Hindawi
Journal of Sensors
Volume 2020, Article ID 8861688, 11 pages
https://doi.org/10.1155/2020/8861688

Research Article

On the Exploitation of Blockchain for Distributed File Storage

Zuoting Ning,[1] Lijun Xiao,[2] Wei Liang,[3] Weiqi Shi,[1] and Kuan-Ching Li[4,5]

[1]*Department of Information Technology, Hunan Police Academy, Changsha, Hunan, China*
[2]*Big Data Development and Research Center, Guangzhou College of Technology and Business, Guangzhou, China*
[3]*College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan, China*
[4]*Department of Computer Science and Information Engineering, Providence University, Taichung, Taiwan*
[5]*School of Computer Science and Engineering, Anhui University of Science and Technology, Huainan, China*

Correspondence should be addressed to Lijun Xiao; ljxiaoxy@126.com, Wei Liang; weiliang99@hnu.edu.cn, and Kuan-Ching Li; kuancli@pu.edu.tw

Received 24 March 2020; Revised 10 July 2020; Accepted 19 August 2020; Published 15 December 2020

Academic Editor: Fei Yu

Copyright © 2020 Zuoting Ning et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Distributed file storage aims to support credible access to data on distributed nodes. There are some application scenarios, for example, data centers, peer-to-peer (P2P) storage systems, and storage in wireless networks. Nevertheless, among these applications, data blocks are inevitably replaced and inaccessible when there exists nodes failure. As a result, data integrity and credibility is absent. To overcome such a challenge, blockchain is explored to protect the distributed data. Through analysis and evaluation, we demonstrate that blockchain advocates data integrity and credibility for distributed file storage, as well as the application of blockchain technology for distributed file storage.

1. Introduction

Distributed file storage contributes to storing data over the network by distributed storage nodes. As known, there is an emergence of a large number of applications involving large data centers and peer-to-peer storage systems [1–5]. All these applications utilize nodes over the internet to approach distributed file storing. To obtain reliable storage in networks as wireless sensor networks(WSNs), additional data recovery may be required [4, 6], especially in case of a disastrous environment [5, 7].

In these applications, data reliability demands inevitably for data redundancy. In order to simplify redundancy, replication is a very suitable form, which is commonly utilized in distributed file storage systems. In research of replication, there are various methods to approach data redundancy, while erasure coding gains better storage efficiency performance. Generally, we segment a file of size S into n parts, namely, each part is of size S/n; after that step, we code these parts into m encoded portion by adopting (m, n) maximum distance separable code (MDSC); at last, all coded portion are stored on M nodes. In this way, we can recover the original file from any set of n linearly independent coded parts. As a result, optimal performance can be harvested, such as better decisions for incredibility and redundancy trade-off. There are some researches utilizing erasure codes to reduce data redundancy [6–8].

At present, among many domains, such as academia and industry, they have paid attention to the distributed ledger and blockchain technology as well as its massive potential in managing complicated systems. The distributed ledger is mainly composed of a certain amount of blocks [9], chained back that utilizes a linked hash-pointer list, so data blocks store valid sequential transactions with digital assets (see Figure 1).

Nevertheless, when any data block is attacked, we cannot recover the source file, since all segmented pieces are linearly independent. In order to solve this issue, literature [8] proposed a scheme that aims to reduce computation complexity by probabilistic-ally verifying blocks of data. Liang et al. [10] put forward a recommendation scheme to obtain data credibility. It is depicted that, by using a homomorphic signature,

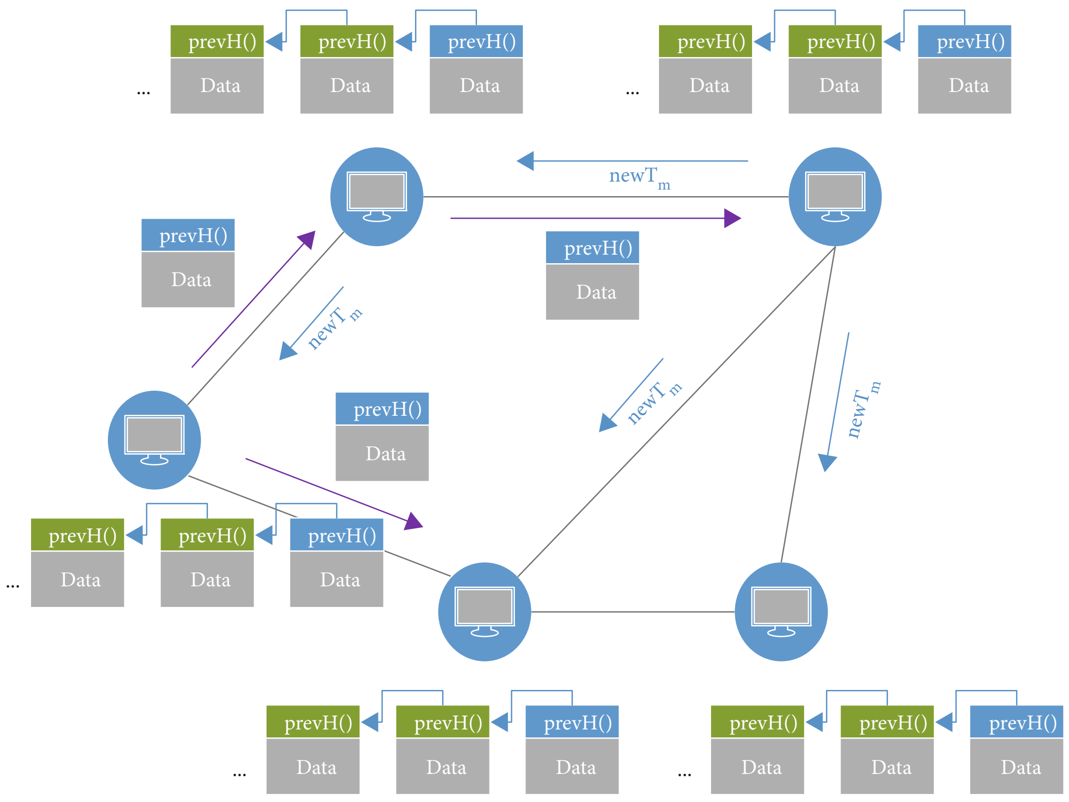

Figure 1: Ledger distribution in peer-to-peer network.

intranetwork verification can harvest data credibility [11, 12], a polynomial-time algorithm is applied to protect networks from malicious attacks [13], and a scheme is designed to resist pollution attacks by utilizing a polynomial hashing function [14].

In this paper, in order to conquer data integrity and incredibility issues, we utilize a blockchain-based approach for distributed file storage. The main contribution of this paper is the following: firstly, we utilized distributed hash table to enhance the credibility for stored files; secondly, we proposed a blockchain-based framework for a distributed file storage system to approach integrity and security; finally, we carried out detailed discussion on the load balance, throughput, and risk of attacks.

2. Background and Related Work

2.1. Distributed File Storage System. Nowadays, there are various types of distributed storage systems, such as cloud storage systems, and peer-to-peer (p2p) storage systems. In all these storage systems, data can be stored, archived, and back up over distributed nodes, such as AmazonS3. Users can make use of their stored files any time anywhere; this is an outstanding advantages as to distributed storage systems. There are many researches focused on the design and construction of distributed file systems. Napster [15], Kazaa [16], and Gnutella [17] implement distributed file systems and prompt it to be an exciting and popular research area. Bit-torrent [18] is one of the most popular and successful peer-to-peer distributed file systems and has more than 100 million online users presently. It is a large-scale deployed in which millions of users log-in and log-out every day. Storage resources, as well as system clients in a distributed file system, are scattered in the network. In these systems, users act as both creators and consumers of data, therefore, to provide massive of incentives by a secure and efficient approach.

Various recent studies have explored and carried out evaluations for distributed file storage systems [2, 6, 19–22]. Some literatures [2, 7, 8, 22–26] proposed and evaluated redundancy management strategies. Among these, replication and erasure codes are compared in bandwidth and reliability trade-off in literatures [2, 7, 8]. Literature [7] argues that, compared with replication, erasure codes can harvest better bandwidth performance. Literature [2] also approves this conclusion through a distributed file storage system. Based on a novel data clustering optimization model, Liang et al. put forward an intrusion detection algorithm for the industrial network [27]. A hybrid strategy is proposed in the literature [8] to conquer the repair problem. The node storing the replication can generate new pieces and then deliver them to the new users. As a result, it only transfers only S/n to a new segment. Nevertheless, supporting an extra replication decreases the bandwidth-performance; that is because when the replication is polluted or lost, new segments cannot be produced. Meanwhile, there exists support [28] for static policies to solve data block replications. These schemes should be manually configured and primarily focus on archival purposes [29], by utilizing disk caching, MixApart [30], and Rhea [31] research data retrieval. What is more, literature [30] schedules tasks by using remote data and local caching, while static analysis strategy to harvest storage performance is utilized [31]. Liang et al. [32] put forward an efficient protocol to approach identity authentication in the IoT environment.

2.2. Blockchain. The blockchain technology is a chain of blocks based on time-stamp that is jointly sustained by each on-chain node. Every block acts as a container role aggregating all on-chain transactions and chained by cryptography technology. That is, each participating block is chained together and signed by their private secret key as well as their respective hash value. Once a new block is created, this new block will be chained together. In this way, the blockchain provides a steady data storage, so any deletion or update on processed transactions is impracticable [33]. Due to this characteristic, we can make full use of this advantage in the proposed work. In other words, all transactions are reliable without a third-party authority. The advantage of blockchain resists all stored data from repudiation. Moreover, user identity and authenticity are guaranteed by cryptography and digital signatures, so thus any illegal read or write will be refused over the blockchain.

Bitcoin, which is regarded as the first practice of the blockchains, is a public distributed ledger; it plays an essential role in promoting blockchain. After that, smart contracts [34] emerged, which is an autonomous program deployed on the blockchain network and makes all transactions intelligently. In the practice of blockchain, smart contracts act the role of triggers [35]. For example, based on the smart contract, all services will not hold funds unless all tasks in the contract have been finished. According to this theory, Ethereum regards and promotes the smart contract to the top level. Nowadays, blockchain has turned to be a promising topic in both industry and academy area, and combining blockchain and distributed file system becomes an exciting and promising solution, in which blockchain provides incentives and security for distributed files. Up to now, the popular and famous distributed file systems are IPDFS [36], Storc [37], Swarmer [38], and PPIO [32]. In these systems, IPDFS is a peer-to-peer distributed file system that is used to store and access files, applications, websites, and data; Storc is another peer-to-peer decentralized cloud storage platform allowing users to share data and has no need of any third-party data provider; Swarmer, based on Ether, is a distributed storage platform and content-distributed service, and PPIO that permits users to store and retrieve data on web anywhere and anytime is a programmable distributed storage network.

With the introduction of the blockchain, three distributed file systems utilize File-coin [39], Ether [40], and Meta-disk [41] as correspondingly stimulative mechanisms. Based on the industrial blockchain network environment, Liang et al. proposed secure data storage and recovery strategy [3], while Zhang et al. utilize the blockchain to improve 5G performance [42].

3. Problem Statement

Due to issues in integrity, trust, control, and credibility, we focus in this paper on overcoming the issue of integrity and credibility for distributed file storage. There are various systems and platforms for distributed file storage, and they aim to collect all kinds of data. Notably, this incurs a severe privacy problem, since most users have no knowledge of these actions, much less about control of such actions. To solve this problem, we suppose in this work that all provided services should obey the smart contracts, especially some assigned protocols. Based on this, this proposed work devotes to the following issues:

Data Credibility. Our research focuses on the data credibility for distributed file storage; we should guarantee that authorized users must control all personal data. Meanwhile, the systems and platforms regard the services as guests who have corresponding permissions.

Data Integrity. All data should be verified and detected to guarantee the integrity of stored data. All data-trace is totally transparent for each authorized user, and any illegal modification is impractical on the platform.

Access Control. Any users should be granted access permission as they log in the system or platform. These permissions should define which resources the users can utilize. Within the permissions, users can change the access range of their stored data. Meanwhile, all participating users must store data access control strategies or policies on the blockchain. Thus, illegal access is hardly impossible.

4. Our Solution

4.1. Distributed System. In this paper, we design a decentralized system. There are mainly three parts comprising the system: nodes, users, and services, as shown in Figure 2. Users can store and utilize their corresponding distributed files, as all operations on distributed data are supported by the services; the nodes play an essential role as storing users' distributed files encrypted with their private keys. In order to simplify user authentication, we produce message digest for each stored file on the chain. In the proposed system, blockchain is very critical, since it only accepts two kinds of data, namely, G_{access} and T_{data}. The former is utilized for access control, while the latter is used for user data storage or data retrieval. The two types of operations can be arranged on the SDK (Software Development Kit), and users can use them through complete services.

To describe the proposed system in details, we assume the following application service: when a user intends to store files on the proposed system, he will first install the system application. The user signs up on the proposed system, and the system will generate an identity for the user, so then this identity will be informed to the blockchain, as well as the user's permissions. The user's files will be segmented into pieces and then encrypted by their shared keys. After that, all segmented files are stored on the nodes in a distributed way. In the meanwhile, all encrypted files are sent to the chain with T_{data}, and a special pointer produced by the hash of segmented files to the blockchain is maintained.

When users issue data query requests, they can use T_{data} together with the aforementioned unique pointer. Once the blockchain receives this request, it will check the identity of the users by verifying users' digital signature. Only when

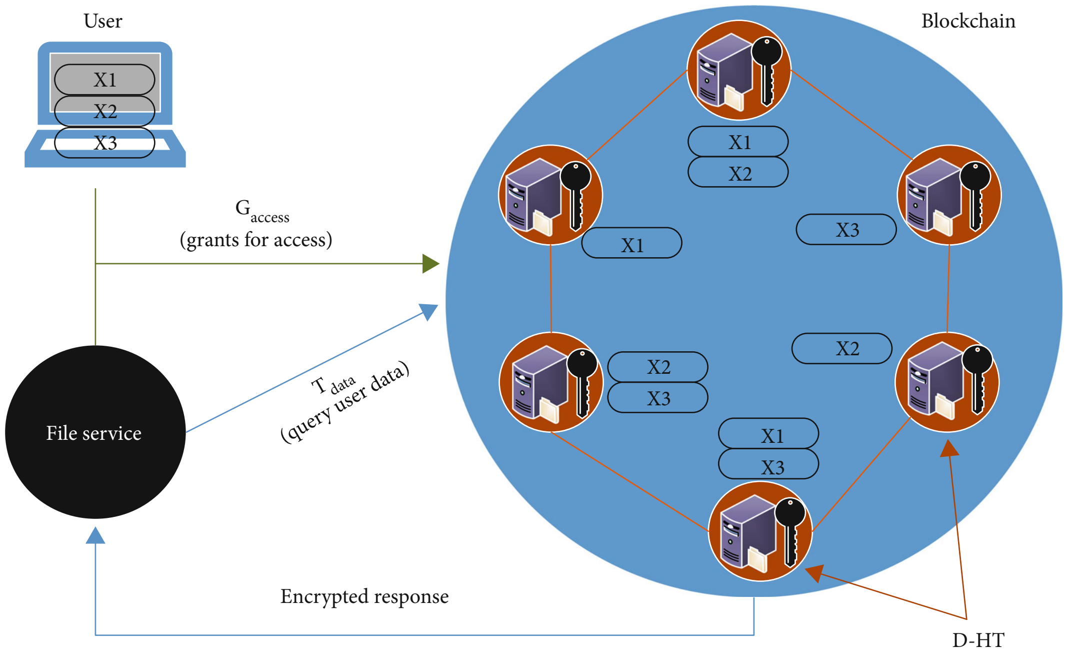

Figure 2: System framework.

the users have passed verification, they can carry out the operation within their authorized permission. They can have an overview of their file data and modify corresponding permissions. All these operations are recorded in the blockchain. We utilized D-HT (Distributed Hash Table) to carry out a key-value store for off-blockchain in this implementation, and it interacts with the blockchain through an interface. During the processing, the D-HT is utilized by the nodes on the chain, so any general operations, such as read and write, shall be approved by D-HT, and thus, users' files can be of high availability.

4.2. Building Blocks. In this subsection, to approach the proposed solution, a detailed description of how to building the blocks follows next. Accordingly, the process of building blocks follows Bitcoin [43].

(1) Identities. To identify users, we utilize a pseudo-identity scheme. That is, each user on the chain can produce pseudo-identity by their public keys, and the practical requirements determine the number of pseudo-identities, which can critically improve the user's privacy. In this work, we explore compound identities that are originated from the existing theory. As there may be more than two participators during transactions, some participators can hold this compound identity, though the remaining have no permissions to use it. As depicted in protocol one, we assume there are only one owner and one guest, and this protocol describes how we implement this operation. To guarantee the credibility, we utilize asymmetric key pairs to authorize user's identity, as to encrypt or decrypt the user's distributed files, we use asymmetric key that can promote the efficiency of encryption and decryption. In this way, all data are secure for each one of the users. The compound identity is defined as the following:

$$Compound_{o,g}^{public} = \left(pk_{sig}^{o,g}, sk_{sig}^{o,g}\right). \tag{1}$$

As to the whole identity, we formulate it as a 5-tuple:

$$Compound_{o,g} = \left(pk_{sig}^{o,g}, sk_{sig}^{o,g}, pk_{sig}^{g,o}, sk_{sig}^{g,o}, sk_{enc}^{o,g}\right). \tag{2}$$

(2) Policy. In this paper, we define a group of permissions, which a data owner o grants a guest g, as POLICY$_{o,g}$. Supposing the following scenarios, if the owner o deploys an application that calls for access to o's location or contacts, this can be denoted as, POLICY$_{o,g} = o_{\text{location}}, o_{\text{contacts}}$. It describes that all types of data should be stored according to this way, supposing the service does not tear up the protocol and error-mark the data, then safeguards, which are utilized to avoid this, should be preferentially recommended to SDK. Moreover, according to this method, every user can quickly verify the legality of service, since any change is fully visible

(3) Auxiliary Functions. In the function $ChkPolicy(pk_{sig}^{sk}, m_p)$, Parse$(m)$ de-serializes these messages that are delivered to a general transaction containing the arguments $ChkPolicy(pk_{sig}^{sk}, m_p)$, as is described in the protocol two. This function validates whether

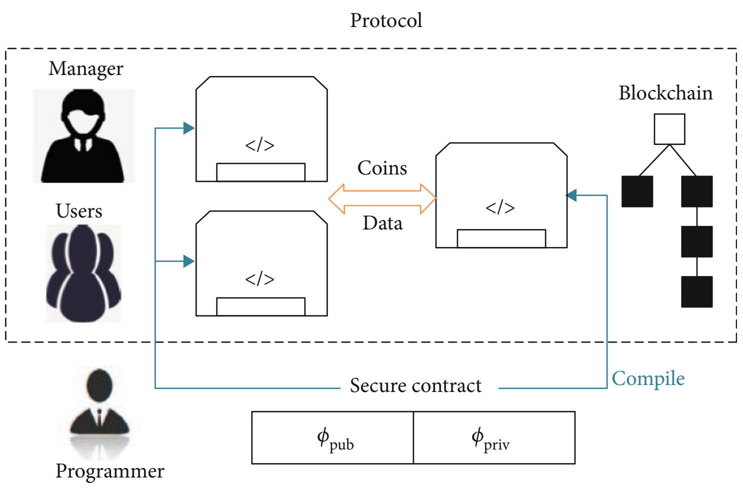

FIGURE 3: Smart secure contract.

the initiator has relevant permissions, which can guarantee the validity of each operation

Protocol 1: *Compound identity description.*

```
1: Procedure CompoundIdentity(o,g)
2:   $o$ and $g$ compose of a secure channel
3:   $u$ executes:
4:     $(pk_{sig}^{o,g}, sk_{sig}^{o,g}) \leftarrow O_{sig}()$
5:     $sk_{enc}^{o,g} \leftarrow O_{enc}()$
6:     $u$ shares $sk_{enc}^{o,g}, pk_{sig}^{o,g}$ with $s$
7:   $s$ executes:
8:     $pk_{sig}^{g,o}, sk_{sig}^{g,o} \leftarrow O_{sig}()$
9:     $g$ shares $pk_{sig}^{g,o}$ with $o$
10:// Both $o$ and $g$ have $sk_{enc}^{o,g}, pk_{sig}^{o,g}, pk_{sig}^{g,o}$
11:   Return $pk_{sig}^{o,g}, pk_{sig}^{g,o}, pk_{sig}^{o,g}$
12: end Procedure
```

Protocol 2: *Verify permission.*

```
1:Procedure:ChkPolicy($pk_{sig}^{sk}, m_p$)
2: $g \leftarrow 0$.
3: $a_{policy}=H(pk_{sig}^{sk})$
4: if $L[a_{policy} \neq \varphi]$ then
5:   $pk_{sig}^{o,g}, pk_{sig}^{g,o}, POLICY_{o,g} \leftarrow Excute(L[a_{policy}])$
6:     if $pk_{sig}^{sk} = pk_{sig}^{o,g}$ or
7:     $(pk_{sig}^{sk} = pk_{sig}^{o,g}$ and $m_p \in POLICY_{o,g})$ then
8:     $g \leftarrow 1$.
9:     endif
10: endif
11: return $s$
12:end Procedure
```

In order to guarantee the validity of each operation, we create a checking-policy function $ChkPolicy(pk_{sig}^{sk}, m_p)$ to verify permission, which simplifies the access verification compared with these existing approaches [22, 35, 39].

4.3. Smart Secure Contracts. In this section, we explore a blockchain-based framework for a distributed file storage system to approach security. As depicted in Figure 3, the framework is composed of two parts:

(1) In contracts, it contains each user's operation data, including variate φ_{private} that denotes user's private data that carries out computation for distributed data. Suppose the following scenario that, during an open auction, only the winner's final bids approach to the seller, and the other bids are totally refused. Thus, variate φ_{private} guarantees the security for users' distributed data

(2) Variate φ_{public}, which has no user's private data, denotes users' public data. Meanwhile, we define the cryptography protocol that is used during the transaction on the chain

Security guarantees. Security guarantees mainly include the following aspects:

(i) Chain-to-Chain Privacy. Chain-to-chain privacy indicates that the user's distributed file or data should be protected against any users not included on the blockchain, only if the legal users intend to inform others of their information. In our proposed protocols, all users should interchange data and depend on blockchain to guarantee fairness. That is, all users transmit their encrypted files or data to the chain, what is more, as also all transactions are based on zero-knowledge authorization

(ii) Security. As chain-to-chain privacy prevents the user's data from the public chain, all users' data are entirely independent of each other. Meanwhile, asymmetric encryption guarantees the authenticity

```
1:Declare Member(Seller/* M parties */)
2:Declare Timeouts(/* timeouts */)
3:Declare Function contract auction(In &input, Out &output)
4:Set win = −1
5:Set btprice = −1
6:Set secdprice = −1 7:loop
8:    for each j < m do
9:        if input.pat[j].value > btprice then
10:           Let secdprice = btprice
11:           Let btprice = input.pat[j]
12:           Let winner = j
13:       else if input.pat[j].value > secdprice then
14:           Let secdprice = input.pat[j].value
15://The winner pays the bidder
16://The others are refused
17:Let output.seller.value = secdprice
18:Let output.pat[win].value = btprice − secdprice
19:Let output.win = win
20:for each j < m do
21:    if j ≠ win then
22:        output.pat[j].value = input.pat[j].value
```

Algorithm 1: Process for public auction.

and confidentiality. We take a public auction as an example to describe the security of the scheme. The above Algorithm 1 shows the process of a public auction. In this example, auction transaction contains φ_{private}, which indicates that who wins the bidder and how much he should pay. Meanwhile, variate φ_{public}, which depends on the deposits, is used to avoid the winners from abandoning

An auction is of the abovementioned specified requirements, especially in terms of security and confidentiality, and this is accomplished by cryptocurrency, as present in some existing systems [44, 45]. The program, as depicted in the algorithm, declares timeout parameters. The timeout parameters are declared as P1 < P2 < P3. P1: the contract stops receiving bids after P1. P2: the bidder should tell the price within time P2; otherwise, its input bid is regarded as 0. By doing this, the auction transaction continues. P3: supposing the auction manager abandons the bid, bidders may withdraw their bids when time P3 elapses.

Variate φ_{public} plays an essential role in the auction transaction. As it not only checks the time but also manage the timeout. The system will invoke the function only if the operation completes within P3. Otherwise, the system will invoke the manager's TimeOut function.

5. Theoretical Analysis

5.1. Credibility. In a blockchain system, it supposed that all nodes should be untrustworthy. That is, every node should be verified. Moreover, nodes' resources for computing determine their credibility level [43]. For example, a node m, $m \propto$ resources (m) denotes that how much weight node m votes, which means that wither the node is vulnerable when there is high energy consumption, or there is high latency of a transaction.

In this paper, as to formulate the value for all nodes' trust, we compute each data block b of a node like the following:

$$rely\_t_m^{(b)} = \frac{1}{1 + e^{(-\beta)(\#legitimate - illegitimate)}}, \qquad (3)$$

in which the step size is defined by β.

In the above equation, it is regarded that these nodes on-chain has higher weight as well as more efficiency in computation. Due to this reason, these nodes have the ability to resist fraud attacks.

5.2. Risk of Attack. In a blockchain, there is the risk of fraud, but this is complex, because it should approach 51%, the risk occurs. However, the risk exists, although hardly impossible to reach such a high percent of nodes failure. The current public blockchain structure is vulnerable to some particular scenarios: the software update, blockchain entry changes, as an example. This is due to, when all transactions are processing, any new participators can have knowledge of the decisions of the network. As to the network, based on its majority rules, 51% of undergoing transactions could do any operation on the chain.

51% attack on the chain may sharply increase the vulnerability, as there may exist fork attack. This is because more than two networks share the resources of a single network, leading to a quick decrease in computation ability. Namely, the cost of launch attack on the networks is lower, and cause to a growing risks for the network.

The probability that an attack chain can catch up with the honest chain is shown in the following equation.

$$q_z = \begin{cases} 1, p \leq q \\ (q/p)^z, p > q \end{cases}. \quad (4)$$

p represents the probability that honest miners find the next block, q is the probability that attackers find the next block, and q_z is the probability that attackers change the trading content of the current blocks.

5.3. Load Balance. In our research, load balancing is an important index. We are aiming at distributing all requests efficiently on each node, and this can harvest great improvement in load balancing. In this paper, any nodes act the role of sustaining as many online I/O connections as possible; we use variate (*NumAct[*s_j*]*) to represent the number of connections, while variate s_j denotes storage media. These two variables should be stored to the nodes within any transactions, correspondingly. Generally, the more the number of connections to the storage media, the lower the proportional of the throughput of the nodes. That is, if we intend to approach the better performance of load balancing, the participating nodes should have fewer connections. For ease of description, we introduce function f_{ub}, and formulate the load balancing as the following:

$$f_{ub}\left(\vec{s}\right) = \sum_{s_j \in \vec{s}} \left(\frac{1}{NumAct\left[s_j\right] + 1}\right). \quad (5)$$

Where function f_{ub} approaches maximum when all storing nodes have the lest count of active connections. Function f_{ub} approaches to the upper bound when the very lest connecting number occurs. This is an exciting discovery for each storage media. As a result, we can optimize the function f_{ub} and harvest the following formulation:

$$f_{ub}^{*}\left(\vec{s}\right) = \left|\vec{s}\right| \times \frac{1}{\min \forall_s (NumAct[s] + 1)}. \quad (6)$$

5.4. Throughput. As to distributed file storage systems, we devote to harvest best throughput performance. We store data in the manner of tiers, so thus, we can make full use of fast storage characteristics of tiers, which helps harvest optimized throughput. Once there is a request, the system will check the ability level for reading and write throughput of storing node node$_\mathrm{i}$ by a quick I/O test, and the read and write throughput is denoted as *ReadTh[node*$_i$*]*, *WriteTh[node*$_i$*]*, respectively. After that, we calculate the average value and store them on the node.

To approach the maximum throughput, any operation for distributed file storage is of the optimal write or read throughput. Generally, to obtain the common value, we regard the proportional peak value of nodes' throughput as the final value. Moreover, in order to scale the throughput down, we introduce the logarithm function for these throughput values.

Table 1: Theoretical results.

	Declare-or-refuse [46]	Multiple lock [47]	Blockchain
Chain-to-chain cost	$O(P^2)$	$O(P^2)$	$O(P)$
Amount of rounds	$O(P)$	$O(1)$	$O(1)$

Similarly, we formulate the throughput function f_{tm} as:

$$f_{tm}\left(\overrightarrow{node}\right) = \sum_{node_i \in \overrightarrow{node}} \left(\frac{\log\left(WriteTh[node_i]\right)}{\log\left(\max \forall_{node} WriteTh[node]\right)}\right). \quad (7)$$

As to the storage nodes, by computing function f_{tm}, we retrieve the throughput for storage node, when a specified number of nodes with the optimal throughput are on the chain. Once there are always many nodes with optimal throughput, the function approaches the upper bound. Therefore, we can optimize the function f_{tm}^{*} and formulate it as:

$$f_{tm}^{*}\left(\overrightarrow{node}\right) = \left|\overrightarrow{node}\right|. \quad (8)$$

5.5. Theoretical Results. In Table 1, we assume that there are P participators who intend to calculate a one-bit outcome and send it to all P participators. We made comparisons among literature [46, 47], and blockchain, and we conclude that blockchain is most useful for distributed file storage. Public storage in the blockchain-based system was first approved in the literature [48]. Fairness can hardly be impractical in general models for multiparticipator transactions, which is proposed in the literatures [49, 50]. Base on the script language, some works about construct abstractions for protocols emerge, for example, "Declare-or-refuse" [46] or "multiple locks" [47].

5.6. Blockchain Application Systems. Table 2 shows four blockchain-based application systems, and we make a comparison in terms of blockchain form, protocol, cryptocurrency, and intelligent contracts. Super-ledger [51], which is an open-source system based on blockchain, was developed to improve the efficiency of distributed file storage. It was developed by superior language and supported any application on the chain, and meanwhile, it supported distributed components and maintained membership.

The multiple chain [52] system aims to create the private key for users, as well as deploy the blockchain. It depends on the API to expand the core API, which permits managing all transactions, assets, and resources. This system has good operationality for users to interact with networks, such as users can directly utilize command tools, and distributed clients can carry out transactions with the network by JSON, especially Ruby, Node.js, and Cij. This characteristic makes this system have excellent convenience of operation.

Table 2: Blockchain application systems.

System	Blockchain	Agreement	Crypto currency	Intelligent contracts
Super-ledger	Based on permission	SIEVE	No	Yes
Multiple chain	Based on permission	PBTF	Multiple currencies	Yes
ETH	Public chain	PoS	Ether	Yes
LTC	Public chain	Scrypt	LTC	No

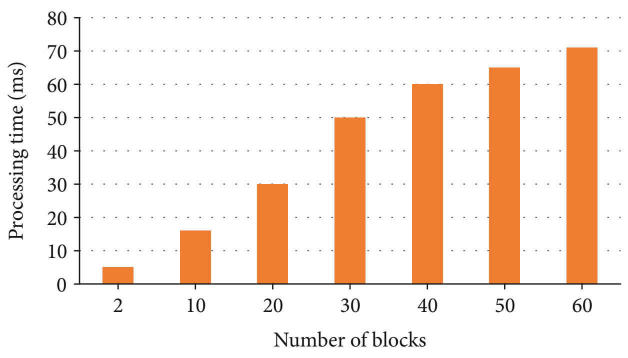

Figure 4: Processing time overhead.

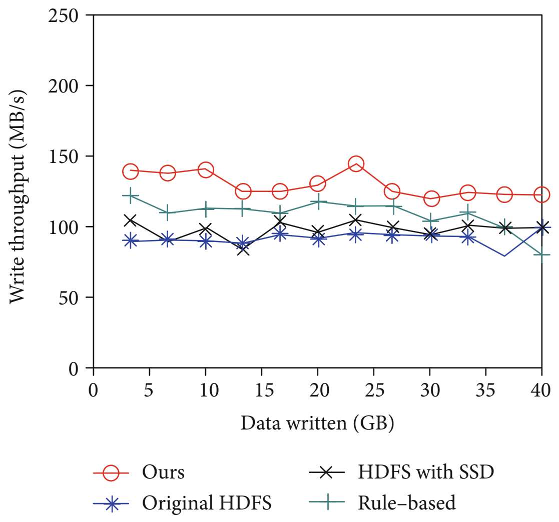

Figure 5: Average throughput for writing data.

As one of the many systems, ETH [53] is very popular with distributed file storage for nowadays, especially on its excellent advantage of smart contracts in blockchain. This platform can both run on fysieke computer and virtual machine, meanwhile, and it can be programmed with general procedure language. Therefore, it is an exciting platform for users in distributed file storage.

LTC [54], which is illustrated in Table 2, is a public chain-based technology for the distributed file storage system. It has very distinct features, such as fast speed for all transactions and marvelous efficiency for file storage. As its all transactions are executed in intensive memory, it needs very fewer nodes to participate computations, even though its transactions are encrypted and signed either by symmetric or asymmetric manners.

From Table 2, we can conclude that, in most systems, there are fewer smart contracts, which might cause risks for blockchain application. In this system, when a user deploys blockchain, there is a trade-off on cryptocurrency and blockchain. Moreover, this system is capable of supporting all applications based on blockchain. Users can assemble their own infrastructure, just like some popular cloud platforms, such as Amazon and Google.

6. Experiment and Evaluation

6.1. Experimental Results. The evaluation is executed on a Windows 10 machine equipped with an Intel(R) Core(TM) i7-7700M CPU @ 3.60 GHz, 16 GB RAM. The transaction nodes have been deployed in a virtual machine which is supported with Ubuntu 16.04.

Processing time overhead refers to the time consumed by the transaction nodes to verify data blocks. The experimental results are shown in Figure 4. At first, the processing time is about 5.5 ms. As more data blocks are generated, the processing time overhead increases; especially, there is a sharp increase of time overhead when the number of blocks changes from 20 to 30. After that, the time overhead increases smoothly. When the number of blocks comes to 60, the processing time overhead approaches about 71 ms.

6.2. Evaluation. In order to support our proposed scheme, we carried out evaluations and utilized the mostly adopted benchmarks, namely DFSIO [4], which is mainly focused on measuring network throughput for users' general operations, such as read and write. Additionally, this benchmark is based on a distributed approach.

The principal evaluation methodology is as the following: we focus on the data storing policy as well as the optimization goals, especially for data writing and reading throughput. Moreover, we make a comparison among the proposed schemes, the HDFS [4] and the rule-based strategy.

As depicted in Figure 5, the comparison result, we carry out about 20 times of evaluation and obtain the average throughput of every node for writing. The blockchain-based method gets the highest throughput, nearly about 138 MB/s, which is mainly due to the full use of its advantages, such as the optimal design of storage tier. However, the curve smoothly descends when the storage space (mainly memory) is mostly consumed, and this is a universal phenomenon for distributed file storage systems. The HDFS performs the worst, since its throughput only approaches average about 88 MB/s; this is due to the abandon of storage metrics.

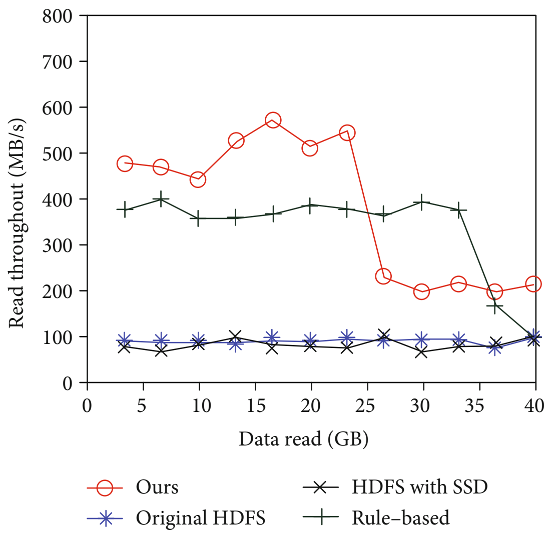

FIGURE 6: Average throughput for reading data.

Figure 6 shows the results of the READ throughput, where the HDFS strategy changes smoothly around 99 MB/s, while the HDFS with SSD policy exhibits a similar trend. While our proposed scheme harvests the best performance, the observations are twofold. As the former one, the proposed scheme equally distributes all requests onto all nodes, while the proposed scheme utilizes more HDDs storage media as the latter one. Accordingly, it can write more data blocks than the other policies. Neither the Original HDFS nor HDFS gets worse performance for reading data, respectively. Especially, HDFS yields the worst read performance.

7. Conclusion and Future Work

The distributed file storage system is susceptible to malicious use and fraud attack; users sometimes cannot have full control over their data. In this paper, we innovatively explores blockchain in distributed file storage, users no longer require a third-party, and own heavy supervision of their data. Through analysis and evaluations, our proposed scheme significantly improves data integrity and credibility for distributed file storage. Besides, based on blockchain, decisions on distributed file storage shall be more easier and reasonable. Finally, we carried out detailed discussion on the latest relative systems and demonstrated the advantages of this proposed work in distributed file storage.

As future directions, considering the network latency of the blockchain-based system, we will investigate the time tolerance of blockchain-based distributed file system and focus on the combination of network coding and blockchain to explore optimal network performance.

Data Availability

The test data, simulation data, and the proposed method used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This work is supported by the Science and Technology Projects of Hunan Province of China under Grant No. 2017SK1040, the Science Research Project of Education Department of Hunan Province under Grant No. 19B180, the Natural Science Foundation of Hunan Province under Grant No. 327 2018JJ2107, the Scientific research project of Guangzhou College of Technology and Business under Grant No. KA202031, the National Natural Science Foundation of China under Grant No. 61702180, and the Natural Science Foundation of Hunan Province under Granted No. 2019JJ50167.

References

[1] C. Xie, Y. Sun, and H. Luo, “Secured data storage scheme based on block chain for agricultural products tracking,” in *2017 3rd International Conference on Big Data Computing and Communications (BIGCOM)*, pp. 45–50, Chengdu, China, 2017.

[2] T. T. Nguyen, T. K. Vu, and M. H. Nguyen, “BFC: high-performance distributed big-file cloud storage based on key-value store,” in *2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pp. 1–6, Takamatsu, Japan, 2015.

[3] W. Liang, K.-C. Li, J. Long, X. Kui, and A. Y. Zomaya, “An industrial network intrusion detection algorithm based on multifeature data clustering optimization model,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2063–2071, 2020.

[4] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, “The hadoop distributed file system,” in *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pp. 1–10, Incline Village, NV, USA, 2010.

[5] A. Kamra, J. Feldman, V. Misra, and D. Rubenstein, *Growth Codes: Maximizing Sensor Network Data Persistence*, ACM SIGCOMM, 2017.

[6] Y. Tang, F. Li, and Y. Wu, “Research on the network coding for distributed storage file system based on the wavelet support vector machine,” *Journal of Computational and Theoretical Nanoscience*, vol. 13, no. 12, pp. 9628–9632, 2016.

[7] W. Liang, M. Tang, J. Long, X. Peng, J. Xu, and K.-C. Li, “A secure fabric blockchain-based data transmission technique for industrial internet-of-things,” *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3582–3592, 2019.

[8] J. Li, C. Pan, and M. Lu, “A seismic data processing system based on fast distributed file system,” *International Journal of Computers & Technology*, vol. 14, no. 5, pp. 5779–5788, 2015.

[9] B. Karthikeyan, A. Delignat-Lavaud, C. Fournet et al., “Formal verification of smart contracts,” in *Proceedings of the ACM-Workshop on Programming Languages and Analysis for Security*, pp. 91–96, Vienna, Austria, 2016.

[10] W. Liang, W. Huang, J. Long, K. Zhang, K.-C. Li, and D. Zhang, “Deep reinforcement learning for resource protection and real-time detection in IoT environment,” *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6392–6401, 2020.

[11] D. Liao, G. Sun, G. Yang, and V. Chang, "Energy-efficient virtual content distribution network provisioning in cloud-based data centers," *Future Generation Computer Systems*, vol. 83, pp. 347–357, 2018.

[12] Coinmarketcap, "Crypto-currency market capitalizations," 2016, https://coinmarketcap.com/.

[13] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Meard, "Resilient network coding in the presence of byzantine adversaries," in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, pp. 616–624, Barcelona, Spain, 2007.

[14] T. Ho, B. Leong, R. Koetter, M. Meard, M. Effros, and D. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings*, p. 144, Chicago, IL, USA, 2014.

[15] M. Giesler and M. Pohlmann, *The Anthropology of File Sharing: Consuming Napster as a Gift*, ACR North American Advances, 2003.

[16] N. S. Good and A. Krekelberg, "Usability and privacy: a study of kazaa P2P file-sharing," in *Proceedings of the conference on Human factors in computing systems - CHI '03*, pp. 137–144, Ft. Lauderdale, FL, USA, 2003.

[17] M. Ripeanu, "Peer-to-peer architecture case study: Gnutella network," in *Proceedings First International Conference on Peer-to-Peer Computing*, pp. 99-100, Linkoping, Sweden, 2001.

[18] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "The bit torrent p2p file sharing system: measurements and analysis," in *Peer-to-Peer Systems IV*, pp. 205–216, Springer, 2005.

[19] S. Wang and L. Cao, "Inferring implicit rules by learning explicit and hidden item dependency," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 3, pp. 935–946, 2020.

[20] K. Zhang, Y. Zhu, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Deep learning empowered task offloading for Mobile edge computing in urban informatics," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7635–7647, 2019.

[21] K. Ikeda, *qBitcoin: A Peer-to-Peer Quantum Cash System*, Social Science Electronic Publishing, 2017.

[22] K. K. Mar, Z. Q. Hu, C. Y. Law, and M. Wang, "Secure cloud distributed file system," in *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, pp. 176–181, Barcelona, Spain, 2016.

[23] D. C. M. Segura, M. D. C. Oliveira, T. K. Okada et al., "Availability in the flexible and adaptable distributed file system," in *2015 14th International Symposium on Parallel and Distributed Computing*, pp. 148–155, Limassol, Cyprus, 2015.

[24] S. Wang, L. Hu, Y. Wang, Q. Z. Sheng, M. Orgun, and L. Cao, "Intention nets: psychology-inspired user choice behavior modeling for next-basket prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 6259–6266, 2020.

[25] J. Yin, J. Wang, J. Zhou, T. Lukasiewicz, and J. Zhang, "Opass: analysis and optimization of parallel data access on distributed file systems," in *2015 IEEE International Parallel and Distributed Processing Symposium*, pp. 623–632, Hyderabad, India, 2015.

[26] K. Zhang, S. Leng, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Artificial intelligence inspired transmission scheduling in cognitive vehicular communications and networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1987–1997, 2019.

[27] W. Liang, Y. Fan, K.-C. Li, D. Zhang, and J. Gaudiot, "Secure data storage and recovery in industrial blockchain network environments," *in IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6543–6552, 2020.

[28] A. Purtell, "Support tiered storage policies in HDFS," 2015, https://issues.apache.org/jira/browse/HDFS-4672.

[29] T.-W. N. Sze, "Support archival storage in HDFS," 2015, https://issues.apache.org/jira/browse/HDFS-6584.

[30] M. Mihailescu, G. Soundararajan, and C. Amza, "Mix apart: decoupled analytics for shared storage systems," in *Presented as part of the 11th {USENIX} Conference on File and Storage Technologies ({FAST} 13)*, pp. 133–146, San Jose, CA, USA, 2013.

[31] C. Gkantsidis, D. Vytiniotis, O. Hodson, D. Narayanan, F. Dinu, and A. I. Rowstron, "Rhea: Automatic Filtering for Unstructured Cloud Storage," in *Presented as part of the 10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, pp. 343–355, Lombard, IL, USA, 2013.

[32] W. Liang, S. Xie, J. Long, K. C. Li, D. Zhang, and K. Li, "A double PUF-based RFID identity authentication protocol in service-centric internet of things environments," *Information Sciences*, vol. 503, pp. 129–147, 2019.

[33] W. Liang, D. Zhang, X. Lei, M. Tang, K.-C. Li, and A. Zomaya, "Circuit copyright blockchain: blockchain-based homomorphic encryption for IP circuit protection," *IEEE Transactions on Emerging Topics in Computing*, p. 1, 2020.

[34] S. Omohundro, "Cryptocurrencies, smart contracts, and artificial intelligence," *AI Matters*, vol. 1, no. 2, 2014.

[35] I. Weber, X. Xu, R. Riveret, G. Governatori, A. Ponomarev, and J. Mendling, "Untrusted business process monitoring and execution using blockchain," in *Business Process Management. BPM 2016*, vol. 9850 of Lecture Notes in Computer Science, , pp. 329–347, Springer, 2016.

[36] J. Benet, "Ipfs-content addressed, versioned, p2p file system," 2014, https://arxiv.org/abs/1407.3561.

[37] S. Wilkinson, T. Boshevski, J. Brandoff, and V. Buterin, "Storj a Peer-to-Peer Cloud Storage Network," 2014, https://storj.io/storj2014.pdf.

[38] V. Trón, A. Fischer, and Nagy, *State channels on swap networks: claims and obligations on and off the blockchain*, Ethersphere Orange Papers, 2016.

[39] J. Benet and N. Greco, *Filecoin: A Decentralized Storage Network*, Protoc. Labs, 2018.

[40] G. Wood, "Ethereum: a secure decentralized generalized transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, 2014.

[41] S. Wilkinson, J. Lowry, and T. Boshevski, *Metadisk: A Blockchain-Based Decentralized File Storage Application*, Storj Labs Inc., Technical Report, 2014.

[42] K. Zhang, Y. Zhu, S. Maharjan, and Y. Zhang, "Edge intelligence and blockchain empowered 5G beyond for the industrial Internet of Things," *IEEE Network Magazine*, vol. 33, no. 5, pp. 12–19, 2019.

[43] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Consulted*, vol. 1, no. 2012, p. 28, 2008.

[44] G. Wood, "Ethereum: a secure decentralized transaction ledger," http://gavwood.com/paper.pdf.

[45] E. Ben-Sasson, A. Chiesa, C. Garman et al., "Zerocash: decentralized anonymous payments from bitcoin," in *2014 IEEE*

Symposium on Security and Privacy, pp. 459–474, San Jose, CA, USA, 2014.

[46] I. Bentov and R. Kumaresan, "How to use bitcoin to design fair protocols," in *Advances in Cryptology – CRYPTO 2014*, pp. 421–439, Springer, 2014.

[47] R. Kumaresan and I. Bentov, "How to use bitcoin to incentivize correct computations," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security - CCS '14*, pp. 30–41, Scottsdale, AZ, USA, 2014.

[48] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: the blockchain model of cryptography and privacy-preserving smart contracts," in *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 839–858, San Jose, CA, USA, 2016.

[49] R. Cleve, "Limits on the security of coin flips when half the processors are faulty," in *Proceedings of the eighteenth annual ACM symposium on Theory of computing - STOC '86*, pp. 364–369, Berkeley, CA, USA, 1986.

[50] G. Asharov, A. Beimel, N. Makriyannis, and E. Omri, "Complete characterization of fairness in secure two-party computation of boolean functions," in *Theory of Cryptography*, pp. 199–228, Springer, 2015.

[51] E. Androulaki, A. Barger, V. Bortnikov et al., "Hyperledger fabric: a distributed operating system for permissioned blockchains," 2018, https://arxiv.org/abs/1801.10228.

[52] M. Samaniego and R. Deters, "Internet of Smart Things-Iost: Using Blockchain and Clips to Make Things Autonomous," in *2017 IEEE International Conference on Cognitive Computing (ICCC)*, pp. 9–16, Honolulu, HI, USA, 2017.

[53] V. Buterin, "Ethereum white paper, 2013," April 2018, https://github.com/ethereum/wiki/wiki/White-Paper.

[54] "Litecoin," 2011, February 2018, https://litecoin.org/.