Only two transactions are required for the establishment of payment channels; creating smart contract and funding it opens the channel, while re-claiming the funds closes it [114], [115].

Payment channels serve the same purpose as sidechains; however, they are fundamentally very different. Like sidechains, they push transactions off of the main chain to

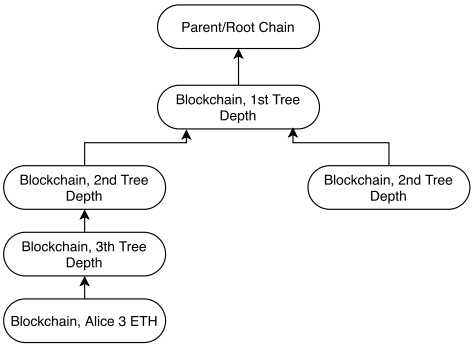

FIGURE 14. An Ethereum example of Plasma. Plasma composes blockchains within blockchains (i.e. blockchains in a tree). This example has depth up to 3 chains. Block commitments flow down and exits can be submitted to any parent chain, ultimately being committed to the root blockchain.

prevent blockchain from being bloated. Unlike sidechains, however, they do not require a separate blockchain for the execution. Technically, a payment channel uses a smart contract to enable users to transact without publishing their transactions to the blockchain. For instance, Alice can set up a payment channel with Bob in which she blocks/escrows 5 tokens for a period of time. Now, Alice can send signed transactions to Bob from the escrow amount, and Bob can validate them privately in a secure manner (via a smart contract) without mediation on the main chain. If Bob or Alice want to close the payment channel at any point, they can broadcast the most recent signed transaction message to the main chain in order to finalize their transfer of funds [90]. In this section, we present some of the mainstream payment channels, which include Lightning Network [36], Raiden Network [40], μRaiden [69] and Trinity [70].

1) LIGHTNING NETWORK

Bitcoin's Lightning Network is a typical off-chain solution [36]. The Lightning Network is a decentralized scalable system for instant, high-volume micropayments that removes the risk of delegating custody of funds to trusted third parties. Technically, Lightning uses smart contract functionality in the blockchain to enable instant payments across a network of participants. The Lightning network provides many advantages: (1) Instant Payments; the payment speed measured in milliseconds to seconds; (2) High throughput: the lightning network achieves a high number of transactions which can grow up of millions to billions per second across the network; and (3) Low cost: Lightning Network allows for exceptionally low fees thanks to the off-blockchain transactions. In brief, the inception of Lighting Network created the groundwork for a throughput of nearly unlimited number of transactions per second with very low fees. Lightning Network is a very promising off-chain solution to scale blockchain, but not without some drawbacks and open problems. The first problem is that Lighting scales transactions and not users. It can scale the number of transactions done by a particular group quite well, but enlarging the group takes more space for on-chain transactions. The second issue is that the network provides transactions that are less secure than Bitcoin. Finally, Lightning only works for Bitcoin's micropayments.

2) RAIDEN NETWORK

Raiden Network is a fast, cheap, and scalable network; its main functionality is a token transfer in Ethereum [40]. The Raiden Network is an off-chain scaling solution, enabling near-instant, low-fee and scalable payments for Ethereum [40]. It is complementary to the Ethereum blockchain, and works with any ERC20 compatible token. ERC20 is a token standard, which describes the functions and events that an Ethereum token contract has to implement. Technically, token is created by a smart contract, most often on the Ethereum blockchain. Raiden Network transfer fees are lower than on-chain transaction fees; instead of paying for a global consensus, you only pay for forwarding peer-to-peer consensus [40]. As Lightning Network, Raiden Network does have some drawbacks and open issues; these include the fact that Raiden transfers require some of your tokens to be locked up (deposited) in a smart contract for the lifetime of the payment channel, and that when multiple channels likely open at the same time, payment channel deposits are expected to be comparatively small, making it difficult to transfer large amounts of tokens over the network of channels.

3) μRaiden

μRaiden (Micro Raiden) is a payment channel framework for frequent, fast, and free ERC20 token-based micropayments between two parties. It comes as a set of open source libraries, documentation, and code examples for multiple use cases. Whereas its parent network, Raiden, aims for multihop transfers via a network of bidirectional payment channels, μRaiden can already enable unidirectional payment channels [69]. μRaiden relies on Ethereum's own security; only the sender and receiver, identified by their private keys, are able to interact with their channel, both on-chain and off-chain. μRaiden transfers are free. Only opening and closing payment channels incur standard Ethereum fees. μRaiden is a promising solution for Ethereum's micropayments, but not without some drawbacks. For example, μRaiden does not support multihop transfer fees, and therefore only allows tokens to be sent unidirectionally to predetermined receivers [69].

4) TRINITY

Trinity is a universal off-chain solution [70], adopting state channel technology as an off-chain scaling solution for Neo [71]. Trinity provides a series of solutions, such as protocol layer, free basic services, and incentives for value-added service providers. More specifically, Trinity is applicable to the blockchain transfer of Neo utox and NEP-5 standard tokens. Trinity achieves scalability and privacy protection of Neo assets through state channel technology [70], [71]. Technically, Trinity consists of four layers; Channel Service Layer (CSL), Channel Network Layer (CNL), State Channel

Layer (SCL), and Block Layer (BL). In addition to state channel technology, Trinity uses smart contract and implements Proof-of-Assets (PoA) consensus [70], [71].

Table 4 shows a comparison between blockchain payment systems and conventional payment systems (e.g., PayPal and MasterCard) in terms of transaction fees and confirmation time. PayPal has a payment transaction fee of 5 % + $ 0.05 [112]; this incurs, for example, a total fee of $ 0.1 for payment transfers of $1. MasterCard has a payment transaction fee of 0.19% + 0.53 [116]. Payment transfers of $1 would incur a total cost of $0.532. Bitcoin includes transactions into blocks every ten minutes. Payments are widely considered to be secure on Bitcoin after the confirmation of six blocks, which takes about one hour. The Stellar-based operation fee is 10^{-5} XLM [63], which is about $\$\ 6 \times 10^{-7}$. Lightning Network payment transactions are free. Only the opening and closing of payment channels incur standard fees. These fees are differents for an off-chain solution to another (e.g, Raiden Network requires $0.05 to open the channel and the same amount to close it). All the fees are converted into US dollars to facilitate the comparison between the different payment systems.

TABLE 4. A comparison of transaction fees and time between conventional payment systems (PayPal and MasterCard) and blockchain payment systems.

Payment System	Transactions Fees	Confirmation Time
PayPal [112]	5 % + $ 0.05 [a]	Few hours to several days
MasterCard [116]	0.19% + 0.53 [b]	N/A
Bitcoin [1]	$ 0.16	About one hour
Lightning Network [36]	$ 0 [c]	Milliseconds to seconds
Raiden Network [40]	$ 0 [c]	Sub-seconds
μRaiden [69]	$ 0 [c]	N/A
Stellar [63]	$\$\ 6 \times 10^{-7}$	Few seconds
Trinity [70]	N/A	Real-time payments

[a]: 5% of the paid amount plus a fixed amount of $ 0.05; [b]: 0.19% of the paid amount plus a fixed amount of $ 0.53; [c]: Direct payment; this means, without including open and close channels standard fees.

VI. ANALYSIS AND COMPARISON OF EACH CATEGORY

In this section, we compare the scalability solutions covered in this survey, in terms of performance, the type of applications they support, security (double spending), decentralization, and their advantages and disadvantages. Tables 5 and 6 compare some common characteristics of each solution of the scalability of the blockchain. Table 5 shows the advantages and disadvantages of each solution, whereas Table 6 compares these solutions in terms of throughput, latency, decentralization and cost, as well as whether they are designed for payment, and vulnerable to double spending. Finally, their advantages and disadvantages are illustrated. The objective of the scalability is to process a high number of transactions per second in a rapid and cheap fashion, without sacrificing security and the decentralization of the network. First layer solutions struggle to achieve this objective; most of these solutions increase the throughput by putting a lot of transactions in a block and thus increasing the interval time. Alternatives to *PoW*, such as *DPoS*, also succeed in solving scalability bottlenecks. Even though they can process a high number of transactions they lose in terms of security or decentralization. For example EOS, which uses DPoS, only has 21 nodes that are involved in the validation and creation of blocks; indeed, a consensus between only 15 of the 21 validators is sufficient to add a block to the blockchain. One of most promising first layer solutions of scalability is sharding, which solves throughput and latency problems at the same time. However, sharding suffers from the security problem, due to its 1% attack (i.e., the network is compromised if only one shard is compromised); a good security analysis is required to implement a sharding-based blockchain protocol. DAG-based blockchains increase the throughput with no transaction cost. Unlike traditional blockchains, such as Bitcoin, DAG-based blockchains do not need miners; they are resistant to attacks from Quantum computers. However, DAG-based blockchains have some drawbacks. They are vulnerable to attacks such as double spending, and since many DAG-based blockchains (such as IOTA) use a statistical analysis for transaction confirmation (e.g., Monte Carlo simulations), there is no analysis to determine the number of sample simulations that are necessary for a given transaction confirmation. Second layer solutions also succeed in mitigating the scalability issues of the blockchain. Payment channels solve throughput, latency and cost issues because transactions are handled outside of the blockchain; however, they suffer a number of limitations that include: (1) If two participants need to exchange assets between each other, payment channels require both participants to be online during the same transactions; (2) They require tokens to be locked in the blockchain, before executing transactions; and (3) They cannot be used for arbitrary applications. On the other hand, sidechains allow a reduction of transaction costs and increase transaction throughput. Unlike payment channels, sidechains do not require participants to be online to process transactions. In a broader sense, the main advantages of using second layer solutions is that the main chain (i.e., first layer) does not need to go through any structural changes because the second layer is added as an extra layer. Thus, second layer solutions have the potential to achieve high throughput without sacrificing network security. However, second layer solutions introduce new security risks and challenges (e.g., Plasma) or have a limited applicability (e.g., payment channels). This being said, Blockchain developers are encouraged to move as much work as possible from the main chain to the second layer for best performance and cost.

Finally, an interesting idea is to combine first and second layer solutions at the same time, or to combine two or more first layer solutions for the purpose of solving/mitigating the scalability issue. For example, Logos [10], aims to implement Lightning Network with sharding.

VII. COMPARISON BETWEEN THIS PAPER AND EXISTING SURVEYS

To the best of our knowledge, there is no thorough literature review contribution covering the same concerns as our paper. There exist a few related surveys in the open literature

TABLE 5. Comparison among different approaches solutions of scalability based on their advantages and disadvantages.

Layer	Solutions	Advantages	Disadvantages
First Layer	Sharding [4]–[10], [12]–[16]	• Parallel processing: Each node does not have to process all transactions in the system but only those in its shard.	• The network is compromised if only one shard is compromised (i.e., 1% attack).
	SegWit [58]	• Fixes transaction malleability issue; • Raises the block size; thus, increases the throughput; • Reduces UTXO growth; • Linearly scales the signature-hashing (i.e., solves the quadratic hashing problem).	• Causes hard forks on Bitcoin; • Leads to the problem of centralization.
	EOS.IO [55]	• It can allegedly handle millions of users with free usage; • It can handle millions of transactions per second.	• Only 15-21 producers can validate the block; • Vulnerable to bot accounts.
	MAST [76]	• Strong privacy; • Reduces the size of data; allows the reduction of the size of transactions.	• Does not enable smart contracts on Bitcoin.
	Big Block [56]	• Increases the size of the block; thus, increases the throughput; • Reduces transactions cost.	• Increases confirmation time; • Leads to the problem of centralization.
	Stellar [63], [64]	• Transactions are nearly free and can handle several currencies; • Tethering token to a traditional asset USD.	• Payment channel only.
	DAG [80], [81], [85], [86].	• High throughput with a fast confirmation time; • Removes the need for miners .	• Vulnerable to attacks such as double spending.
Second Layer	Plasma [35]	• Lower transaction costs and faster operations; • Plasma helps to ensure security (e.g., detects double spending attacks); • It does not require all participants to be online, unlike Lightning Network.	• The confirmation signatures are limiting; • Long waiting periods (7-14 days) for users who want to withdraw their funds; • Complex implementation.
	RSK [72]	• Reduces transaction costs and increases throughput; • High level of security.	• It requires users to lock up some of their bitcoins; • High energy consumption.
	Lightning Network [36]	• Transactions are free; • Instant (i.e., no waiting time; very low latency).	• Payment channel only; • It requires the two parties to be online at the same time; • Limited to Bitcoin transaction.
	Raiden Network [40]	• Fast, transactions are free, and supports multihop transfer fees; • Supports all ERC20 tokens, unlike Lightning Network; • Allows unlimited, bidirectional transfers between two participants.	• Like Lightning Network, it works on; payment channel only; • Requires tokens to be locked up in a smart contract for the lifetime of the payment channel.
	μRaiden [69]	• Fast and free ERC20 token; only opening and closing payment channels incur standard Ethereum gas fees; • Many-to-one unidirectional payment channel protocol.	• Like Lightning Network; payment channel only; • μRaiden only allows to send tokens unidirectionally.
	Trinity [70], [71]	• Low transaction fees with real time payments; • It allows multiple transactions to be made off-chain with exceptional speed and settled on-chain to ensure security; • It achieves decentralization by broadcasting the result to the entire network.	• Payment channel only.

[18]–[21]. Kim *et al.* [18] have categorized blockchain scalability solutions into: (1) On-chain solutions; (2) Off-chain solutions; (3) Side-chain solutions; (4) Child-chain solutions; and (5) Inter-chain solutions. However, side-chain, child-chain, and inter-chain solutions can be considered as off-chain solutions because they all perform transactions outside the main chain. Kim *et al.* [18] have provided an analysis of existing solutions; however, the analysis did not go deeper. Indeed, they neither covered nor compared sharding-based blockchain protocols and DAG-based blockchain protocols. Our work provides a different taxonomy, consisting of two layers: (1) First layer solutions,

TABLE 6. Comparison of the blockchain scalability solutions based on their performances, cost, application, vulnerable to double spending, and decentralization.

Category	Solution	Throughput	Latency	Cost	Application	Double Spending	Decentralization
First Layer	Sharding	Good	Low	Low	Arbitrary	Invulnerable	(✓, ×)
	SegWit	Medium	N/A	Low	Arbitrary	Invulnerable	✓
	EOS.IO	Excellent	Low	Low	Arbitrary	Invulnerable	✓
	MAST	Good	Low	Low	Arbitrary	Invulnerable	✓
	Big Block	Good	Low	Low	Arbitrary	Invulnerable	✓
	Stellar	Good	Low	Low	Payment	Invulnerable	✓
	DAG	Good	Low	Low	Arbitrary	Vulnerable	(✓, ×)
Second Layer	Plasma	Excellent	N/A	Low	Arbitrary	Invulnerable	✓
	RSK	Excellent	Low	Low	Arbitrary	Invulnerable	✓
	Lightning Network	Excellent	Low	Low	Payment	Invulnerable	✓
	Raiden Network	Excellent	Low	N/A	Payment	Invulnerable	✓
	μRaiden	Excellent	Low	N/A	Payment	Invulnerable	✓
	Trinity	Excellent	Low	Low	Payment	Invulnerable	✓

✓: has property; ×: does not have property; (✓, ×): can either have property or not, depending on each candidate (see Tables 2 and 3).

TABLE 7. Comparison between this work and existing surveys.

Reference	Layer 1 (On-Chain Solutions)	Layer 2 (Off-chain Solutions)	Sharding Taxonomy	Security Analysis of Sharding-Based Blockchain Protocols	Detailed comparison among scalability solutions
This work	●	●	✓	✓	✓
Kim et al. [18]	◑	◑	×	×	×
Yu et al. [19]	●	○	○	○	×
Zhou et al. [21]	●	●	×	×	×
SoK [20]	●	○	○	○	×

●: Surveys the category; ◑: Partially surveys the category; ○: Does not survey the category.

which include all solutions that propose modifications within the blockchain structure (e.g., sharding-based blockchain protocols and DAG-based blockchain protocols); and (2) Second layer solutions, which propose mechanisms that are implemented outside the blockchain (e.g., μRaiden [69], Bitcoin RootStock [72]). This paper also provides a detailed comparison among existing scalability solutions. Yu *et al.* [19] present a systematic and comprehensive survey focusing on sharding in blockchain. Guangsheng *et al.* present six well-known sharding-based blockchain protocols: Elastico [4], OmniLedger [5], Monoxide [13], RapidChain [6], Chainspace [16], and Ethereum sharding 2.0 [11]. These protocols are characterized in terms of intra-consensus-safety, cross-shard-atomicity, and general improvements. However, many sharding-based blockchain protocols are missing in their characterization and the scope of the survey [19], is limited to the sharding solution. Here, there is an exploration of more sharding-based blockchain protocols as well as a proposed taxonomy of the sharding-based blockchain protocols based on committee formation and intra-committee consensus. Wang *et al.* [20] provide a systematic and comprehensive review of blockchain sharding solutions. They present a general design of sharding-based blockchain protocols and discuss key design challenges in each component. These components are: (1) consensus protocols; (2) epoch randomness; (3) cross-shard transactions; and (4) epoch reconfiguration. In addition, Wang *et al.* [20] compare state-of-the-art sharding-based blockchain protocols, which are RSCoin [100], Chainspace [16], Elastico [4], OmniLedger [5], and RapidChain [6]. However, the scope of the survey [20], is limited to the sharding-based solutions and it only compares five sharding-based blockchain protocols. Our work overviews more than fifteen sharding-based blockchain protocols and provides a security analysis based on computing the failure probability and years to fail. Furthermore, our work studies all existing scalability solutions. Zhou *et al.* [21] overview blockchain scalability solutions and classify them into 3 layers: (1) Layer 1 solutions (on-chain solutions) include block data, consensus, sharding, as well as DAG-based blockchains; (2) Layer 2 solutions (non on-Chain) include payment channels, sidechains, cross-chains, and off-chains; and (3) Layer 0 solutions include data propagation. However, the authors [21] do not cover sharding-based solutions in detail; instead they explain these solutions at a high level, without classification and comparison. Furthermore, they do not cover all existing sharding-based solutions. Finally, they do not consider security analysis of these solutions. Our paper focuses on sharding solutions (the most promising scalability solutions) and classifies sharding-based blockchain protocols into three categories: (1) Sharding-based *PoW* and *PFT*; (2) Sharding-based *PoS* and *PFT*; and (3) Sharding-based on other consensus. Our paper presents a performance-based comparative analysis (i.e., throughput and latency) of the advantages and disadvantages of existing scalability solutions. Furthermore, our paper provides findings (e.g., CoDAG [86], Stellar [63], μRaiden [69], Bitcoin RootStock [72]) and insights (e.g., a comparison between conventional payment systems such as PayPal and blockchain payments systems) that are not covered in [21].

VIII. CONCLUSION

This work surveys the most well-known solutions of the scalability issue. We define a taxonomy in which we classify and survey these solutions by carefully discussing and comparing their performances, strengths and weaknesses. The first layer category includes all approaches that propose modification within the blockchain such as consensus and data storage. The second layer category includes all the approaches that propose modifications outside the chain. To summarize, the present work introduces a performance-based comparative analysis (i.e., throughput and latency), and the advantages and disadvantages of existing scalability solutions.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Working Paper, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] A. Hafid, A. S. Hafid, and M. Samih, "A methodology for a probabilistic security analysis of sharding-based blockchain protocols," in *Proc. Int. Congr. Blockchain Appl.* Cham, Switzerland: Springer, 2019, pp. 101–109.

[3] A. Hafid, A. S. Hafid, and M. Samih, "New mathematical model to analyze security of sharding-based blockchain protocols," *IEEE Access*, vol. 7, pp. 185447–185457, 2019.

[4] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 17–30.

[5] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "OmniLedger: A secure, scale-out, decentralized ledger via sharding," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 583–598.

[6] M. Zamani, M. Movahedi, and M. Raykova, "RapidChain: Scaling blockchain via full sharding," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Jan. 2018, pp. 931–948.

[7] *The ZILLIQA Technical Whitepaper*, ZILLIQA Team and Others, vol. 16, Sep. 2017, p. 2019.

[8] Harmony Team, Technical Whitepaper. *Harmony*. Accessed: Feb. 11, 2020. [Online]. Available: https://harmony.one/whitepaper.pdf

[9] A. Manuskin, M. Mirkin, and I. Eyal, "Ostraka: Secure blockchain scaling by node sharding," 2019, *arXiv:1907.03331*. [Online]. Available: http://arxiv.org/abs/1907.03331

[10] M. Zochowski. (2018). *A Highly Scalable Decentralized Transaction System, Version 1.0*. [Online]. Available: https://logos.network/whitepaper.pdf

[11] Ethereum 2.0. *Ethereum Roadmap*. Accessed: Jan. 28, 2020. [Online]. Available: https://docs.ethhub.io/

[12] V. Buterin. *Ethereum Sharding FAQ*. Accessed: Jan. 28, 2020. [Online]. Available: https://github.com/ethereum/wiki/wiki/Sharding-FAQ

[13] J. Wang and H. Wang, "Monoxide: Scale out blockchains with asynchronous consensus zones," in *Proc. 16th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2019, pp. 95–112.

[14] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," in *Proc. Int. Conf. Manage. Data (SIGMOD)*, 2019, pp. 123–140.

[15] A. G. Stegos, "A platform for privacy applications," White Paper Version 1.0, 2019. [Online]. Available: https://stegos.com/docs/whitepaper

[16] M. Al-Bassam, A. Sonnino, S. Bano, D. Hrycyszyn, and G. Danezis, "Chainspace: A sharded smart contracts platform," 2017, *arXiv:1708.03778*. [Online]. Available: http://arxiv.org/abs/1708.03778

[17] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, Apr. 2014. [Online]. Available: https://gavwood.com/paper.pdf

[18] S. Kim, Y. Kwon, and S. Cho, "A survey of scalability solutions on blockchain," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2018, pp. 1204–1207.

[19] G. Yu, X. Wang, K. Yu, W. Ni, J. A. Zhang, and R. P. Liu, "Survey: Sharding in blockchains," *IEEE Access*, vol. 8, pp. 14155–14181, 2020.

[20] G. Wang, Z. J. Shi, M. Nixon, and S. Han, "SoK: Sharding on blockchain," in *Proc. 1st ACM Conf. Adv. Financial Technol.*, Oct. 2019, pp. 41–61.

[21] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to scalability of blockchain: A survey," *IEEE Access*, vol. 8, pp. 16440–16455, 2020.

[22] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and open research challenges," *IEEE Access*, vol. 7, pp. 10127–10149, 2019.

[23] K. Huang, X. Zhang, Y. Mu, F. Rezaeibagha, X. Du, and N. Guizani, "Achieving intelligent trust-layer for Internet-of-Things via self-redactable blockchain," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2677–2686, Apr. 2020.

[24] A. Shahnaz, U. Qamar, and A. Khalid, "Using blockchain for electronic health records," *IEEE Access*, vol. 7, pp. 147782–147795, 2019.

[25] L. Mertz, "(Block) chain reaction: A blockchain revolution sweeps into health care, offering the possibility for a much-needed data solution," *IEEE Pulse*, vol. 9, no. 3, pp. 4–7, May 2018.

[26] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, "Blockchain technologies for the Internet of Things: Research issues and challenges," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2188–2204, Apr. 2019.

[27] H. Yao, T. Mai, J. Wang, Z. Ji, C. Jiang, and Y. Qian, "Resource trading in blockchain-based industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3602–3609, Jun. 2019.

[28] A. Nordrum, "Govern by blockchain dubai wants one platform to rule them all, while Illinois will try anything," *IEEE Spectr.*, vol. 54, no. 10, pp. 54–55, Oct. 2017.

[29] C. Shen and F. Pena-Mora, "Blockchain for cities—A systematic literature review," *IEEE Access*, vol. 6, pp. 76787–76819, 2018.

[30] J. Abou Jaoude and R. George Saade, "Blockchain applications-usage in different domains," *IEEE Access*, vol. 7, pp. 45360–45381, 2019.

[31] *Bitnodes*. Accessed: Dec. 19, 2019. [Online]. Available: https://bitnodes.earn.com

[32] *Visa*. Accessed: Mar. 23, 2020. [Online]. Available: https://usa.visa.com/run-your-business/small-business-tools/retail.html

[33] *Network Number 1*. Accessed: Dec. 19, 2019. [Online]. Available: https://www.ethernodes.org/network/1

[34] V. Buterin. Accessed: Oct. 25, 2019. [Online]. Available: https://github.com/vbuterin/scalability_paper

[35] J. Poon and V. Buterin, "Plasma: Scalable autonomous smart contracts," White Paper, 2017, pp. 1–47. [Online]. Available: https://plasma.io/plasma.pdf

[36] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," DRAFT Version 0.5.9.2, 2016, pp. 1–59. [Online]. Available: https://lightning.network/lightning-network-paper.pdf

[37] H.-W. Wang. (2017). *Ethereum Sharding: Overview and Finality*. Accessed: Sep. 8, 2019. [Online]. Available: https://medium.com/@icebearhww

[38] H.-W. Wang. *On Sharding Blockchains*. Accessed: Mar. 27, 2020. [Online]. Available: https://github.com/ethereum/wiki/wiki/Sharding-FAQ

[39] J. Garzik, "Block size increase to 2MB," Bitcoin Improvement Proposal, Tech. Rep., 2015, vol. 102.

[40] R. Network-Fast. (2018). *Cheap, Scalable Token Transfers for Ethereum*. [Online]. Available: https://raiden.network/

[41] Komodo, "Advanced blockchain technology, focused on freedom," Working Paper, 2018. [Online]. Available: https://docs.komodoplatform.com/

[42] E. Kokoris-Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford, "Enhancing bitcoin security and performance with strong consistency via collective signing," in *Proc. 25th USENIX Secur. Symp. USENIX Secur.*, 2016, pp. 279–296.

[43] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, Jul. 1982.

[44] G. Danezis and S. Meiklejohn, "Centrally banked cryptocurrencies," 2015, *arXiv:1505.06895*. [Online]. Available: http://arxiv.org/abs/1505.06895

[45] BloXroute Team. (2018). *The Scalability Problem, (Very) Simply Explained*. Accessed: Feb. 7, 2020. [Online]. Available: https://medium.com/bloxroute

[46] BitRewards. (2018). *Blockchain Scalability: The Issues, and Proposed Solutions*. Accessed: Feb. 7, 2020. [Online]. Available: https://medium.com/bitrewards/

[47] BitInfoCharts. *Cryptocurrency Statistics*. Accessed: Feb. 7, 2020. [Online]. Available: https://bitinfocharts.com/
[48] Blockchain.com. *Bitcoin Charts*. Accessed: Apr. 11, 2020. [Online]. Available: https://www.blockchain.com/es/charts
[49] J. Wroughton and T. Cole, "Distinguishing between binomial, hypergeometric and negative binomial distributions," *J. Statist. Educ.*, vol. 21, no. 1, Mar. 2013.
[50] T. Hanke, M. Movahedi, and D. Williams, "DFINITY technology overview series, consensus system," 2018, *arXiv:1805.04548*. [Online]. Available: http://arxiv.org/abs/1805.04548
[51] Bitcoin.org. *Unspent Transaction Output, UTXO*. Accessed: Jan. 22, 2020. [Online]. Available: https://bitcoin.org/en/glossary/unspent-transaction-output
[52] S. Delgado-Segura, C. Pérez-Sola, G. Navarro-Arribas, and J. Herrera-Joancomartí, "Analysis of the bitcoin UTXO set," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Cham, Switzerland: Springer, 2018, pp. 78–91.
[53] R. Pass and E. Shi, "Hybrid consensus: Efficient consensus in the permissionless model," in *Proc. 31st Int. Symp. Distrib. Comput. (DISC)*, 2017, pp. 1–56.
[54] F. McKeen, I. Alexandrovich, A. Berenzon, C. V. Rozas, H. Shafi, V. Shanbhogue, and U. R. Savagaonkar, "Innovative instructions and software model for isolated execution," in *Proc. HASP ISCA*, vol. 10, no. 1, 2013.
[55] EOS.IO. *EOS.IO Technical White Paper v2*. Accessed: Feb. 21, 2020. [Online]. Available: https://github.com/EOSIO/Documentation
[56] J. Garzik, "Block size increase to 2MB," Bitcoin Improvement Proposal, Tech. Rep., 2015, vol. 102.
[57] *Bitcoin Unlimited*. Accessed: Feb. 27, 2020. [Online]. Available: https://www.bitcoinunlimited.info/
[58] L. Eric, L. Johnson, and W. Pieter. (2015). Segregated Witness. Github Repository. Accessed: Jan. 15, 2020. [Online]. Available: https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki
[59] R. Jeremy, M. Naik, and N. Subramanian, "Merkelized abstract syntax trees," Tech. Rep., 2014.
[60] W. Pieter. (2014). Dealing with malleability. Github Repository. Accessed: Jan. 15, 2020. [Online]. Available: https://github.com/bitcoin/bips/blob/master/bip-0062.mediawiki
[61] BitcoinCore. *Segregated Witness Benefits*. Accessed: Jan. 15, 2020. [Online]. Available: https://bitcoincore.org/en/2016/01/26/segwit-benefits/linear-scalingof-sighash
[62] *EOSIO*. Accessed: Jan. 26, 2020. [Online]. Available: https://eos.io/strategic-vision/
[63] D. Mazieres, "The stellar consensus protocol: A federated model for internet-level consensus," Stellar Develop. Found., Tech. Rep., 2015, vol. 32.
[64] Stellar. *What is Stellar?*. Accessed: Feb. 3, 2020. [Online]. Available: https://www.stellar.org/overview
[65] V. Buterin. *Minimal Viable Plasma*. Accessed: Jan. 28, 2020. [Online]. Available: https://ethresear.ch/t/minimal-viable-plasma/426
[66] B. Jones and K. Fichter. *More Viable Plasma*. Accessed: Jan. 28, 2020. [Online]. Available: https://ethresear.ch/t/more-viable-plasma/2160
[67] V. Buterin. *Plasma Cash: Plasma With Much Less Per-User Data Checking*. Accessed: Jan. 28, 2020. [Online]. Available: https://ethresear.ch/top
[68] V. Robinson. *Plasma Debit: Arbitrary-Denomination Payments in Plasma Cash*. Accessed: Jan. 28, 2020. [Online]. Available: https://ethresear.ch/top
[69] μRaiden. *A Payment Channel Framework for Fast and Free Off-Chain ERC20 Token Transfers*. Accessed: Feb. 28, 2020. [Online]. Available: https://raiden.network/micro.html
[70] Trinity. *Universal Off-Chain Scaling Solution*. Accessed: Feb. 28, 2020. [Online]. Available: https://trinity.tech/
[71] Trinity, "Universal off-chain scaling solution," Trinity White Paper. Accessed: Feb. 28, 2020. [Online]. Available: https://trinity.tech/#/writepaper
[72] S. D. Lerner, "RSK," White Paper. Accessed: Mar. 19, 2020. [Online]. Available: https://www.rsk.co/Whitepapers/RSK-White-Paper-Updated.pdf
[73] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.
[74] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 839–858.
[75] G. Becker, "Merkle signature schemes, Merkle trees and their cryptanalysis," Ruhr-University Bochum, Bochum, Germany, Tech. Rep., 2008.
[76] J. Rubin, M. Naik, and N. Subramanian, "Merkelized abstract syntax trees," Tech. Rep., 2014.
[77] M. Al-Bassam, A. Sonnino, and V. Buterin, "Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities," 2018, *arXiv:1809.09044*. [Online]. Available: http://arxiv.org/abs/1809.09044
[78] P. Maymounkov and D. Mazières, "Kademlia: A peer-to-peer information system based on the XOR metric," in *Proc. Int. Workshop Peer-to-Peer Syst.* Cham, Switzerland: Springer, 2002, pp. 53–65.
[79] V. Buterin and V. Griffith, "Casper the friendly finality gadget," 2017, *arXiv:1710.09437*. [Online]. Available: http://arxiv.org/abs/1710.09437
[80] V. Buterin and V. Griffith, "SPECTRE: A fast and scalable cryptocurrency protocol," 2017, *arXiv:1710.09437*. [Online]. Available: http://arxiv.org/abs/1710.09437
[81] *IOTA*. Accessed: Apr. 18, 2020. [Online]. Available: https://www.iota.org/get-started/what-is-iota
[82] P. Serguei, "The tangle," Tech. Rep., 2016, p. 131.
[83] Y. Sompolinsky and A. Zohar, "Phantom: A scalable blockdag protocol," *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 104, 2018.
[84] H. Pervez, M. Muneeb, M. U. Irfan, and I. U. Haq, "A comparative analysis of DAG-based blockchain architectures," in *Proc. 12th Int. Conf. Open Source Syst. Technol. (ICOSST)*, Dec. 2018, pp. 27–34.
[85] T. Zhou, X. Li, and H. Zhao, "DLattice: A permission-less blockchain based on DPoS-BA-DAG consensus for data tokenization," *IEEE Access*, vol. 7, pp. 39273–39287, 2019.
[86] L. Cui, S. Yang, Z. Chen, Y. Pan, M. Xu, and K. Xu, "An efficient and compacted DAG-based blockchain protocol for industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4134–4145, Jun. 2020.
[87] *NANO*. Accessed: Apr. 19, 2020. [Online]. Available: https://nano.org/foundation
[88] *XDAG*. Accessed: Apr. 19, 2020. [Online]. Available: https://xdag.io/
[89] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1432–1465, 2nd Quart., 2020, doi: 10.1109/COMST.2020.2969706.
[90] V. Sivaraman, S. B. Venkatakrishnan, R. Shaileshh Bojja, N. Kathleen, Y. Parimarjan, M. Lei, G. Fanti, and M. Alizadeh, "High throughput cryptocurrency routing in payment channel networks," in *Proc. 17th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2020, pp. 777–796.
[91] Digiconomist. *Bitcoin Energy Consumption Index*. Accessed: Apr. 17, 2020. [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption
[92] University of Cambridge. *Cambridge Bitcoin Electricity Consumption Index*. Accessed: Apr. 17, 2020. [Online]. Available: https://www.cbeci.org/comparisons/
[93] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen, and D. I. Kim, "A survey on consensus mechanisms and mining strategy management in blockchain networks," *IEEE Access*, vol. 7, pp. 22328–22370, 2019.
[94] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer, "On scaling decentralized blockchains," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Cham, Switzerland: Springer, 2016, pp. 106–125.
[95] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-NG: A scalable blockchain protocol," in *Proc. 13th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2016, pp. 45–59.
[96] *Bitcoin Cash*. Accessed: Apr. 21, 2020. [Online]. Available: https://www.bitcoincash.org/
[97] *Bitcoin Classic*. Accessed: Apr. 21, 2020. [Online]. Available: https://bitcoinclassic.com/
[98] H. Chen and Y. Wang, "SSChain: A full sharding protocol for public blockchain without data migration overhead," *Pervas. Mobile Comput.*, vol. 59, Oct. 2019, Art. no. 101055.
[99] A. Efe Gencer, R. van Renesse, and E. G. Sirer, "Service-oriented sharding with aspen," 2016, *arXiv:1611.06816*. [Online]. Available: http://arxiv.org/abs/1611.06816
[100] G. Danezis and S. Meiklejohn, "Centrally banked cryptocurrencies," 2015, *arXiv:1505.06895*. [Online]. Available: http://arxiv.org/abs/1505.06895

[101] J. R. Douceur, "The sybil attack," in *Proc. Int. Workshop Peer-to-Peer Syst.* Cham, Switzerland: Springer, 2002, pp. 251–260.
[102] M. Rosenfeld, "Analysis of hashrate-based double spending," 2014, *arXiv:1402.2009*. [Online]. Available: http://arxiv.org/abs/1402.2009
[103] E. Androulaki, C. Cachin, A. De Caro, and E. Kokoris-Kogias, "Channels: Horizontal scaling and confidentiality on permissioned blockchains," in *Proc. Eur. Symp. Res. Comput. Secur.* Cham, Switzerland: Springer, 2018, pp. 111–131.
[104] G. Wang, Z. J. Shi, M. Nixon, and S. Han, "SMChain: A scalable blockchain protocol for secure metering systems in distributed industrial plants," in *Proc. Int. Conf. Internet Things Design Implement.*, Apr. 2019, pp. 249–254.
[105] S. Li, M. Yu, C.-S. Yang, A. Salman Avestimehr, S. Kannan, and P. Viswanath, "PolyShard: Coded sharding achieves linearly scaling efficiency and security simultaneously," 2018, *arXiv:1809.10361*. [Online]. Available: http://arxiv.org/abs/1809.10361
[106] C. Huang, Z. Wang, H. Chen, Q. Hu, Q. Zhang, W. Wang, and X. Guan, "RepChain: A reputation-based secure, fast and high incentive blockchain system via sharding," 2019, *arXiv:1901.05741*. [Online]. Available: http://arxiv.org/abs/1901.05741
[107] Y. Liu, F. R. Yu, X. Li, H. Ji, and V. C. M. Leung, "Blockchain and machine learning for communications and networking systems," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1392–1431, 2nd Quart., 2020, doi: 10.1109/COMST.2020.2975911.
[108] R. C. Merkle, "Protocols for public key cryptosystems," in *Proc. IEEE Symp. Secur. Privacy*, Apr. 1980, pp. 122–133.
[109] Y. Huang, H. Wang, L. Wu, G. Tyson, X. Luo, R. Zhang, X. Liu, G. Huang, and X. Jiang, "Characterizing EOSIO blockchain," 2020, *arXiv:2002.05369*. [Online]. Available: http://arxiv.org/abs/2002.05369
[110] S. Somin, G. Gordon, and Y. Altshuler, "ERC20 transactions over ethereum blockchain: A Network Perspective," Tech. Rep.
[111] Y. Kwon, H. Kim, J. Shin, and Y. Kim, "Bitcoin vs. Bitcoin cash: Coexistence or downfall of bitcoin cash?" in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Francisco, CA, USA, May 2019, pp. 935–951.
[112] PayPal. *Low, Flat-Rate Merchant Services Fees*. Accessed: May 9, 2020. [Online]. Available: https://www.paypal.com/ca/business/fees
[113] *Wallet Investor*. Accessed: May 11, 2020. [Online]. Available: https://walletinvestor.com/converter/gas/ethereum/1
[114] L. Gudgeon, P. Moreno-Sanchez, S. Roos, P. McCorry, and A. Gervais, "SoK: Off the chain transactions," IACR Cryptol. ePrint Arch., Tech. Rep., 2019.
[115] H. S. Galal, M. ElSheikh, and A. M. Youssef, "An efficient micropayment channel on ethereum," in *Proc. Data Privacy Manage., Cryptocurrencies Blockchain Technol.* Cham, Switzerland: Springer, 2019, pp. 211–218.
[116] MasterCard. *2019–2020 U.S. Region Interchange Programs and Rates*. Accessed: May 12, 2020. [Online]. Available: https://www.mastercard.us/en-us.html
[117] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1432–1465, 2nd Quart., 2020, doi: 10.1109/COMST.2020.2969706.

ABDELATIF HAFID (Member, IEEE) received the B.Sc. degree in mathematics and applications from the University of Moulay Ismail, Meknes, Morocco, and the M.Sc. degree in mathematical engineering from the University of Abdelmalek Essaâdi, Tangier, Morocco. He is currently pursuing the Ph.D. degree with the University of Moulay Ismail, in cooperation with the University of Montreal, Montreal, QC, Canada. His current research interests include applied probability, statistics, and blockchain.

ABDELHAKIM SENHAJI HAFID (Member, IEEE) was an Assistant Professor with Western University (WU), Canada, a Research Director of the Advance Communication Engineering Center (venture established by WU, Bell Canada, and Bay Networks), Canada, a Researcher with CRIM, Canada, a Visiting Scientist with GMD-Fokus, Germany, and a Visiting Professor with the University of Evry, France. He is currently a Full Professor with the University of Montreal. He is also the Founding Director of the Network Research Laboratory and the Montreal Blockchain Laboratory, and a Research Fellow with CIRRELT, Montreal, QC, Canada. Prior to joining the University of Montreal, he spent several years, as a Senior Research Scientist, at Bell Communications Research (Bellcore), NJ, USA, working in the context of major research projects on the management of next generation networks He has extensive academic and industrial research experience in the area of the management and design of next generation networks. His current research interests include the IoT, Fog/edge computing, blockchain, and intelligent transport systems.

MUSTAPHA SAMIH received the Ph.D. degree in fundamental and applied mathematics from the University of Montpellier, France. He is currently a Full Professor with the University of Moulay Ismail, Meknes, Morocco. His current research interests include applied probability, statistics, and blockchain.