IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received February 4, 2020, accepted March 4, 2020, date of publication March 10, 2020, date of current version March 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2979881

When Blockchain Meets Distributed File Systems: An Overview, Challenges, and Open Issues

HUAWEI HUANG, (Member, IEEE), JIANRU LIN, (Member, IEEE), BAICHUAN ZHENG, ZIBIN ZHENG, (Senior Member, IEEE), AND JING BIAN
School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China
National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou 510006, China

Corresponding author: Zibin Zheng (zhzibin@mail.sysu.edu.cn)

This work was supported in part by the National Key Research and Development Program under Grant 2016YFB1000101, in part by the National Natural Science Foundation of China under Grant 61902445 and Grant 61722214, and in part by the Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2016).

ABSTRACT Constructing globally distributed file systems (DFS) has received great attention. Traditional Peer-to-Peer (P2P) distributed file systems have inevitable drawbacks such as instability, lacking auditing and incentive mechanisms. Thus, Inter-Planetary File System (IPFS) and Swarm, as the representative DFSs which integrate with blockchain technologies, are proposed and becoming a new generation of distributed file systems. Although the blockchain-based DFSs successfully provide adequate incentives and security guarantees by exploiting the advantages of blockchain, a series of challenges, such as scalability and privacy issues, are also constraining the development of the new generation of DFSs. Mainly focusing on IPFS and Swarm, this paper conducts an overview of the rationale, layered structure and cutting-edge studies of the blockchain-based DFSs. Furthermore, we also identify their challenges, open issues and future directions. We anticipate that this survey can shed new light on the subsequent studies related to blockchain-based distributed file systems.

INDEX TERMS Blockchain, distributed file systems, IPFS, swarm.

I. INTRODUCTION

There have been many attempts dedicated to constructing a distributed file system. The phenomenal popularity and study of Peer-to-Peer (P2P) services, such as Napster [1], Gnutella [2], Kazaa [3] and Morpheus [4], make the implementation of distributed file systems an exciting and promising research field. As one of the most successful P2P distributed file system, BitTorrent [5] has supported over 100 million online users. It has a large-scale deployment where tens of millions of nodes join and churn everyday. In a distributed file system, storage resources and system clients are dispersed in the network. Each user is both a creator and a consumer of data stored in the system. Thus, the challenge is to provide considerable incentives in an efficient, secure and practical manner.

By far, the biggest distributed file system is HyperText Transfer Protocol (HTTP), which is a web server used to upload data. Then, other peers can access to a particular data anywhere allover the world. To ensure the data accessibility in web servers, a maintaining cost needs to pay. Such maintaining cost increases along with the growth of data popularity. Moreover, another problem is that there are very few ways to share the burden of information dissemination with the clients directly. This is because HTTP lacks upgrading design and thus fails to take advantages of the advanced file distribution techniques proposed in the past few years. Meanwhile, P2P technique had been gathering a great pace and soon dominated the majority of data packets in the Internet. Such P2P file systems, like BitTorrent [5], optimize resources brilliantly by giving different pieces of popular data to clients and enabling them swap the missing parts between each another. In this way, the bandwidth consumption of hosts can be balanced and the overall cost of operational expenditure (OPEX) can be also degraded.

Although BitTorrent has a lot of advantages aforementioned, the following inevitable drawbacks cannot be ignored:

1) Downloading is unstable, which limits BitTorrent to be widely used in specific occasions.

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

2) Unable to verify file publishers, and it is hard to guarantee the credibility of the content downloaded.
3) There is no incentive mechanism such that the *seed* nodes are not rewarded for sharing their bandwidth and storage resources.

Anticipating to replace HTTP, Zeronet [6] adopted BitTorrent as the file distribution mechanism for Web content. However, simply sharing bandwidth, storage and computing resources cannot provide the brilliant experience as HTTP users expect.

Recently, blockchain has become a buzzword in both industry and academia, and the combination of blockchain and distributed file system is becoming a promising solution, where blockchain is expected to provide incentives and security for the stored files in systems. Currently, the popular blockchain-based distributed file systems include IPFS [7], Swarm [8], Storj [9], and PPIO [10]. Within those file systems, IPFS is a peer-to-peer distributed file system for storing and accessing files, websites, applications and data; Swarm is a distributed storage platform and content distribution service based on Ethereum; Storj is another peer-to-peer decentralized cloud storage platform that allows users to share data without relying on a third-party data provider; and PPIO is a decentralized programmable storage network that permits users store and retrieve any data from anywhere on web. With respect to the combination with blockchains, IPFS, Swarm, and Storj file systems adopt Filecoin [11], Ethereum [12], and Metadisk [13] as their incentive mechanisms, respectively. PPIO exploits up to 4 proof algorithms, which are explained in Section III, for its incentive layer.

Considering that the technologies of all distributed file systems are similar to IPFS and Swarm, we review the recent cutting-edge studies of blockchain-based DFSs mainly focusing on IPFS and Swarm.

The contribution of this survey includes the following aspects.

- This paper first introduces the layered structure of blockchain-based DFSs. We then make a comprehensive taxonomy of the cutting-edge studies on the *scalability* and *privacy* perspectives.
- We also clarified the challenges, open issues and future directions of the blockchain-based DFSs.
- To the best of our knowledge, this is the first survey related to the blockchain-based DFSs. Our review in this article can help subsequent researchers well understand both the current development and the future trends of the blockchain-based DFS.

The rest of this paper is organized as follows. In Section II, we explain necessary preliminaries and basic concepts. Section III shows the layered structure of distributed file systems. Section IV summarizes the cutting-edge studies. Section V discusses open issues, challenges and future directions. Finally, section VI concludes this paper. We also show the structure of this survey in Figure 1.

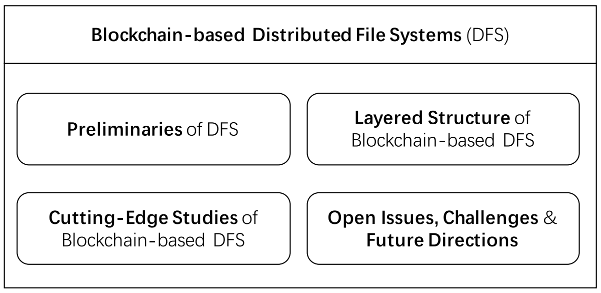

Blockchain-based Distributed File Systems (DFS)
Preliminaries of DFS
Layered Structure of Blockchain-based DFS
Cutting-Edge Studies of Blockchain-based DFS
Open Issues, Challenges & Future Directions

FIGURE 1. The structure of this article.

II. PRELIMINARIES

Since the blockchain-based distributed file systems emphasized on this article have a close correlation with the basic data structure of blockchains, we first introduce the preliminaries of Merkle Tree and Merkle DAG. Then, we have an overview of BitTorrent, which can help us understand the rationale of distributed file systems such as IPFS and Swarm.

A. MERKLE TREE AND MERKLE DAG

Merkle Tree [14] is a binary tree built based on a cryptographic hash function. Each leaf in an merkle tree has a hash value which is computed by one or multiple imported values. Each parent node derives its hash value from children's value which is recursively dependent on all values in its subtree. Figure 2 illustrates an example of a merkle tree, each leaf (H1-H4) obtains its value though computing imported value (D1-D4) and parents (H5-H6) derive values from their children (H1-H4) and finally the root of this merkle tree (H7) is obtained which is relevant with every value in the tree.

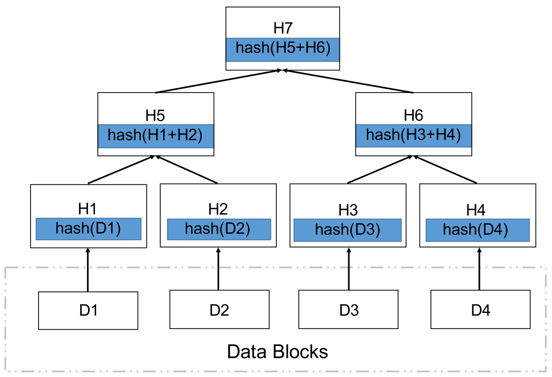


FIGURE 2. An illustration of a Merkle Tree.

In blockchain area, merkle tree is usually used for integrity validations (for example the block validation in bitcoin [15]) and quick validations (for example the light peers in Ethereum [12]). Since a tiny change in a merkle tree can drastically change the root of the tree, we can do integrity validation by simply storing the root. To validate if a node is in a merkle tree, only the a few hashes of nodes are needed, instead of the entire tree. For example in Figure 2, H4 and H5 are needed to validate if H3 is in the tree. Using

H3, H4 and H5, a root (H8) can be computed. By comparing H7 and H8, we can confirm that H3 is in the tree if two roots are the same, or H3 is not in the tree if two roots are different.

Similar to the concept of merkle tree, Merkle DAG (Directed Acyclic Graph) [7] is used in IPFS as a data object model. An object in IPFS is a structure containing two attributes: *Data* and *Links*. Each Link structure includes three attributes: Name, Hash and Size. Using this object structure, IPFS can compose objects and build a directed acyclic graph. In IPFS, merkle DAG organizes the structure of a file or even a file directory which is shown in Figure 3. In this figure, there are two files (example.js and hello.txt) and one file path (dir) in the root path of this file directory, example.js is divided into three different data pieces and file path dir has two files: other.txt and example.txt (here file content of example.txt in dir and hello.txt in root path are exactly the same therefore they are linked to the same object), each object derives its value though computing its children's value and the content of data.

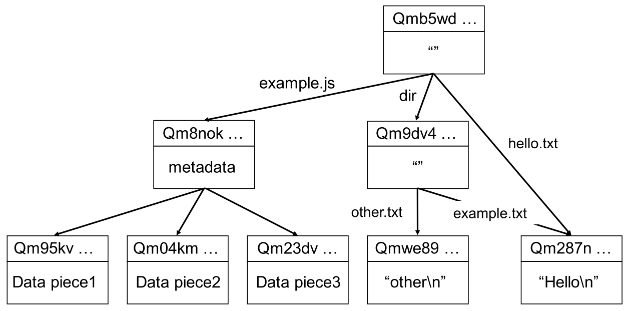


FIGURE 3. Merkle DAG in IPFS.

B. OVERVIEW OF BitTorrent

As mentioned earlier, BitTorrent (BT) is one of the most popular distributed file systems. Basically, the process of file sharing in BitTorrent is illustrated in Figure 4, and can be described with the following 5 steps:

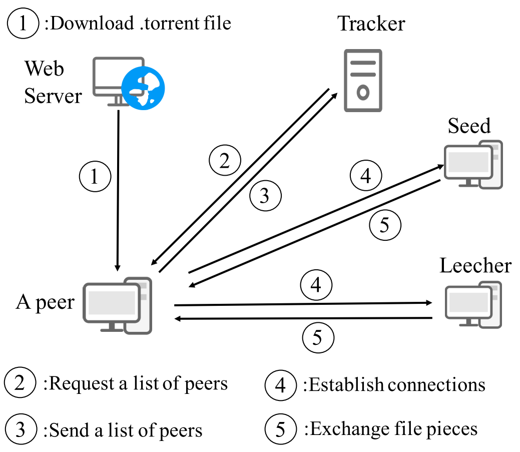


FIGURE 4. Mechanism of file-sharing in BitTorrent, which can help us well understand the blockchain-based distributed file system IPFS.

- Peer A interacts with a Web server and downloads a *.torrent* file.
- Peer A interacts with the tracker which peer A finds in the **.torrent* file, and requests a list of peers that are in the Torrent network.
- Tracker sends a list of a specified number of peers that are in the Torrent network.
- Peer A selects randomly a part of candidate peers from the list as its neighbors and establishes connections with each of them.
- Then peer A can exchange file pieces with its neighbors using swarming technique.

In BitTorrent, an overlay network called Torrent is established when each file is being distributed. Torrent is composed by peers in a network which can be classified into two types: seed and leecher. A seed is a client which has a complete copy of a file, while a leecher is a client which is downloading a file. Besides seed and leecher, the Web servers and trackers are also required. If a peer wants to join a Torrent network, it can obtain a *.torrent* file from a Web server. This file contains information of a file including its name, length, hash digest and the URL of the tracker. A tracker is a special peer storing the meta information of peers which are active in a Torrent network. A peer can interact with a tracker and obtain the list of IP/Port pairs of other peers in a Torrent, and then select randomly about 20-40 peers from the list as its neighbors. In BitTorrent, a file exchanging technique called swarming is adopted to separate a file into fixed-size pieces each of which is usually with a 256 KB in size [5]. When a piece is fully downloaded, a peer compares its SHA1 hash value with the value in the *.torrent* file. If match, the peer announces the availability of this complete piece to its neighbors for further file exchanging and downloading.

III. THE LAYERED STRUCTURE OF BLOCKCHAIN-BASED DISTRIBUTED FILE SYSTEMS

In this section, we present the typical layered structure of blockchain-based distributed file systems by particularly emphasizing on IPFS and Swarm. The structure is shown in Table 1 in detail. Generally, we classify 7 layers behind the popular distributed file systems, i.e., Identities Layer, Data Layer, Data-swap Layer, Network Layer, Routing Layer, Consensus Layer and Incentive Layer. Each layer is a critical module for distributed file systems. We summarize their functions and related references in Table 1.

A. IDENTITY LAYER

To archive the content distribution between nodes in P2P file system, each node has to be identified by a unique identifier, which needs to ensure collision-free. It means that two different data objects can never map to the same identifier. In IPFS, the encrypted hash (in *multi-hash* format) of a public key, i.e., *NodeId*, is used to identify each node. The format of multi-hash is ⟨hash function code⟩⟨hash digest length⟩⟨hash digest bytes⟩. Nodes periodically check public keys and *NodeId* when connecting with each other. In Swarm

TABLE 1. Layered structure of distributed file systems such as IPFS & Swarm.

Layer	Function	Examples and References
Identity Layer	Identity layer assigns unique id for each node	Keccak hash [16]
Data layer	Data layer organizes file structure in distributed file system	Merkle DAG [7]
Data-swap layer	Data swap layer formulates file sharing strategy between each node	BitSwap [5]
Network layer	Network layer enables nodes to discover other nodes, establish connections and exchange file with each other in a secure environment	libP2P [17], Devp2p [18]
Routing layer	Routing layer enables each file piece to be located and accessible by nodes in the network	Distributed sloppy hash table (DSHT) [19], Distributed preimage archive (DPA) [20]
Consensus layer	Consensus layer ensures ledger recording transactions in each node are basically correct and encourages users to maintain the consistency of the network	"Expected Consensus" [21], Proof-of-Work [12]
Incentive layer	Incentive layer establishes reward and punishment mechanism of distributed file system, encourages nodes in the network to be active and honest in the transaction	Filecoin [11], SWAP, SWEAR and SWINDLE [8]

systems, the node hash-address is generated by Keccak 256bit SHA3 [16] using the public key of an Ethereum account.

B. ROUTING LAYER

Generally, the functionalities of the routing layer of a distributed file system includes: 1) maintaining peer-connection topology such that specific peers and data objects can be located, 2) responding to the queries from both local and remote peers, and 3) communicating with distributed hash tables.

IPFS adopts Distributed Sloppy Hash Table (DSHT) [19], which is implemented based on S/Kademlia [22] and Coral [23]. Such the DSHT located in a peer can help find 1) the network addresses of other peers, and 2) the group of peers who can serve specific data objects. The conventional Distributed Hash Table (DHT) stores small values. For larger values, DSHT stores references, i.e., the *NodeIds* of peers who can serve a block. It should be noticed that IPFS is highly modular, and DSHT is just a temporal protocol that can be displaced in the future.

Swarm implements its routing layer using Distributed Preimage Archive (DPA) technique [20]. In such DPA, a source object is divided into equal-sized chunks which are then synced to different nodes. When receiving these content-addressed chunks other nodes could sync them to their neighbors that are in the same address space.

C. NETWORK LAYER

Under the framework of IPFS, an advanced generic P2P solution, named libP2P [17], is exploited as the network layer. libP2P is developed based on bittorrent DHT implementation. Based on libP2P, IPFS can use any network protocol to transfer data. If underlying network is not stable, IPFS can alter to choose UTP [24] or SCTP [25]. IPFS achieves this free shifting mainly by using *multiaddr* formatted technique [7], which combines addresses and corresponding protocols.

Swarm relies on the Ethereum P2P network, which is comprised of three different protocols: 1) RLPx (Recursive Length Prefix) [26] for node discovery and secure data transmission, 2) DevP2P [18] for node session establishment and message exchange, and 3) Ethereum subprotocol [27]. DevP2P [18] is inspired by libP2P and has security properties that are beneficial to Swarm. When discovering through RLPx, Swarm nodes establish TCP connections and send "HELLO" messages including *NodeId*, listening port and other attributes based on DevP2P. Sessions start to transmit data packets. Due to the ecosystem of Ethereum, Swarm has a large number of long-term nodes, which support the robustness and stability of Swarm systems.

D. DATA LAYER

There are four levels of the data model in IPFS:

- Block: an arbitrary-sized piece of data.
- List: a collection of blocks or other lists.
- Tree: a collection of blocks, lists, or other trees.
- Commit: a snapshot in the version history of a tree.

Such data model is similar to that of Git [28]. Based on this data model, IPFS systems employ Merkle DAG to store data. Merkle DAG identifies data and links in each data object with *multi-hash* technique [7], which protects stored data from tampering, and makes file path to be retrieved easily because data object is converted into string-formatted path (with a format like /ipfs/object-hash/object-name). To divide a file into independent blocks, IPFS exploits many algorithms such as rsync rolling-checksum algorithm [29], and Rabin Fingerprints [30].

Swarm also defines a set of data structures:

- Chunk: a fixed-size (maximum 4 KB [7]) piece of data.
- File: a complete set of chunks.
- Manifest: a mapping between paths and files, which handles file collection.

Chunker, which is a Swarm's component for splitting and recovering files, is able to process live stream data. After being split, chunks are collected to calculate the Swarm hashes, in which a hash algorithm is used to obtain the root hash of the Merkle Tree. The root hash is then used to identify a specific file and avoid tampering. During this procedure, the hash of each chunk is also calculated and is treated as a reference to this chunk.

E. INCENTIVE LAYER

1) INCENTIVE LAYER OF IPFS

As shown in Figure 5, Filecoin [11] is a blockchain-based digital payment system, which supports digital storage and data retrieval for IPFS users. It is adopted as an incentive layer for IPFS. There are two markets in Filecoin: a Storage Market

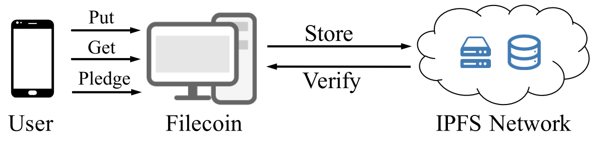


FIGURE 5. Mechanism and position of Filecoin [11], which is adopted by IPFS and exploits blockchain as its fundamental components.

and a Retrieval Market. The data of Storage Market is stored on the Filecoin blockchain, and the data of Retrieval Market is recorded off-chain.

These two markets provide data storage and data retrieval services via a network composed of *Storage Clients*, *Storage Miners*, *Retrieval Clients* and *Retrieval Miners*. Those participants are explained as follows.

- *Storage Clients* are those who need file storage services. They are on the demand side of Storage Market.
- *Storage Miners* are the nodes which provide storage to a Filecoin system using its free disk space. They are on the supply side of Storage Market. The transactions occurred on the Storage Market contribute new blocks to the Filecoin blockchain.
- *Retrieval Clients* are those who desires to retrieve a specific resource from the network. They are the demand side of the Retrieval Market.
- *Retrieval Miners* are those who provide network resources, such as bandwidth, helping retrieval clients search for the retrieval information. They are on the supply side of the Retrieval Market.

To store data in Filecoin, a storage client first submits a bid order to Storage Market. If a storage miner intends to take a bid order, it has to send a request order to Storage Market. When Storage Market is receiving a bid order and a request order, storage clients and storage miners start to exchange blocks and submit a signed deal order to Storage Market. After that, storage miner must prove the data stored in its dedicated uniquely physical storage by repeatedly generating proofs of replication, which is then verified by IPFS.

To retrieve data from Filecoin, similarly a retrieval client first submits a bid order to Retrieval Market. When Retrieval Market is receiving a request order from a retrieval client, the retrieval miners begin to transport data and submit a signed deal order to Retrieval Market to confirm whether a retrieve deal is succeeded or not.

2) INCENTIVE LAYER OF SWARM

In Swarm, incentive scheme consists of two important parts: 1) bandwidth incentives, and 2) storage incentives. This is because bandwidth and storage are the two most important resources in a distributed file system.

a: BANDWIDTH INCENTIVES

In the context of Swarm, the service of delivering chunks is chargeable, and nodes can trade services for services or services for tokens. In order to motivate nodes to provide stable services in a credible context, Swarm proposes the Swarm Accounting Protocol (SWAP) [8]. Firstly, nodes negotiate chunk price when communicating in the handshake protocol. Different prices mean varying bandwidth costs. After chunk price is set, chequebook contract is used to secure the payment. Chequebook contract is a kind of smart contract and has ether (Ethereum token) balance. Another secure payment called *channel contract* is later proposed by Swarm and can be seen in [8]. Both modes of payment support secure off-chain transactions and delayed updates. All of the transactions are stored in the state of Ethereum blockchain which cannot be tampered. Finally, nodes establish network connection and exchange data.

b: STORAGE INCENTIVES

Swarm encourages nodes to preserve the data that has been uploaded to network.Normally long-term data preservation is not realistic. Unpopular chunks do not bring enough profits and may be cleaned up to make room for new chunks. In order to guarantee long-term availability of data, owner of each chunk needs to compensate for storage of nodes. To manage storage deals, Swarm adapts a set of incentive schemes: *SWAP*, *SWEAR* and *SWINDLE*, which are described as follows.

- **SWAP [8]:** Nodes establish connections with their registered peers that are the target nodes they want to compensate to and sign contracts with. Then they can swap information including syncing, receipting, price negotiation and payments.
- **SWEAR [8]:** Registered peers are responsible for their promises of long-term storage and they must register via the SWEAR (Secure Ways of Ensuring Archival or Swarm Enforcement And Registration) [8] contract on Ethereum by uploading their deposit. Peers are stood to be punished and lose deposit in an on-chain litigation process if they violate the rules.
- **SWINDLE [8]:** Nodes provide signed receipts for stored chunks. When dispute about whether the rules are violated has occurred, nodes that lost the chunk can submit a *challenge* to the SWINDLE (Secured With Insurance Deposit Litigation and Escrow) [8] contract by uploading the receipt of the lost chunk. Nodes can also propose the refutation of a *challenge* by uploading the chunk or proof of custody. Swindle contract decides which one is guilty by checking the hash of the chunk.

When chunks are being forwarded, a chain of contracts are created based on the incentive schemes aforementioned, which elegantly solve the disputes between nodes.

F. DATA-SWAP LAYER

IPFS adopts BitSwap [5] as its data-swap layer. BitSwap is based on BitTorrent protocol. In detail, BitSwap nodes provide the blocks they are holding to each other directly, aiming to spread the blocks within their group. The debt of a node raises when it receives target blocks and decreases when it contributes blocks that the other nodes desire. Thus, BitSwap encourages nodes to cache and contribute blocks positively.

To prevent the nodes that never share, each BitSwap node checks the debt of the other peers before they exchange blocks. BitSwap nodes also keep ledgers that record the transferring history, and exchange ledgers with each other when establishing connections. This exchange-policy protects BitSwap ledger from tampering, and isolates the malicious nodes that lose ledger intentionally.

In Swarm, nodes store chunks for selling to get profits when they receive a data-retrieve request. If nodes do not have the target chunk claimed in the retrieve request, they pass the retrieve request to the nearest neighbor node. During managing storage transactions, *receipts* play an important role. When Swarm nodes interact with any contracts, receipts are generated and stored in Swarm. In this way, the source of a chunk is accessible, and a commitment in case of litigations can be traced.

G. CONSENSUS LAYER

Consensus mechanism is critical for every blockchain system. In a large distributed network, multiple peers form a network cluster normally through asynchronous communications. Network could be congested, resulting in that the error messages propagate all over the system. Thus, peers could be failed if they cannot communicate with other with a consensus network view [31]. Therefore, it is necessary to define a resilient consensus protocol that can work in the unreliable asynchronous networks for distributed file systems. The aim of such consensus protocol is to ensure that each peer reaches a secure, reliable and consistent state without a centralized synchronizer.

In the following, we review several typical consensus protocols proposed by recent representative studies.

1) "EXPECTED CONSENSUS" ALGORITHM OF FileCoin

Different from Ethereum which only has one main chain, Filecoin [11] contains not only a single main chain, but also a *storage market* as well. Users in Filecoin interact with the storage market. These interactions of users are stored in the main-chain ledger. Three proofs that play an important role in consensus process of Filecoin are summarized as follows.

- **Transaction Proof:** After miner and user have reached a deal, the main chain locks the token of the user and deposit of the miner. Main chain also records the information about the transaction including hard disk sector of miner, details of deposit, transaction fee and storage deadline, etc.
- **Proof-of-Replication (PoRep):** A file is divided into pieces and each piece is accepted by a storage miner. At this time, a storage miner may pretend to store a piece (this type of behavior is called a generation attack [32]). Furthermore, a miner may obtain a piece from another peer instead of itself (this type of behavior is called an outsourcing attack [32]). Another case is that a miner may create multiple fake peers and pretend to store several replications of a file piece (this type of behavior is called a Sybil attack [32]). To prevent these network attacks, Filecoin requires each miner to submit the proof of replication to the main chain. Such the Proof-of-Replication ensures that each miner stores file pieces truly and independently.
- **Proof-of-Spacetime (PoSt):** To prove that miners keep storing a file piece in the effective time of transaction, each miner has to submit proof of spacetime to the main chain regularly. In the current design of Filecoin, the proof is committed by providing spacetime every 20,000 blocks (roughly consuming 6 days to mine on average) [21] to prove that the file piece is not missing. Storage market has to validate the proofs uploaded by miners and decides whether to punish miners every 100 blocks (50 minutes to mine on average) [21].

The consensus algorithm of Filecoin is called *Expected Consensus* [21], in which a ticket is computed in each round of consensus process. By comparing the ticket value and the effective storage of each peer, a peer or several peers can be the leaders of this round. A leader can select transactions to pack in the new blocks generated. When a block is packed, it will be sent to other peers for synchronization. Transactions in a block are executed by Ethereum virtual machine (EVM) [33], and the state of each account will be updated.

2) CONSENSUS OF ETHEREUM

There are four stages of Ethereum: Frontier, Homestead, Metropolis and Serenity. In the first three stages, proof-of-work (PoW) [34] is adopted as the consensus mechanism of Ethereum, while in the fourth stage, the proof-of-stake (PoS) [35] will be adopted.

In PoW, each miner packs transactions from the transaction pool and constructs a new block in a sequential order. Then miners adjust the nonce value constantly which is imported to PoW function [34] with the block header. A target indicator is also computed according to the difficulty of the blockchain. By comparing the result of this function with the target indicator, the miner decides whether it wins in the consensus process. When a miner confirms that it has won, it starts to broadcast its new block to other peers. Upon receiving a block from other peers, a miner stops computing to validate the nonce value of the newly received block. Each transaction of the new block is executed by EVM. After the processing of all transactions included in this new block, the state of this peer will be updated [33]. Currently, the average time of consensus in Ethereum is around 15 seconds, which ensures the consistency of all peers [36].

3) CONSENSUS ALGORITHMS OF OTHER FILE SYSTEMS

a: STORJ'S PROOFS OF RETRIEVABILITY

Designed as a decentralized cloud object storage, Storj [9] proposed *Proof of Storage* in its first-version white paper. Interestingly, we found that in version 2.0 of Storj's white paper [37], the consensus algorithm has been changed to *Proofs of Retrievability* [38]. Proofs of Retrievability aims at

TABLE 2. Scalability-related studies of distributed file systems.

Category	Reference	File System	Methodology
Scalability Evaluation	Wennergren *et al.* [39]	IPFS	Analyzed scalability performance via varying cluster sizes and replication factors.
	Shen *et al.* [40]	IPFS	Evaluated storage system of IPFS on Amazon EC2 cloud, by emphasizing on the data I/O operations in a client's point of view.
	BlockIPFS [41]	IPFS	Nyaletey *et al.* proposed a solution by integrating blockchain and IPFS, which can trace the access events of each file on IPFS. They evaluated the scalability of BlockIPFS by varying the number of system nodes.
Storage Optimization	Chen *et al.* [42]	IPFS	Proposed a new storage model based on Zigzag code [43], for the block storage scheme adopted by IPFS.
	Norvill *et al.* [44]	IPFS	Proposed an off-chain approach that moves contract-generation codes to IPFS database, aiming to improve the storage performance of distributed file system.
	Design of Ethereum [45]	Swarm	Advocated that a chain of contracts in Swarm should be configured in off-chain storage.
	White paper [9]	Storj	Storj encrypts data using sharding technique and enables data availability based on Reed-Solomon erasure coding [46].
	Lightning Network [47], Plasma [48]	Off-chain file systems	Allows blockchain clients conducting transactions in the off-chain manner, aiming to offload the storage pressure of main-chain.

ensuring a certain piece of file exists on a host. It offers a high availability of files under an ideal proof, in which messages are with minimum size, and pre-processing is minimal. According to the new white paper [37], Poof of retrievability is still under ongoing research and implementation. We then analyze the reason behind the change of Storj's consensus algorithms. It probably because of that the current reputation systems, including proof of storage, fail to solve the *cheating client attacks* [37]. In such cheating attacks, it is hard to independently verify whether a privately verifiable audit under a reputation system was issued or not as claimed. Thus, the proof of storage lacks publicly verifiable practices.

b: PPIO'S 4 PROOF SCHEMES

PPIO [10] exploits difference proof algorithms, i.e., PoRep, PoSt, Proof of Download (PoD), and Light Proof of Capacity (LPoC), in which *PoD* and *LPoC* are two brand new proof mechanisms created by PPIO. *PoD* particularly supports the media streaming related service. *LPoC* is designed to cold start storage miners. However, because LPoC technically occupies hard disk resources with no real values, the PPIO team has decided to abandon the implementation of LPoC.

H. SUMMARY OF THE LAYERED STRUCTURE

As the efficient decentralized storage layer of the next generation Internet, both IPFS and Swarm use similar technologies. They provide low-latency data retrieval, fault-tolerant guarantees and decentralized/distributed storage solutions.

In identities layer, multi-hash technique [7] is used by IPFS which can store the hash function and hash digest. Swarm uses the account address of Ethereum directly. In network layer, Swarm adapts the secure and stable network of Ethereum. IPFS uses libP2P which is a more generic solution. The incentive layer of Swarm relies smart contracts of Ethereum, which support automated auditing and delayed payment. This saves transaction costs of Swarm and remains secure. Filecoin relies on proofs and consensus of blockchain which is an overuse of blockchain. The PoW consensus of Swarm stands the test of time while the *expected consensus* of IPFS remains waiting the test of real-world.

Swarm inherits directly technology design of Ethereum. For example, the identities layer, network layer and consensus layer of Swarm are the same as Ethereum. As Swarm benefits from Ethereum with its large ecosystem, secure and living network and reliable funding sources. IPFS is highly modular and can replace existing component with state-of-the-art technology. In conclusion, the technology of Swarm is more stable while the technology of IPFS is more advanced.

IV. CUTTING-EDGE STUDIES OF BLOCKCHAIN-BASED DISTRIBUTED FILE SYSTEMS

In this section, we discuss recent cutting-edge studies of blockchain-integrated distributed file systems, mainly emphasizing on IPFS and Swarm.

A. SCALABILITY

With the number of transactions increasing in blockchain networks, each peer has to validate and store a growing size of transactions periodically. This incurs a huge burden of both storage and performance to each peer. In addition, the limited size of each block and the latency of consensus-achieving must be taken into account, because these factors induce the delayed transactions. Meanwhile, as the cluster size and data replications growing in network, the performance of IPFS and Swarm degrades severely.

In this part, we review several studies paying attention on the scalability issues of distributed file systems, mainly focusing on IPFS and Swarm. These works can be classified into two categories: 1) scalability evaluation, and 2) storage optimization. For convenient identification, we summarize these studies on Table 2.

1) SCALABILITY EVALUATION

Although the performance of IPFS is under doubting by academia, we only found few research studies that evaluate or discuss the scalability of IPFS. The representative papers are reviewed as follows. Wennergren *et al.* [39] discuss and analyzes the scalability performance of IPFS. They conducted simulations with varying cluster sizes and replication factors. The simulation results show that the average download time of data stored in IPFS increases as cluster size and replication factor grow. In consequence, the response time among peers in an IPFS network grows, and the downloading speed reduces as well. The authors mentioned that the limited bandwidth of each instance of IPFS could be one of the critical reasons for the low scalability of IPFS.

Recently, Shen *et al.* [40] conducted the systematic evaluations of IPFS storage system by deploying real geographically-distributed instances on Amazon EC2 cloud. The authors emphasize on the data I/O operations from a client's perspective. The extensive measurement results show that the access patterns of clients can severely affect the I/O performance of IPFS. Further quantitative analysis indicates that downloading and resolving operations could be bottleneck factors while clients are reading objects from remote nodes.

To address the traceability problem of a distributed file system, Nyaletey *et al.* [41] proposed a solution combining the blockchain and IPFS which named BlockIPFS, which can trace and audit the access events of each file on IPFS. The authors conducted a group of experiments to evaluate the scalability of the proposed BlockIPFS by varying the number of nodes. Then, they measured the latency consumed by uploading, downloading, and reading transactions of each file stored in system. The measurement results show that the increasing number of nodes does not cause drastic growing of transaction times. Unfortunately, the scale of their experiments is too small since the number of nodes in BlockIPFS system is ranging from 3 to 27. Thus, this group of experiments makes the scalability of their system unknown under a very large-scale deployment.

2) STORAGE OPTIMIZATION

a: ERASURE CODES

To guarantee high data availability, some distributed file systems, e.g., Sia [49] and Storj [9], adopt the erasure codes for their storage strategy. In a typical (N, K) erasure code [50], an original file is usually divided into a number K (>1) of blocks. Each block is then encoded to a larger number N $(\geq K)$ of coded blocks. Out of those N encoded blocks, any K of them can reconstruct the original file. Thus, exploiting erasure codes can improve the storage resilience of distributed file systems. For example, to improve user experience of a P2P file system, Chen *et al.* [42] proposed a new storage model based on zigzag [43] and blockchain techniques. The new storage model aims at improving the block storage strategy adopted by IPFS.

b: STORING DATA OFF-CHAIN

On the other hand, we also found other optimized solutions related to the storage of transactions and smart contracts. For instance, to improve the storage performance of distributed file systems, Norvill *et al.* [44] proposed a solution that moves the contract-generation code to an off-chain by treating IPFS as a storage database. In their proposal, Ethereum loads complex contract codes by sending a simple hash value to IPFS peers. By this way, system clients only have to send hash values rather than the full codes when performing fast synchronizations. Thus, the bulk of network traffic can be reduced. In the design of Swarm [45], a chain of contracts is configured to maintain the basic operations. These contracts increase the data size of blockchain such that Swarm is hard to be operated as a full blockchain ledger. Thus, according to reference [44], we know that the developers of Ethereum have been working on Swarm towards an off-chain storage. Some other off-chain solutions such as Lightning Network [47] and Plasma [48] allows participants to execute transactions in the off-chain manner, such that a large portion of on-chain transactions and smart contracts can be offloaded from the main-chain. Thus, integrating the off-chain techniques will bring new solutions to the storage policy of future distributed file systems.

B. PRIVACY

In Swarm and IPFS, data uploaded to the distributed file systems by users is divided into several pieces, which are then stored in different peers. Although the data uploaded can be encrypted, the data content stored in the network is accessible by every peer. Besides, according to the design of IPFS and Swarm, transactions that record developments of a peer can be easily collected. User's information can be revealed through the graph analysis of transactions. For example, according to [51], a client can be identified through the peers it directly connects to. Thus, transactions stored in blockchain behind distributed file systems are publicly visible.

To address these issues, a number of efforts have been devoted to the privacy-preserving of distributed file systems. Through an extensive literature review, we have found many privacy-preserving solutions, mechanisms and applications. Some representative works are classified into two main categories: 1) Access Control, and 2) Peer Anonumity. We also compare several attributes of these studies in Table 3.

1) ACCESS CONTROL

In the distributed file systems such as IPFS and Swarm, although users are not permitted freely to share data within a specific group of peers, this is necessary when taking the privacy issues into account. To provide access control when sharing files, Steichen *et al.* [52] proposed a modified version of IPFS named *acl-IPFS* based on Ethereum. An acl-IPFS peer is constructed by an IPFS peer and an Ethereum account. The uploading, downloading and transferring of data in IPFS networks are achieved though the interaction with smart contracts residing in Ethereum. Smart contracts dynamically maintain the access control lists of each file in acl-IPFS. Users can grant or revoke a permission of a file through smart contracts, too. Aiming to enhance the

TABLE 3. Privacy-related studies of distributed file systems.

Category	Reference	Technology Foundation	Privacy Level	Functionality
Access Control	acl-IPFS [52]	Smart contract, Ethereum, IPFS	Strong	Provided an access control list for the files shared in the proposed acl-IPFS system.
	Ali *et al.* [53]	Consortium blockchain, Sidechain, IPFS	Strong	Proposed a *modular consortium* architecture for privacy preserving towards IoT data.
	Nizamuddin [54]	IPFS, Ethereum smart contract	Strong	Proposed a solution to the authenticity of original online published digital works.
	Wang *et al.* [55]	IIPFS, Ethereum and Attribute-Based Encryption (ABE)	Unclear	Proposed a smart contract-based access control mechanism for decentralized storage systems.
	Naz *et al.* [56]	IPFS, Smart contract, RSA signatures	Strong	Proposed an IPFS-based secure data sharing framework to deliver digital assets.
	Nyaletey *et al.* [41]	IPFS, Hyperledger Fabric	Strong	Proposed a solution BlockIPFS to trace the access events and provide audit access to files stored on IPFS.
	Huang *et al.* [31]	Ethereum, local databases	Strong	Proposed an Ethereum-based access control and trustworthiness protection for multiple-domain participants.
Peer Anonymity	Zerocoin [57]	Bitcoin laundry system, zero-knowledge proof	Limited	Offered a limited anonymity to the Bitcoin account addresses based on zero-knowledge proof.
	E-Voting System [58]	Bitcoin laundry system, Zerocoin	Limited	Provided an electronic-voting system based on Zerocoin, aiming to solve the privacy issues in original Bitcoin.
	Zerocash [59]	Zero-knowledge Argument, decentralized anonymous payment scheme	Strong	Provided a strong anonymous transactions by covering up the origins, destinations and the total amount of a payment.
	Mixcoin [60]	Accountable mixes, an independent cryptographic accountability layer	Strong	Proposed an anonymous payment protocol that can be deployed immediately with no changes to Bitcoin.
	ReportCoin [61]	IPFS, blockchain	Strong	Proposed a blockchain-based reporting system for the management of smart city.

privacy preserving towards IoT data, Muhammad *et al.* [53] proposed a *modular consortium* architecture by combining the techniques of IoT and blockchains. The proposed architecture can provide decentralized management for IoT data by exploiting the advantages of blockchain and IPFS. Nizamuddin *et al.* [54] studied the authenticity of online digital and multimedia content. To provide the originality proof, authors proposed an authenticity solution based on IPFS and smart contracts. Based on IPFS, Ethereum and attribute-based encryption (ABE) technologies, Wang *et al.* [55] investigated the data storage and sharing mechanism for distributed storage framework, in which no trusted private-key-generator is required. To achieve the fine-grained access control, a data owner can distribute secret keys for other users, and encrypt his data under a certain access policy. Then, towards transparency and quality of data, Naz *et al.* [56] proposed a secure digital-asset sharing framework based on integrated technology by combining IPFS, blockchain and encryption mechanisms. Next, Huang *et al.* [31] proposed an Ethereum-based network-view sharing platform, which can bring global trustworthiness for multiple domains such as different IoT domain networks. In particular, the domain view of each partner is stored in their local databases, while the Ethereum-based system provides the access control and trustworthiness protection over all participants.

2) PEER ANONYMITY

The privacy preservation of blockchain peers attracts particular attention in recent years. For example, considering that the original Bitcoin system has significant limitations on the privacy of Bitcoin peers, Miers *et al.* [57] proposed *Zerocoin*, which enables a limited anonymity to the Bitcoin account addresses based on zero-knowledge proof. However, the proposed Zerocoin cannot guarantee the full anonymity because at least the number of minted and spent coins, and the denomination of transactions are visible to all users of this system. Using *Zerocoin*, Takabatake *et al.* [58] then proposed a new Bitcoin laundry middleware for Bitcoin. In this middleware, authors mentioned that the origin of transactions can be hidden and miners are able to validate transactions without signatures. However, the destination of a transaction and the amount of a payment are still exposed to users. Thus, the proposed e-voting system is also with a limited anonymity. Moreover, the execution speed of this voting system is also an obstacle that is hard to address. To address this problem, *Zerocash* [59] is claimed to fulfill a strong anonymity for payments, because it hides the transaction amount and the values of user-held coins, by invoking the Zero-Knowledge Succinct Non-interactive Arguments of Knowledge (*ZK-SNARKs*) [62]. *Mixcoin* [60] provided a combined service that transfers funds from multiple source addresses to multiple destination addresses. Thus, the relationship between two accounts is hard to be revealed. Zou *et al.* [61] studied an incentive anonymous reporting mechanism based on blockchain and IPFS. The accounts and transactions stored in ReportCoin are open, transparent, and tamper-resistant. Thus, the anonymity of reporting sources can be protected with a high guarantee. The proposed ReportCoin was only evaluated through simulations, which make this work less convincing. The practicality of the proposed incentive mechanism requires more convincing proofs by real implementations.

V. OPEN ISSUES, CHALLENGES AND FUTURE DIRECTIONS

In this section, we discuss open issues, challenges and future directions of distributed file systems with respect to 4 perspectives: *Scalability*, *Privacy*, *Applications* and *Big Data*.

A. SCALABILITY ISSUES

1) SCALABILITY PERFORMANCE

We have reviewed some representative studies [39] related to the scalability performance measurement of DFSs in previous section. These existing works have shown us some insights of DFSs. For example, Wennergren *et al.* [39] mentioned that the limited bandwidth of each instance of IPFS could be one of the critical reasons for the low scalability of IPFS. The quantitative analysis [40] of systematic evaluations towards IPFS storage system indicates that downloading and resolving operations could be bottlenecks while IPFS clients are reading objects from remote nodes. Nyaletey *et al.* [41] evaluated the scalability of the proposed BlockIPFS by varying the number of nodes. However, the scale of their experiments is too small, making the scalability performance of their system unclear under a very large-scale deployment.

Through the studies [39], [40], [44], we see that the current distributed file systems, such as IPFS and Storj, are still in their immature stages. For example, IPFS still faces some notable shortcomings, including the bottlenecks of resolving and downloading, and the high latency of I/O operations. Thus, to achieve the large-scale commercial applications, IPFS must solve a number of challenges such as storage optimization, geo-distributed deployment of nodes, and file request performance, etc.

On the storage-optimization perspective, although the conventional Erasure coding Zigzag codes [43] can be used to improve the storage efficiency for the proposed IPFS-based systems, some open issues should not be ignored. For example, reconstructing original files could bring a high consumption of both disk I/O and bandwidth to some associated peer nodes.

Another critical problem that IPFS needs to address is how to update the contents already stored on its system. This is because all data stored in the IPFS network is a series of hash addresses. Once a change occurs on a file stored in IPFS, the hash address changes, too. Therefore, an efficient update mechanism should be developed for IPFS.

Finally, to improve the scalability of blockchain-based DFSs, we believe that to develop new solutions that can improve the efficiency of DFS's structure layers can be a promising direction. We wish to see the related studies will be proposed soon.

2) PERFORMANCE MEASUREMENT METHODOLOGY

The performance measurement of IPFS and Swarm considering Quality of Service (QoS) metrics still need to be further conducted widely and deeply in future. Especially when integrating them into business models, users desire to know which one (either IPFS or Swarm) matches their requirements best. Fortunately, Zheng *et al.* [63] proposed a real-time performance monitoring framework for blockchain systems. This work has evaluated four famous blockchain systems, i.e., Ethereum [12], Parity [64], Cryptape Inter-enterprise Trust Automation (CITA) [65] and Hyperledger Fabric [66], with respect to the QoS metrics of *transactions per second*, *average response delay*, *transactions per CPU*, *transactions per memory second*, *transactions per disk I/O* and *transactions per network data*. Such comprehensive performance evaluation results give us insightful viewpoints over the 4 well-known blockchain systems. Their experimental logs and technique report [67] can be found from `http://xblock.pro`. In addition, Curran and de Graaff [68] mentioned that they plan to analyze the performance of IPFS while a website is under an unexpected surge of visitors. However, we cannot find the subsequent technique report of their measurements.

3) SYSTEM MEASUREMENT STANDARDS

Based on the existing studies aforementioned, new system measuring standards need to be proposed for IPFS and Swarm. Generally, the system testing can be separated into two phases [69]: a standardization phase and a testing phase. In the former phase, a series of metrics have been designed to show the performance of systems in terms of Transactions Per Second, Contract Execution Time and Consensus-Cost Time. In the latter phase, systems are tested in different situations. For example, failures including network shutdown and high memory occupation could be injected. Then, the designed metrics could show the performance under different failures, which can help identify different types of failures. Furthermore, the transaction amount that are received by a blockchain system in one second could be adjusted in a testing environment. Thus, the system performance under different transaction rates could be measured.

Through the further review described above, we see that the system measurement of distributed file systems is still in its immature stage. Thus, we look forward to seeing exciting new studies on this topic.

B. PRIVACY AND SECURITY ISSUES

Some current versions of DFS such as IPFS, do not tolerate Byzantine attacks. For instance, every peer can access every file stored on IPFS as long as it joins in the system. This situation makes privacy and security issues are weaknesses for IPFS systems. Therefore, to import some privacy-protection means such as smart contract-based Access Control mechanisms [31] and encryption technologies [55] over the data stored on blockchain-based DFSs could be feasible solutions.

In addition, researchers are also considering that will Reed-Solomon erasure coding [46] be implemented for IPFS. Note that, Reed-Solomon coding is very popular in the datacenters as they provide great disk-savings against data replication. IPFS has not yet addressed such data replication problem. On the other hand, adopting such erasure coding can also enhance the privacy and security level for DFSs. This is because each data chunk is encoded under an erasure coding, even if a peer gets a chunk, it doesn't know what the content is. Furthermore, if a malicious attacker outside a DFS intends to eavesdrop from the DFS peers, the attacker must have all encoded pieces of data chunks associated to the desired file. It would be very difficult if the malicious attacker is blocked by an access-control mechanism.

In summary, we anticipate to see new solutions regarding data privacy & security of IPFS are going to be implemented in near future.

C. APPLICATION ISSUES

IPFS is providing business solutions to enterprises. A growing number of applications based on IPFS have been developed. According to the original design, IPFS is used for storing data. For example, Jia *et al.* [70] developed a decentralized music-sharing platform called *Opus* employing both IPFS and Ethereum. Opus provides encrypted storages using IPFS. The keys of these encrypted data are traded using smart contracts. Opus is also able to prevent monopoly of streaming platforms, track the digital ownership of artists and compensate artists with reasonable monetary price. Not only playing as a game-changer in music domain, IPFS has been adopted by other areas. For instance, Tenorio-Forn *et al.* [71] proposed a decentralized publication system for open-access science based on IPFS. Their proposed distributed systems can record reviewers' reputation, and handle the transparent governance processes.

Recently, the IPSE team [72] proposed a new revolutionary search engine, which is implemented on top of IPFS and blockchain. Such IPSE focuses on user privacy and search efficiency, because it allows users to search network files on IPFS and access to the file without relying on a centralized entity such as Google or Baidu. More importantly, IPSE also enables users to take full control of their own network data by exploiting encryption technologies and smart contracts. Thus, IPSE is a good example that integrates a distributed file system with Blockchain technologies.

It can be seen that most of these applications leverage the decentralized characteristics of IPFS. With the integration of Filecoin, smart contracts are imported into IPFS. This new feature brings a great potential to IPFS. Thus, smart contracts make the application development based on IPFS or Swarm a promising direction.

D. BIG DATA ISSUES

IPFS and Swarm can be also well combined with big data applications. We discuss the big data issues considering the following two aspects: *big data storage* and *big data analytics*.

On one hand, regarding big data storage, IPFS and Swarm can store data with their decentralized and secure characteristics. For example, Confais *et al.* [73] proposed an object store for Fog and Edge Computing using IPFS and Scale-out Network Attached Storage systems (NAS) [74]. The proposed system alleviated the issues of high latency of cloud computing architecture and thus is suitable for the Internet of Things (IoT). According to [75], in the era of the fifth Generation Communications Network (5G), more IoT facilities require larger and more secure storages. To meet this requirement, the blockchain-based distributed file systems such as Swarm and IPFS can play an important role as the secure storage layer for IoT.

On the other hand, with respect to big data analytics, the transactions on blockchains and the logs in file systems can be used for data analytics. For example, the analytics of transactions collected from blockchain systems can be used to extract the trading patterns of users. The data analytics of peer's credit is also useful when deciding whether to sign deals with peers. As representative works of data analytics, Chen *et al.* [76]–[78] analyzed a large-scale of smart contracts collected from Bitcoin and Ethereum. The authors then successfully detected a large number of market manipulations and *Ponzi Schemes* [79] using data mining and machine learning methods. Their studies can be viewed as a pioneer on combining big data analytics with blockchains. The technique reports, datasets and even data-analytics codes [80] can be downloaded from `http://xblock.pro`. Using similar approaches, transactions and other data in blockchain networks can be analyzed such that malicious peers and potential attacks existing in distributed file systems can be detected.

Since we have not found any further studies related to the big data issues, we believe that this topic will become a very promising direction for the research community of blockchain-based distributed file systems.

VI. CONCLUSION

The new generation of blockchain-based distributed file systems, such as IPFS and Swarm, have shown their great potentials with their key characteristics: novel solutions of incentive, low-latency data retrieval, automated auditing, and censorship-resistant, etc. This paper first presents the rationale, layered structure and an overview of blockchain-based distributed file systems, particularly focusing on IPFS and Swarm systems. Then, we review the cutting-edge studies, and reveal a series of challenges that constrain their development. Open issues and future directions are also discussed. We believe that the blockchain-based distributed file systems will become very promising solutions for the next-generation websites and data-sharing platforms. We anticipate that this article can trigger blooming investigations on blockchain-based distributed file systems.

REFERENCES

[1] M. Giesler and M. Pohlmann, "The anthropology of file sharing: Consuming Napster as a gift," in *ACR North American Advances in Consumer Research*, vol. 30, P. A. Keller and D. W. Rook, Eds. Valdosta, GA, USA: Association for Consumer Research, 2003, pp. 273–279.

[2] M. Ripeanu, "Peer-to-peer architecture case study: Gnutella network," in *Proc. 1st Int. Conf. Peer-Peer Comput.*, 2001, pp. 99–100.

[3] N. S. Good and A. Krekelberg, "Usability and privacy: A study of kazaa P2P file-sharing," in *Proc. Conf. Hum. Factors Comput. Syst. (CHI)*, 2003, pp. 137–144.

[4] H.-W. Tseng, Q. Zhao, Y. Zhou, M. Gahagan, and S. Swanson, "Morpheus: Creating application objects efficiently for heterogeneous computing," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 53–65.

[5] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "The bittorrent P2P file-sharing system: Measurements and analysis," in *Proc. Int. Workshop Peer-Peer Syst.* Berlin, Germany: Springer, 2005, pp. 205–216.

[6] J. E. Cater and J. Soria, "The evolution of round zero-net-mass-flux jets," *J. Fluid Mech.*, vol. 472, pp. 167–200, Dec. 2002.

[7] J. Benet, "IPFS–content addressed, versioned, P2P file system," 2014, *arXiv:1407.3561*. [Online]. Available: http://arxiv.org/abs/1407.3561

[8] V. Trón, A. Fischer, and Nagy, "State channels on swap networks: Claims and obligations on and off the blockchain (tentative title)," Ethersphere Orange Papers, Ethersphere, Tech. Rep., 2016.

[9] S. Wilkinson, T. Boshevski, J. Brandoff, and V. Buterin, "Storj a peer-to-peer cloud storage network," 2018. [Online]. Available: https://storj.io/storj.pdf

[10] *PPIO: A Decentralized Programmable Storage and Delivery Network*. Accessed: Jan. 2020. [Online]. Available: https://www.pp.io/docs/

[11] J. Benet and N. Greco, "Filecoin: A decentralized storage network," Protoc. Labs, San Francisco, CA, USA, Tech. Rep., 2018.

[12] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, Apr. 2014.

[13] S. Wilkinson, J. Lowry, and T. Boshevski, "Metadisk a blockchain-based decentralized file storage application," Storj Labs Inc., Atlanta, GA, USA, Tech. Rep., 2014, pp. 1–11.

[14] M. Szydlo, "Merkle tree traversal in log space and time," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2004, pp. 541–554.

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

[16] G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche, "Keccak," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Springer, 2013, pp. 313–314.

[17] *LibP2P*. Accessed: Jan. 2020. [Online]. Available: https://github.com/libp2p

[18] *DevP2P*. Accessed: Feb. 2020. [Online]. Available: https://github.com/ethereum/devp2p

[19] M. J. Freedman, E. Freudenthal, and D. M. Eres, "Democratizing content publication with coral," in *Proc. Conf. Symp. Netw. Syst. Design Implement.*, 2004, p. 18.

[20] *Distributed Preimage Archive of Swarm*. Github Archive, Nov. 2019. [Online]. Available: https://github.com/ethereum/go-ethereum/wiki/Swarm—distributed-preimage-archive

[21] *Expected Consensus*. Github Archive, 2019. [Online]. Available: http://www.ipfs.cn/news/info-100327.html

[22] I. Baumgart and S. Mies, "S/kademlia: A practicable approach towards secure key-based routing," in *Proc. Int. Conf. Parallel Distrib. Syst.*, 2007, pp. 1–8.

[23] M. J. Freedman and D. Maziéres, "Sloppy hashing and self-organizing clusters," in *Proc. Int. Workshop Peer-Peer Syst.*, 2003, pp. 45–55.

[24] S. Shalunov, G. Hazel, J. Iyengar, and M. Kuehlewind, "Low extra delay background transport," Internet-draft, Internet Engineering Task Force, Tech. Rep., 2010.

[25] R. Stewart, Q. Xie, and M. C. Allman, *Stream Control Transmission Protocol (SCTP): A Reference Guide: A Reference Guide*. Reading, MA, USA: Addison-Wesley, 2001.

[26] A. Chockalingam and G. Bao, "Performance of TCP/RLP protocol stack on correlated fading DS-CDMA wireless links," *IEEE Trans. Veh. Technol.*, vol. 49, no. 1, pp. 28–33, 1998.

[27] S. K. Kim, Z. Ma, S. Murali, J. Mason, A. Miller, and M. Bailey, "Measuring ethereum network peers," in *Proc. Internet Meas. Conf.*, 2018, pp. 91–104.

[28] *The Homepage of GIT*. Accessed: Mar. 2020. [Online]. Available: https://git-scm.com/

[29] A. Tridgell and P. Mackerras, "The Rsync algorithm," Tech. Rep. TR-CS-96-05, 1996.

[30] A. Z. Broder, "Some applications of Rabin's fingerprinting method," in *Sequences II*. New York, NY, USA: Springer, 1993, pp. 143–152.

[31] H. Huang, S. Zhou, J. Lin, K. Zhang, and S. Guo, "Bridge the trustworthiness gap amongst multiple domains: A practical blockchain-based approach," in *Proc. 11th IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.

[32] *Attacks of IPFS*. Accessed: Dec. 2019. [Online]. Available: http://www.ipfs.cn/news/info-100379.html

[33] Y. Hirai, "Defining the Ethereum virtual machine for interactive theorem provers," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Cham, Switzerland: Springer, 2017, pp. 520–535.

[34] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2016, pp. 3–16.

[35] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld, "Proof of activity: Extending bitcoin's proof of work via proof of stake [extended abstract]y," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 42, no. 3, pp. 34–37, Dec. 2014.

[36] *Design Rationale of Ethereum*. Accessed: Dec. 2019. [Online]. Available: https://github.com/ethereum/wiki/wiki/Design-Rationale

[37] S. Wilkinson, T. Boshevski, J. Brandoff, J. Prestwich, G. Hall, P. Gerbes, P. Hutchins, C. Pollard, and V. Buterin. *Storj a Peer-to-Peer Cloud Storage Network (Version 2.0)*. Accessed: Apr. 2018. [Online]. Available: https://storj.io/storjv2.pdf

[38] H. Shacham and B. Waters, "Compact proofs of retrievability," *J. Cryptol.*, vol. 26, no. 3, pp. 442–483, Jul. 2013.

[39] O. Wennergren, M. Vidhall, and J. Sörensen, "Transparency analysis of distributed file systems: With a focus on interplanetary file system," Tech. Rep., 2018.

[40] J. Shen, Y. Li, Y. Zhou, and X. Wang, "Understanding I/O performance of IPFS storage: A client's perspective," in *Proc. Int. Symp. Qual. Service (IWQoS)*, 2019, pp. 1–10.

[41] E. Nyaletey, R. M. Parizi, Q. Zhang, and K.-K.-R. Choo, "BlockIPFS-blockchain-enabled interplanetary file system for forensic and trusted data traceability," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, Jul. 2019, pp. 18–25.

[42] Y. Chen, H. Li, K. Li, and J. Zhang, "An improved P2P file system scheme based on IPFS and blockchain," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 2652–2657.

[43] I. Tamo, Z. Wang, and J. Bruck, "ZigZag codes: MDS array codes with optimal rebuilding," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1597–1616, Mar. 2012.

[44] R. Norvill, B. B. Fiz Pontiveros, R. State, and A. Cullen, "IPFS for reduction of chain size in Ethereum," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Jul. 2018, pp. 1121–1128.

[45] *IPFS & SWARM*. Accessed: Dec. 2019. [Online]. Available: https://github.com/ethersphere/swarm/wiki/IPFS-&-SWARM

[46] J. S. Plank, "A tutorial on Reed–Solomon coding for fault-tolerance in RAID-like systems," *Softw., Pract. Exper.*, vol. 27, no. 9, pp. 995–1012, Sep. 1997.

[47] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," DRAFT Version 0.5.9.1, Accessed: Jan. 14, 2016. [Online]. Available: https://lightning.network

[48] J. Poon and V. Buterin, "Plasma: Scalable autonomous smart contracts," White Paper. Accessed: 2017. [Online] Available: https://plasma.io/plasma.pdf

[49] D. Vorick and L. Champine, "SIA: Simple decentralized storage," Nebulous Inc., Boston, MA, USA, Tech. Rep., 2014.

[50] H. Huang, S. Guo, W. Liang, K. Wang, and Y. Okabe, "Coflow-like online data acquisition from low-earth-orbit datacenters," *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/TMC.2019.2936202.

[51] G. Fanti and P. Viswanath, "Deanonymization in the bitcoin P2P network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1364–1373.

[52] M. Steichen, B. Fiz, R. Norvill, W. Shbair, and R. State, "Blockchain-based, decentralized access control for IPFS," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Jul. 2018, pp. 1499–1506.

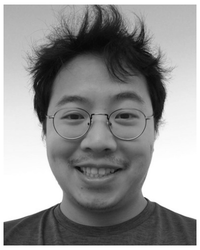
[53] M. S. Ali, K. Dolui, and F. Antonelli, "IoT data privacy via blockchains and IPFS," in *Proc. 7th Int. Conf. Internet Things (IoT)*, 2017, p. 14.

[54] N. Nizamuddin, H. R. Hasan, and K. Salah, "IPFS-blockchain-based authenticity of online publications," in *Proc. Int. Conf. Blockchain*. Cham, Switzerland: Springer, 2018, pp. 199–212.

[55] S. Wang, Y. Zhang, and Y. Zhang, "A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems," *IEEE Access*, vol. 6, pp. 38437–38450, 2018.

[56] M. Naz, F. A. Al-zahrani, R. Khalid, N. Javaid, A. M. Qamar, M. K. Afzal, and M. Shafiq, "A secure data sharing platform using blockchain and interplanetary file system," *Sustainability*, vol. 11, no. 24, p. 7054, 2019.
[57] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed E-cash from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, May 2013, pp. 397–411.
[58] Y. Takabatake, D. Kotani, and Y. Okabe, "An anonymous distributed electronic voting system using Zerocoin," IEICE Technique Rep. IA2016-54, 2016.
[59] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, May 2014, pp. 459–474.
[60] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten, "Mixcoin: Anonymity for bitcoin with accountable mixes," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Berlin, Germany: Springer, 2014, pp. 486–504.
[61] S. Zou, J. Xi, S. Wang, Y. Lu, and G. Xu, "Reportcoin: A novel blockchain-based incentive anonymous reporting system," *IEEE Access*, vol. 7, pp. 65544–65559, 2019.
[62] H. Lipmaa, "Prover-efficient commit-and-prove zero-knowledge SNARKs," in *Proc. Int. Conf. Cryptol. Africa*. Cham, Switzerland: Springer, 2016, pp. 185–206.
[63] P. Zheng, Z. Zheng, X. Luo, X. Chen, and X. Liu, "A detailed and real-time performance monitoring framework for blockchain systems," in *Proc. 40th Int. Conf. Softw. Eng. Softw. Eng. Pract. (ICSE-SEIP)*, 2018, pp. 134–143.
[64] *Parity Documentation*. Accessed: Mar. 2020. [Online]. Available: https://paritytech.github.io/wiki
[65] *Cita Technical Whitepaper*. Accessed: Feb. 2020. [Online]. Available: https://github.com/cryptape/cita
[66] *Hyperledger Fabric Website*. Accessed: Mar. 2020. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-1.4/write_first_app.html
[67] Xblock. (Feb. 2020). *Performance Monitoring*. [Online]. Available: http://xblock.pro/performance/
[68] T. Curran and B. de Graaff, "Analysing the performance of IPFS during flash crowds," Tech. Rep., 2016. Accessed: Feb. 2016. [Online]. Avaliable: https://www.os3.nl/_media/2015-2016/courses/lia/projects/tom-ipfs_proposal.pdf
[69] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data (BigData Congress)*, Jun. 2017, pp. 557–564.
[70] B. Jia, C. Xu, R. Gotla, S. Peeters, R. Abouelnasr, and M. Mach, "Opus-decentralized music distribution using InterPlanetary File Systems (IPFS) on the Ethereum blockchain V0. 8.3," Tech. Rep., 2016. Accessed: Jan. 2017. [Online]. Available: https://icosbull.com/whitepapers/1196/Opus_whitepaper.pdf
[71] A. Tenorio-Fornés, V. Jacynycz, D. Llop-Vila, A. Sánchez-Ruiz, and S. Hassan, "Towards a decentralized process for scientific publication and peer review using blockchain and IPFS," in *Proc. 52nd Hawaii Int. Conf. Syst. Sci.*, 2019, pp. 4635–4644.
[72] IPSE Team. *IPSE: A Search Engine Based on IPFS*. Accessed: Dec. 2019. [Online]. Available: https://ipfssearch.io/IPSE-whitepaper-en.pdf
[73] B. Confais, A. Lebre, and B. Parrein, "An object store for Fog infrastructures based on IPFS and a Scale-Out NAS," in *Proc. RESCOM*, 2017, p. 2.
[74] G. A. Gibson and R. Van Meter, "Network attached storage architecture," *Commun. ACM*, vol. 43, no. 11, pp. 37–45, Nov. 2000.
[75] I. Jovović, S. Husnjak, I. Forenbacher, and S. Maček, "5G, blockchain and IPFS: A general survey with possible innovative applications in industry 4.0," in *Proc. 3rd EAI Int. Conf. Manage. Manuf. Syst.*, 2018, pp. 1–10.
[76] W. Chen, Z. Zheng, J. Cui, E. Ngai, P. Zheng, and Y. Zhou, "Detecting Ponzi schemes on Ethereum: Towards healthier blockchain technology," in *Proc. World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 1409–1418.
[77] W. Chen, J. Wu, Z. Zheng, C. Chen, and Y. Zhou, "Market manipulation of bitcoin: Evidence from mining the Mt. Gox transaction network," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 964–972.
[78] W. Chen, Z. Zheng, E. C.-H. Ngai, P. Zheng, and Y. Zhou, "Exploiting blockchain data to detect smart Ponzi schemes on Ethereum," *IEEE Access*, vol. 7, pp. 37575–37586, 2019.
[79] *Ponzi Scheme*. Accessed: Mar. 2020. [Online]. Available: https://en.wikipedia.org/wiki/Ponzi_scheme
[80] XBlock. (Feb. 2020). *Fraud Detection*. [Online]. Available: http://xblock.pro/fraud-detection/

HUAWEI HUANG (Member, IEEE) received the Ph.D. degree in computer science and engineering from the University of Aizu, Japan, in 2016. He is currently an Associate Professor with the School of Data and Computer Science, Sun Yat-sen University, China. His research interests include blockchain and intelligent distributed computing. He has served a Research Fellow (2016–2018) of JSPS, a Visiting Scholar (2017–2018) with The Hong Kong Polytechnic University, and an Assistant Professor (2018–2019) with Kyoto University, Japan. He received the Best Paper Award from TrustCom2016.



JIANRU LIN (Member, IEEE) is currently a Visiting Researcher with the School of Data and Computer Science, Sun Yat-sen University, China. His research interests include consensus protocols and blockchain.

BAICHUAN ZHENG received the bachelor's degree from the School of Data and Computer Science, Sun Yat-sen University, in 2019. He is currently doing research on new consensus mechanism and database of blockchain.

ZIBIN ZHENG (Senior Member, IEEE) received the Ph.D. degree from The Chinese University of Hong Kong, in 2011. He is currently a Professor with the School of Data and Computer Science, Sun Yat-sen University, China. He serves as the Chairman of the Software Engineering Department. He has published over 120 international journal and conference papers, including three ESI highly cited papers. According to Google Scholar, his papers have more than 9600 citations, with an H-index of 47. His research interests include blockchain, services computing, software engineering, and financial big data. He was a recipient of several awards, including the Top 50 Influential Papers in Blockchain of 2018, the ACM SIGSOFT Distinguished Paper Award at ICSE2010, and the Best Student Paper Award at ICWS2010. He has served as BlockSys'19 and CollaborateCom'16 General Co-Chair, SC2'19, ICIOT'18, and IoV'14 PC Co-Chair.

JING BIAN received the B.Sc. degree in automation, the M.Sc. degree in computational mathematics, and the Ph.D. degree in physics from Sun Yat-sen University, Guangzhou, China, in 1988, 2001, and 2006, respectively. She is currently a Vice-Professor with the School of Data and Computer Science, Sun Yat-sen University. Her current research interests include design and analysis of algorithms, blockchain, electronic commerce, and social networks.