Advance Access publication on 8 May 2020
doi: 10.1093/comjnl/bxaa046

Virtual Block Group: A Scalable Blockchain Model with Partial Node Storage and Distributed Hash Table

Bin Yu[1,2], Xiaofeng Li[1,2] and He Zhao[1,*]

[1] *Hefei Institutes of Physical Science, Chinese Academy of Sciences, 350 Shushanhu Road Hefei 230031, Anhui, P. R. China*
[2] *University of Science and Technology of China, No.96, JinZhai Road Baohe District, Hefei, Anhui, 230026, P.R. China*
[*] *Corresponding author: zhaoh@hfcas.ac.cn*

The inability to scale is one of the most concerning problems looming in blockchain systems, where every node has to store all contents of the ledger database locally, leading to centralization and higher operation costs. In this paper, we propose a model named virtual block group (VBG), which aims at addressing the node storage scalability problem. Adopting the VBG model, each node only needs to store part of block data and saves the VBG storage index to distributed hash table by taking block data as a resource, thus improving the query efficiency of block data. With the incentive mechanism of block data storage, and the storage verification and audit mechanism of block data, the security and reliability of block data storage can be ensured. The analysis and calculation show that this model saves hard drive storage space of the node to a greater extent with a shorter time of requesting block data, in the premise of ensuring secure and reliable block data. Compared to other technologies such as sharding, our model does not change the consensus mechanism or the network topology and retains the reliability and security of the original blockchain system.

Keywords: blockchain; virtual block group; storage scalability; DHT protocol

Received 14 June 2019; Revised 30 January 2020; Accepted 28 March 2020
Handling editor: Kaitai Liang

1. INTRODUCTION

With the increasing popularity of cryptocurrency such as Bitcoin, as well as the accelerated promotion of DAPP applications, the blockchain technology receives more and more attention. Blockchain is a decentralized distributed ledger technology, which is characterized by being transparent, trustable, tamper-proof, traceable and highly reliable. It reduces the trust cost and brings a new reform to the existing traditional informatization and internet [1,2,3].

However, with the unceasing application of blockchain technology, the problem of blockchain storage scalability becomes increasingly obvious. Taking the Bitcoin blockchain and Ethereum (ETH) as examples, which are the most commonly used blockchain systems, up to 25 March 2019, the block count of the Bitcoin blockchain is 568 687, and the blockchain size is 245 GB. The blocks count of ETH is 7435 856, and the blockchain size is 201 GB, which is increasing at an accelerated speed [4]. It is expected that the blockchain size of ETH will reach 1 TB within 2 years, calculating based on historical data [5]. The team of Bitcoin Unlimited has been studying and testing the block size of 1 GB [6], and supporters of BCH have been studying the block size at the TB level using special resources [7]. As the blockchain technology requires that every full node stores all block data, which brings huge pressure to the storage system of nodes in blockchain networks and seriously restricts increasing computers access to blockchain systems. In the future, the complete data backup of blockchain will greatly exceed the capacity that a personal computer can bear.

Many innovative methods have been proposed to solve the problem of blockchain storage scalability. These methods can be divided into two types: off-chain scaling and on-chain scaling [8]. Off-chain solutions reduce block data stored on blockchains as much as possible. Slepak *et al*. have proposed that off-chain solutions may widen the definition for the consensus. It is not required to store each transaction by the node any longer, and the node only needs to trace some transactions requested by a user at special moments [9]. For improving the transaction speed, the lightning network proposed by Poon

et al., it is possible to create a secure network of participants which are able to transact at high volume and high speed by bidirectional payment channels [10]. Decker *et al*. have launched duplex micropayment channels, which enable near-infinite scalability for digital payments based on Bitcoin. The actual transfers are then handled at a higher level through a network of payment service providers [11]. Burchert *et al*. proposed a new layer that sits in between the blockchain and the payment channels, which addresses the scalability problem by enabling trust-less off-blockchain channel funding [12]. Hyperledger Fabric applies the multichannel solution to realize multichain data isolation and protect the privacy of user data [13]. Pegged sidechain enables bitcoins and other ledger assets to be transferred between multiple blockchains [14]. On-chain solutions are to change the design of blockchain storage and optimize the block data structure and storage management. Bano *et al*. have put forward an overview of key approaches for on-chain scalability of blockchains, including collective leadership, sharding and parallel blockchain extension [15]. Proposed by Clifford *et al*., the Xthin (extremely thin) block transmission technology only needs to transmit 1/24 of the original number of bytes [16]. The specification of ETH sharding mechanism proposal, presented by Buterin, is a solution that introduces to on-chain state partition and gains higher throughput [17].

Generally, off-chain solutions often require additional software and may increase the complexity of the main chain. Transactions off-chain would be harder for the public to verify, which could lead to lower transparency and compliance issues. In this work, we take the approach of on-chain scaling, but off-chain solutions could also be further incorporated.

Besides, there are also other projects related to blockchain and storage scalability. Benet proposed IPFS [18], which provides a high-throughput content addressing a block storage model using content addressing hyperlink. Sia [19] uses the smart contract to let the nodes in the network rent storage space from each other. Storj [20,21] divides data into several encryption fragments according to the standard size and stores the fragments in different nodes to ensure data security. However, these studies focus on the file storage, rather than solving the storage problems of blockchain itself.

Through the on-chain scaling design, this paper proposes a scalable blockchain storage model: virtual block group (VBG). The model mainly comprises the following parts.

(i) The blocks with successive heights are combined as a VBG, and block data in VBG is stored by part of nodes.
(ii) With the incentive mechanism of block data storage, and the storage verification and audit mechanism of block data, the security and reliability of block data storage can be ensured.
(iii) Saving the VBG storage index to DHT (distributed hash table) by taking block data as a resource, thus improving the query efficiency of block data.

The structure arrangement of this paper is as follows: the second part introduces the modeling ideas and the overall framework of the model. The third part introduces definitions in this model. In the fourth part, a blockchain system applying the model is constructed. The fifth part implements a prototype of the VBG model and analyzes the scalability, efficiency, reliability and security of the model. The sixth part summarizes this paper and outlines future work.

2. MODELING

2.1. Block as a resource

In a blockchain system, DHT [22,23,24,25,26] is often applied to the distributed query service of resources. Each resource storage index is expressed as a <key, value> pair. The key is the unique identification of a resource and is often the hash value of the resource name or other descriptive information of the resource. The value is the IP address of the node which actually stores the resource or other descriptive information of the node. All <key, value> pairs make up a large hash table of the resource storage indexes. The hash table of the resource storage indexes is divided into multiple small pieces to be distributed to all nodes in the P2P network. Each node is responsible for maintaining one part of the hash table of the resource storage indexes. When a resource is requested through a node, the sole input required is the key of the target resource. As the hash table of the resource storage indexes contains the <key, value> pairs to be requested, the blockchain system routes the requested contents to the corresponding node. Then it can acquire all node address lists that store the resource from the hash table of resource storage indexes.

In the VBG model, successive blocks form a virtual block group, called VBG for short, which also can be regarded as a resource and only needs to be stored by part of nodes. The hash value of the VBG and the ID list of nodes storing the VBG form a <key, value> pair. All <key, value> pairs form a hash table of VBG storage indexes for the whole blockchain system. This hash table is divided into multiple small pieces to be distributed to all nodes in the P2P network. When VBG, block or transaction data is requested ed through a node, the target requested contents are routed to the node containing the <key, value> pairs for the VBG to be requested. The storage nodes address list is acquired from the <key, value> pairs of the VBG, and then the target contents to be requested can be acquired from the closer nodes in the list.

2.2. Scalable storage

The scalable blockchain storage model includes VBG metadata, VBG Merkle tree (described in Section 3.4), block data and DHT-VBG, which is the hash table of VBG storage index. The model is shown in Fig. 1.

The core idea of this model is as follows. The successive *n* blocks are considered as a VBG, and these blocks stored in

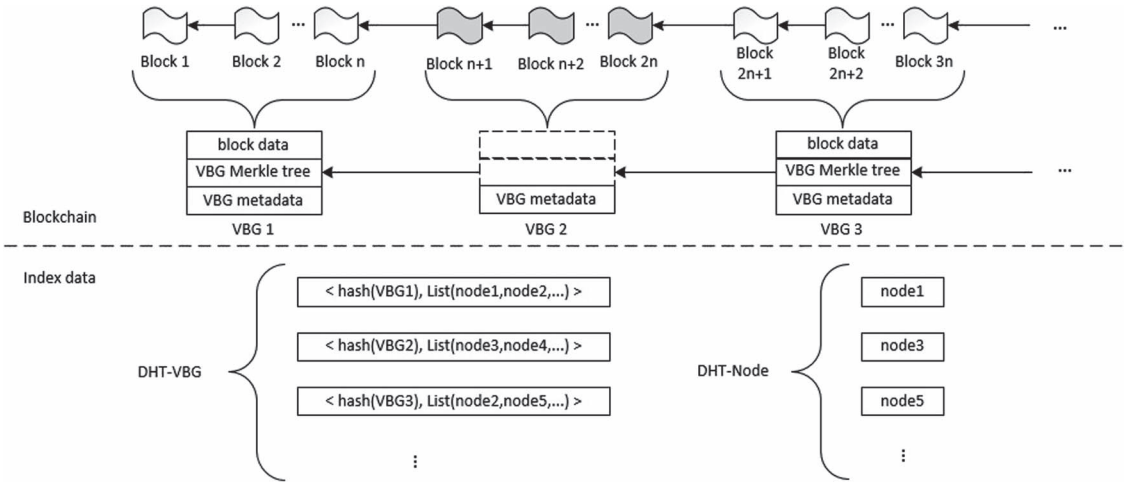

FIGURE 1. The scalable model of blockchain storage.

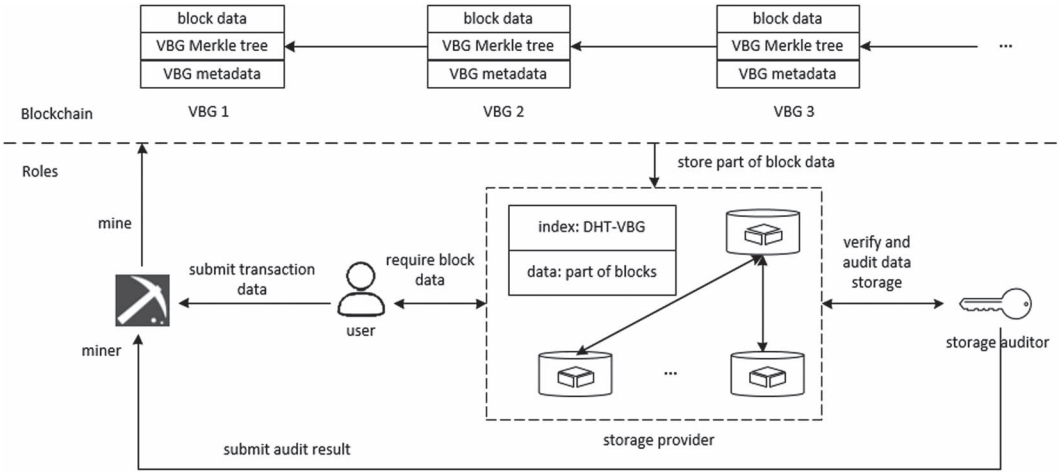

FIGURE 2. The architecture design.

VBG are taken as a resource. Shared access is available for data stored in VBG among various nodes in the P2P network. Such nodes do not need to store all blocks. For example, VBG 2 in Fig. 1 does not store block data but merely stores metadata information of the VBG.

DHT-Node is a DHT indicating neighbor nodes of the P2P network in blockchain systems, and the quantity of neighbor nodes is expressed as $N_{\text{neighborNode}}$. Referring to the design of DHT-Node, this model introduces a DHT-VBG hash table for maintaining the <key, value> pairs of all VBG storage indexes.

2.3. Architecture design

This model combines multiple successive blocks as a VBG, and each node does not need to store all block data. The architecture design is shown in Fig. 2. A user submits transaction data, then a miner receives data and packages transactions to form a block. When the block height reaches a value previously configured in blockchain systems, a miner combines blocks with successive heights as a VBG. A storage provider offers hard drive space to store part of block data. Each storage auditor verifies and audits VBG block data storage and submits audit result to miners.

There are the following main difficulties and challenges in the process of the architecture design: the definition of VBG, the verification and audit of VBG storage, the query efficiency of block data, which are described in detail in Sections 3.1, 4.3 and 3.2 separately.

3. DEFINITION

3.1. Definition 1 (VBG)

All blocks are divided into multiple VBG on the basis of block height. Some VBG store complete block data, which is

TABLE 1. The data structure of VBG metadata.

Data item	Size	Remark
VBG ID	32	The unique identification of VBG, the hash value of VBG metadata
Version	4	The version of VBG
VBG number	4	Similar to the block height, increasing from 1
Block start height	4	The height of the VBG starting block
Block end height	4	The height of the VBG ending block
VBG size	4	The size of all block data included in VBG
Time	4	The UNIX timestamp, accurate to second
VBG Merkle root	32	The hash value of Merkle root for all blocks in VBG

called full VBG, while other VBGs only store the metadata information, which is called simple VBG. When block data, not stored locally, need to be requested through a node, the block data can be acquired from neighbor nodes conveniently and quickly.

Referring to the design of the block header data structure of the Bitcoin blockchain [27], the data structure of VBG metadata is shown in Table 1.

To guarantee the stability and reliability of block data storage in the P2P network, the copy quantity N_{copyVBG} of VBG storage backup, should be kept in balance, and $N_{\text{copyVBG}} \geq 3$. As a new block generated from the blockchain system has instability, all nodes in the network should store these new blocks, and the max quantity of these blocks is $2n-1$, wherein n is the number of blocks in each VBG. When the quantity of these new blocks is $2n$, the earliest n blocks will form a new VBG. The size of VBG S_{VBG} is the minimum size of block data stored by each node. The theoretical minimum value is the size of one block, and the theoretical maximum value is the size sum of all blocks S_{BC}, which is configured in blockchain systems.

3.2. Definition 2 (DHT-VBG)

The VBG storage index is expressed as the form of a <key, value> pair. The key is the unique identification of VBG, namely VBG ID, and its bit length and functions are the same as those of node ID [28]. The value is an ID list of nodes storing this VBG. All <key, value> pairs form a hash table of VBG storage indexes for the whole blockchain system. The ID list of nodes storing the VBG can be located from the hash table of the VBG storage index.

The hash table of VBG index for the whole blockchain system is very large, and it is unfavorable for the maintenance by each node. The hash table of VBG index is divided into multiple small pieces to be distributed to all nodes in the P2P network, namely DHT-VBG.

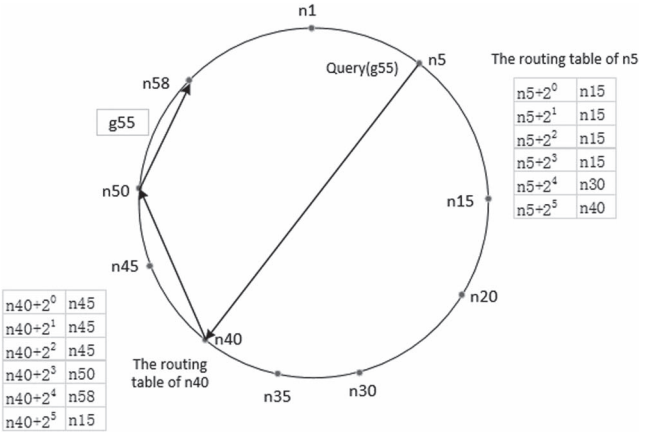

FIGURE 3. The Chord algorithm.

For the P2P network protocol, the Chord algorithm [29] is adopted for the request of resources. The node ID and the key are distributed in the Chord Ring clockwise, from small to large, to a ring with a size of 2^m for resource storage distribution and location, wherein m is the bit length of node ID and key. According to the algorithm, a resource is distributed to a node with a node ID, which is bigger than the resource key, namely the first node on the ring, starting from the key in a clockwise direction, which is called successor node. Each node maintains a routing table (called finger table) with maximum m items for rapid resource location.

When a resource is searched for through a node, firstly whether the successor node holds this resource is checked. If not, a direct search is performed in the routing table of the node from far to near, and the node nearest to the node holding this resource is located. Iterations are performed like this. According to the Chord algorithm, the query time complexity for VBG storage index is $O\left(\log 2^m\right)$. The Chord algorithm is shown in Fig. 3.

A ring with $m = 6$ is constructed in Fig. 3. Assuming that there are 10 nodes and some VBG resources, and VBG g55 needs to be requested through the node n5. The specific steps are as follows:

(i) The successor node n15 is searched for by the node n5. As g55 does not belong to (5, 15], it indicates that the node n15 does not hold the resource g55, and then a search is performed in the routing table of the node n5 from far to near.

(ii) The farthest item in the node n5 routing table is ($n5 + 2^5$, n40) and satisfies $40 \notin (5, 55]$. It indicates that the node n40 is nearer to the node holding g55. Skipping is performed to the node n40 to continue the searching. The successive node of the node n40 is the node n45, and g55

TABLE 2. The instance of VBG storage.

Node	VBG 1	VBG 2	VBG 3
Node 1	Storing block	Storing block	Storing block
Node 2	Not storing block	Storing block	Storing block
Node 3	Not storing block	Not storing block	Storing block
Node n1	Storing block	Not storing block	Not storing block
Node n2	Storing block	Storing block	Not storing block

does not belong to (40, 45]. It indicates that the node n45 does not hold the resource g55. Search is performed on the routing table of the node n40 from far to near.

(iii) Search is performed on the routing table of the node n40 from far to near. It is discovered that (n40 + 2^3, n50) satisfies 50∈ (40, 55]. It indicates that the node n50 is nearer to the node holding g55. Skipping is performed to the node n50 to continue the searching.

(iv) As the successive node of the node n50 is the node n58, and g55 belongs to (50, 58], it indicates that the node n58 holds the resource g55. The query is completed.

3.3. Definition 3 (minimum number of full VBG copies)

Keeping the number of block copies as close as possible is necessary. The new node and the node with free hard drive space should store VBG with the minimum number of full VBG copies in the P2P network, and then the next new node stores another VBG with the minimum number of full VBG copies in the P2P network. Iterations are performed continuously to maintain the stability and reliability of block data storage in the P2P network. Assuming there are three nodes storing three VBGs, and another two new nodes join the P2P network, the instance of VBG storage is shown in Table 2.

VBG 1 is the minimum number of full VBG copies in DHT-VBG before node n1 and node n2 join the P2P network. When new node n1 joins the P2P network, it stores all blocks of VBG 1. Therefore, node n1 is added to the <key, value> pairs of VBG 1 storage index, and VBG with the minimum number of full VBG copies in DHT-VBG is no longer VBG 1 but VBG 1 and VBG 2. When new node n2 joins the P2P network, it stores all blocks of VBG 1 and VBG 2. More iterations may be performed successively, so that the number of block copies in the P2P network tends to be the same or close.

3.4. Definition 4 (VBG Merkle tree)

Taking the hash value of each block in VBG as a leaf node, the hash value is calculated through leaf nodes in pairs, and the calculated hash value is taken as a new parent node. This process is repeated until there is only one node finally, namely VBG Merkle root. Referring to the design of the transaction Merkle tree [30, 31] in blockchain systems, the VBG Merkle tree is shown in Fig. 4.

The nodes that do not store blocks but only VBG metadata information do not store block data and complete Merkle tree in VBG. The verification of this VBG is similar to that of SPV [32]. In VBG composed of n blocks, calculating $\log_2 n$ times hash value is only needed for confirming any block.

For example, in Fig. 4, to verify Block 6, a node without storing block data requests a hash set from neighbor nodes. Using this hash set, including nodes of Hash 6, Hash 5, Hash 56, Hash 78, Hash 5678, Hash 1234 and Hash 1–8, we can trace to VBG Merkle root from the hash value of Block 6 along the Merkle tree. The set of these nodes is called a certification path, which is to confirm the existence and correctness of blocks.

4. CONSTRUCTION

4.1. Joining the P2P network

When a new node joins the P2P network, firstly a message is sent to initial nodes or nodes connected last time. As the node receives a message from other nodes in the P2P network, the list of neighbor nodes is updated, in which neighbor nodes ID saved. The new node is continuously joined to the DHT-Node.

The model should keep the number of full VBG copies tend to be the same or close. According to the Definition 3 in Section 3.3, when a new node joins the P2P network, the VBG with the minimum number of full VBG copies will be updated continuously, and it ensures the stability and reliability of block data storage in the network. Detailed steps are shown below:

(i) A new node joins the DHT-Node. When a node joins the P2P network of the blockchain system, a message is sent. As another node receives this message, the distance is calculated between the node receiving the message and the node sending the message using XOR [33]. If the distance is smaller than a neighbor node of the node receiving the message, this new node will be added to DHT-Node of the node receiving the message.

(ii) The VBG storage index of the new node is acquired. Every node maintains a VBG storage index table. A node sends a get_VBGIndex message to the new node to acquire the VBG storage index of the new node. When this node receives the VBG storage index of the new node, the node verifies VBG storage block data. As for the node failing in the verification, it is not joined into DHT-VBG.

(iii) DHT-VBG in the P2P network is updated. As for the node passing the verification of VBG storage block data, Iterations are performed on VBG storing block data to all storage nodes successively according to the VBG storage index returned. The new node ID is added to the <key, value> pairs of the corresponding VBG. The DHT-VBG update for the P2P network is completed.

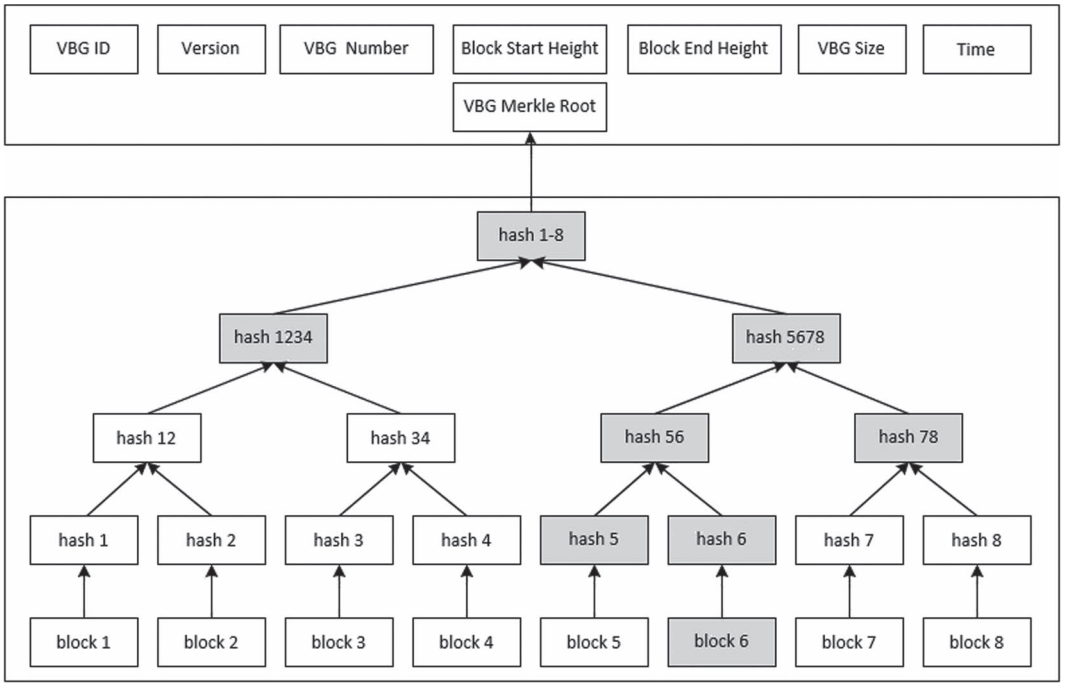

FIGURE 4. The VBG Merkle tree.

4.2. Storage and query

The size of VBG block data stored by a node can be set on the basis of the hard drive space for block data storage that can be offered by the node, meanwhile the minimum number of full VBG stored by each node must be configured. It can prevent the extreme condition that all nodes in the P2P network do not store block data.

After a new node joins the P2P network, the VBG storage of the node needs to be updated. If the node does not store any block data, or the size of block data stored by the node does not reach the minimum value of the percentage for the size of block data configured in the blockchain system, the VBG with the minimum number of full VBG copies is downloaded from neighbor nodes. After downloaded block data passes verification, it is saved to the node and the DHT-VBG of the P2P network is updated. If the size of block data stored by the node exceeds the minimum value of the percentage for the size of block data configured in the blockchain system, the DHT-VBG of the P2P network is directly updated. The block data stored by nodes will change under the following three cases:

(i) When a node joins the blockchain system for the first time, the VBG with the minimum number of full VBG copies in DHT-VBG will be downloaded, until the size of block data storage configured in the blockchain system is reached.
(ii) The blockchain system performs audit regularly, described in Section 4.3. At the time of the periodic audit, if the number of full VBG copies in DHT-VBG is smaller than the threshold value configured in the blockchain system, the VBG block data is backed up to a storage node. According to the Proof of storage in Section 4.4, because the rewards can be obtained if a node provides hard drive space to store VBG block data, more and more nodes will provide hard drive space and keep online as long as possible.
(iii) When a new VBG is produced, a node stores block data of this VBG according to the size of the data storage hard drive space set by the node.

In this model, as each node only stores part of block data, when a node requests block data not stored locally, it is needed to query DHT-VBG at first and acquire a node list storing the requested block data, then acquire the requested block data from closer nodes. Detailed steps are described below:

(i) According to the block information to be requested, the local VBG storage index is queried. If the block data to be requested is stored locally, the block data is directly acquired from the node locally.
(ii) If the block data to be requested is not stored locally, the Chord algorithm described in Definition 2 in Section 3.2 is adopted. The list of VBG storage nodes is acquired from the DHT-VBG storage index.
(iii) The nodes closer to this node are found out from the nodes list storing VBG. The block data is downloaded from these closer nodes.

4.3. Verification and audit

When a node requests block data not stored locally, it is needed to download from other neighbor nodes and to verify the correctness of acquired block data. Firstly, whether the hash value of the block header data information is consistent with the hash value of the block is verified. Then according to the VBG Merkle tree described in Section 3.4 Definition 4, a hash value set of the block certification path is acquired, and verification is performed successively till VBG Merkle root successfully passes the verification.

If a node asserts that it has stored part of block data, the blockchain system will perform audit regularly through the hash challenge to ensure that block data stored by this node is neither lost nor deliberately deleted. Each storage node audits the previous VBG block data storage of the full VBG (the previous VBG of full VBG 1 is the newest full VBG k). For example, a node stores full VBG 1, full VBG 5 and full VBG k, then this node should verify full VBG k, full VBG 4 and full VBG k-1. The audit node takes the VBG number to be verified as the parameter to transmit to the audited node (to reduce the download time and the calculation, it can verify n pieces of block data in full VBG at random). The audited node returns the data locally stored to the audit node as per the specific parameter, and the audit node uses the VBG Merkle tree for verification. As each storage node participates in the storage audit of other storage nodes, there will be multiple nodes for each full VBG to perform audit, and 'a result of more than 50%' is taken as a final audit result of full VBG storage to be written into an audit result pool, thus effectively preventing offline or cheating of the audit node.

At the time of audit, if it is discovered that a node does not store VBG block data or a storage node is unachievable, such node is deleted from the <key, value> pair of the storage index, meanwhile whether the number of storage nodes in the storage index <key, value> pair is smaller than the threshold value configured in the blockchain system is checked. If yes, the data backup process is initiated and the VBG block data is backed up to VBG block data storage nodes. Therefore, the security and reliability of block data storage can be guaranteed after each audit in the P2P network. A detailed control algorithm of full VBG storage audit is shown in Fig. 5.

4.4. Proof of storage

The blockchain system packages some transactions to form a block and stores it to the blockchain through a consensus mechanism. When the height of blockchain reaches a value configured in the blockchain system, a mining node combines blocks with successive heights as a VBG. The VBG does change the original blocks which have already reached a consensus, but it only aggregates a certain amount of block information with successive heights and forms a VBG, including VBG metadata and VBG Merkle tree. After block data is written on the blockchain, the VBG data is determined. The node can optionally continue to store block data within this VBG or delete these block data and then make use of the consensus mechanism, which is the same as that at the time of transactions packaging for forming a block, to store VBG data on the blockchain.

In this model, a node storing block data offers hard drive storage space, guarantees the security of block data in the P2P network, and thus nodes storing block data should be rewarded [18,19,20,21]. One node stores block data, and other nodes audit the authenticity and online status for storage block data. Within one storage cycle (for example, in terms of month), if the percentage of the passing audit number and the total audit number reaches a standard value configured in the blockchain system (such as 90%), this storage node will acquire certain costs, which are taken as rewards to nodes storing block data.

Each storage node needs to be subjected to the regular audit of other nodes after storing VBG block data. As for audit nodes, the audit operation is regular; however, to audited nodes, as the time for joining other neighbor nodes to the network is inconsistent, subjection to audit is not regular. As long as a VBG storage node always keeps online or is only occasionally offline, the storage rewards can be acquired. To earn more rewards, nodes storing block data will be online as long as possible to effectively guarantee the security of block data storage.

Within one storage cycle, the node storing VBG block data should be subjected to the audit of other nodes; meanwhile, the storage audit to other nodes is required to be completed to acquire the storage rewards. The VBG block data storage consensus steps are as follows:

(i) Audit results data is extracted. Mining nodes group data in the audit result pool as per different VBG block data storage nodes and analyze audit results each time on the basis of the principle of 'the minority is subject to the majority'.
(ii) The audit results are counted. If a node storing VBG block data reaches one storage cycle, the audit results are counted, and whether the percentage of the passing audits number to the total audits number reaches a standard value configured in the blockchain system is checked.
(iii) The audit results are packaged. If the audit results reach a standard value configured in the blockchain system, a mining node will package the audit results of this node storing VBG block data and write it to the blockchain.
(iv) Rewards. The mining node acquires the mining rewards; meanwhile, the node storing VBG block data acquires the block data storage rewards of a storage cycle.

This model includes three types of data: transaction data, VBG data and audit results. The procedures for each type of data stored on the blockchain are shown in Fig. 6.

The mining node, the storage node, and the audit node are classified as per their logical functions. Each node in the

```
Algorithm: the full VBG storage audit
Input:    groupNo:    The full VBG number needed to be audited
Output:   result:     The result of this audit
// Get storage index <key, value> pair according to the number of full VBG
1:  pairGroup ← GetDHT-VBG (groupNo)
2:  hashGroup ← pairGroup.Key
3:  lstNode ← pairGroup.Value
4:  for nodeItem in lstNode do
5:      blockNos ← getRandomBlockNos (START_HEIGHT, END_HEIGHT)
// Get the special block data of VBG
6:      dataGroupTmp ← GetGroupData (nodeItem, hashGroup, blockNos)
// Verify the block data of VBG
7:      isValid ← Verify (dataGroupTmp, hashGroup, blockNos)
8:      SubmitVerify (isValid, hashGroup)
// Get the VBG verify result verified by other nodes
9:      lstVerify ← GetLstVerify (hashGroup)
// When the number of the verify result is bigger than the number of the minimum verify
node configured in blockchain systems
10:     if lstVerify.Count >= MIN_VERIFY_NODE then
11:         countVerifySuccess ← GetTrueCount (lstVerify)
12:         if countVerifySuccess * 2 < lstVerify.Count then
13:             RemoveDHT-VBGNode (groupNo, nodeItem)
14:             lstNode.Remove (nodeItem)
15:         end if
16:     end if
17:  end for
18:  if lstNode.Count < CountVBGCopy then
// Get DHT-node of the local node
19:      lstDHTNode ← GetDHT-NodeList()
// Order by distance
20:      lstNodeTmp ← lstDHTNode.Sort()
21:      i ← 0
22:      for nodeItem in lstNodeTmp do
// Save VBG block data to the neighbor nodes
23:          SaveGroup (hashGroup, nodeItem)
24:          i ← i + 1
25:          if i >= CountVBGCopy - lstNode.Count then
26:              break
27:          end if
28:      end for
29:  end if
30:  return true
```

FIGURE 5. The algorithm of the full VBG storage audit.

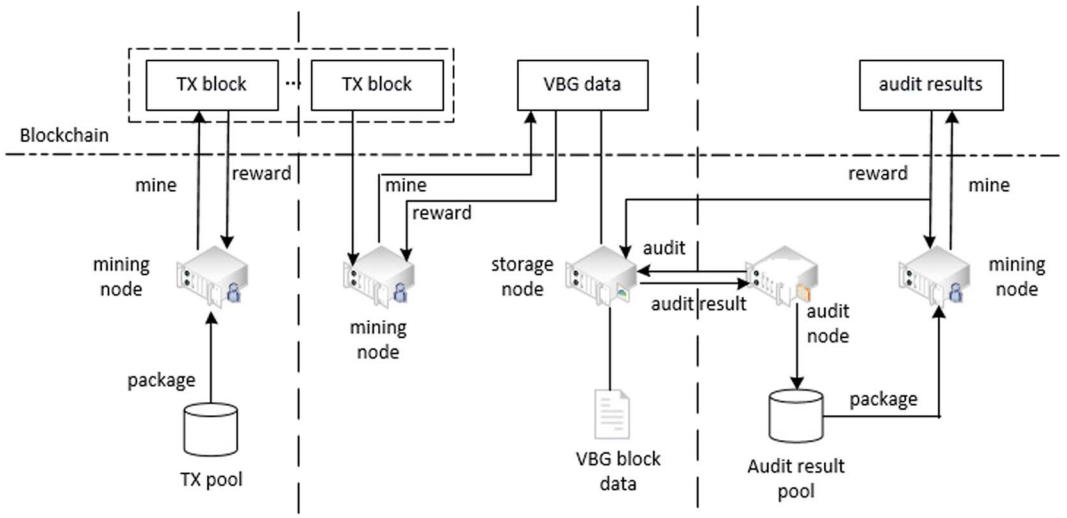

FIGURE 6. The data flow of the model.

P2P network is mutually equipotent and can be any logical function node at different moments. The consensus mechanism in this model is consistent with the consensus mechanism of the existing blockchain system.

The transaction data is packaged to form a block through a mining node, to speed up transaction validation, it is recommended to store all block data for mining nodes. The blockchain system generates many blocks, when the number of blocks is twice as large as n which configured in the blockchain system, a mining node packages earliest n blocks to form a VBG and writes it to the blockchain, then the VBG block data is stored by part of nodes. Audit nodes perform the regular audit for VBG block data storage nodes, a mining node packages audit results, and storage nodes acquire the storage rewards of VBG block data.

5. IMPLEMENTATION & ANALYSIS

We implement a prototype of the VBG model and analyze the scalability, efficiency, reliability and security, then we compare to other related storage scalability schemes including sidechain and sharding.

5.1. Implementation

We implemented a prototype of the VBG model in C#, consisting of approximately 17 200 lines of code. We choose POW [1] as the consensus protocol for packaging transactions into a block and packaging blocks into a VBG. Signatures and public/private key pairs are implemented with the System.Security.Cryptography library, and we use SHA-256 for hash function. The Kad algorithm [33] is adopted to manage the neighbor nodes of each node. The Kad is widely used in the decentralized structured P2P network and is also adopted by ETH network, BitTorrent, and the like. Table 3 shows the parameters in our prototype of the VBG model.

TABLE 3. The implementation parameters.

Parameters	Value	Remark
$S_{\text{maxOfBlock}}$	1 MB	The maximum size of a block
$N_{\text{blockInVBG}}$	1024	The number of blocks of each VBG
N_{copyVBG}	30	The number of VBG storage backup
$N_{\text{neighborNode}}$	64	The number of neighbor nodes
N_{minVBG}	1	The minimum number of VBG stored by each node

In our implementation, each node stores at least one VBG, namely 1 GB of hard drive storage space required.

5.2. Storage scalability

By changing the design of block data storage, each node only needs to store part of block data. The minimum size of block data stored by nodes is at least the size of one block group configured in the blockchain system, and the blockchain system can configure that each node must store the minimum number of full VBG.

The number of full VBG stored by a single node $N_{\text{storageVBG}}$ is

$$N_{\text{storageVBG}} = \frac{N_{\text{VBG}} \times N_{\text{copyVBG}}}{N_{\text{neighborNode}}} \quad (1)$$

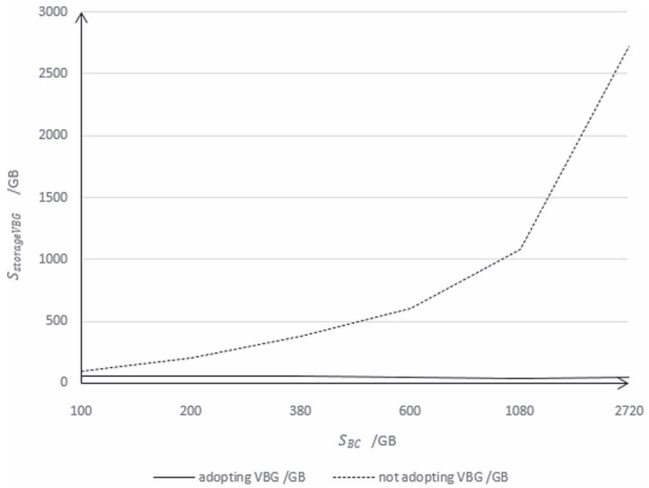

FIGURE 7. The comparison of the storage scalability.

The size of VBG block data stored by a single node $S_{\text{storageVBG}}$ is

$$S_{\text{storageVBG}} = \frac{S_{\text{BC}} \times N_{\text{copyVBG}}}{N_{\text{neighborNode}}} \tag{2}$$

wherein N_{VBG} is the number of VBG in the block system, which continuously increases with the application of the blockchain system. The bigger N_{copyVBG} is, the more reliable block storage will be. To save the size of hard drive space for nodes, it may decrease with the continuous increase of blocks, and its minimum value is 3. The bigger $N_{\text{neighborNode}}$ is, the smaller the number of full VBG distributed to storage nodes will be. Assuming that N_{copyVBG} is 30, the analysis of the storage scalability is shown in Fig. 7.

The model saves lots of hard drive space for nodes. When the size of block data increases, the storage scalability advantage of the VBG model is more significant.

According to the proof of storage in Section 4.4, this model does not change the original consensus mechanism of blockchain systems, so we can further incorporate off-chain solutions into the VBG model:

(i) Adopting a secure, high-speed network, such as the lightning network [10]. It will improve the transaction data volume and speed of blockchain systems.
(ii) Reducing the size of block data by not storing each transaction. When a user requires a transaction not store locally, data can be downloaded from other nodes.
(iii) Introducing a side chain [14] scheme. The detail transaction data store in a side chain, and the important and concise data store in the main chain.

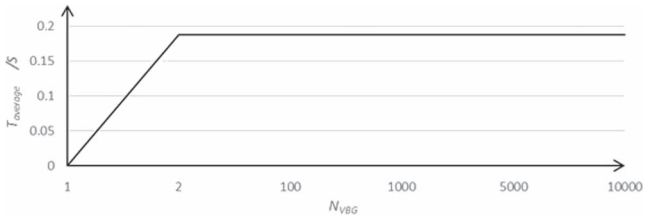

FIGURE 8. The average transmission time.

5.3. Query efficiency

Each node can directly request block data locally in blockchain systems, in which each node locally stores the whole block data. When requesting block data that is not stored locally adopting this model, the list of full VBG storage nodes is queried at first, the time is expressed as T_1; then, the block data is downloaded from full VBG storage nodes, the time is denoted as T_2; after downloading the block data, the block data is verified, the expression of the time is T_3. The total time$T$ for acquiring block data is

$$T = T_1 + T_2 + T_3 \tag{3}$$

According to Definition 2 in Section 3.2, the time complexity of T_1 is $O(\log 2^m)$, wherein 2^m is the maximum number of neighbor nodes, and the time T_1 is negligible. According to Definition 4 in Section 3.4, the time complexity of T_3 is $O(\log_2 n)$, wherein n is the number of blocks in full VBG, and the time T_3 does not affect the analysis result. Under the premise of a certain size of block data, T_2 is related to the network bandwidth between nodes. The bigger the bandwidth is, the smaller T_2 will be, and T_2 is

$$T_2 = \frac{D_{\text{dataLength}}}{20/8} \tag{4}$$

wherein $D_{\text{dataLength}}$ is the size of block data, assumed that the average transmission rate is 20 MPs, the average transmission time of the prototype of VBG model is shown in Fig. 8.

The rate of full VBG stored by a single node $R_{\text{storageVBG}}$ is

$$R_{\text{storageVBG}} = \frac{N_{\text{storageVBG}}}{N_{\text{VBG}}} = \frac{N_{\text{copyVBG}}}{N_{\text{neighborNode}}} \tag{5}$$

According to the prototype of VBG model, the transmission time is 0.4 s to transmit a block with the size of 1 MB, $R_{\text{storageVBG}}$ is 0.47, and the average transmission time is 0.19 s. A VBG node, which stores part of block data, can directly acquire that part of data locally, and it does not bring any bandwidth overhead. When the node requests a specific transaction externally but not an entire block, the node does not need to request the whole block data but only the given transaction

TABLE 4 The failure probability for accessing VBG.

$N_{\text{neighborNode}}$	N_{copyVBG}	$R_{\text{peerOffline}}$	$R_{\text{fileFailure}}$
30	3	0.5	3.079e-5
30	3	0.4	1.576e-5
60	3	0.3	7.890e-7
60	3	0.1	2.922e-8
30	6	0.5	2.631e-8
30	6	0.4	6.898e-9
60	6	0.3	1.456e-11
60	6	0.1	1.997e-14

data, and the transmission time is only 0.2 ms, which can be negligible.

Taking the ETH as an example, the size of the block with a height of 8 737 823 is 21 832 Bytes [34], and the total time T for acquiring the block data is 0.01 s. To transmit a block with the size of 1 MB or so in the bitcoin network, the transmission duration is 0.4 s, and the average transmission time is ∼0.2 s. From the analysis, to save hard drive storage space for nodes, the VBG model increases the time for acquiring block data, but the time is negligible.

5.4. Reliability

When all nodes storing the same full VBG in the P2P network are offline, access to blocks in this VBG will be unsuccessful. The probability of a node being offline is $R_{\text{peerOffline}}$, the size scope is (0, 1), and the failure rate for accessing full VBG is $R_{\text{fileFailure}}$. The expression for the theoretical value is

$$R_{\text{fileFailure}} = \frac{\left(R_{\text{peerOffline}}\right)^{N_{\text{copyVBG}}}}{\binom{N_{\text{neighborNode}}}{N_{\text{copyVBG}}}} \tag{6}$$

wherein $\binom{N_{\text{neighborNode}}}{N_{\text{copyVBG}}}$ refers to different methods for getting N_{copyVBG} nodes from $N_{\text{neighborNode}}$ nodes of DHT-Node; $\left(R_{\text{peerOffline}}\right)^{N_{\text{copyVBG}}}$ refers to the simultaneous offline probability for N_{copyVBG} storage block data nodes. The failure probability for accessing VBG is shown in Table 4.

The failure probability for accessing VBG is very small. The lower the offline probability for each node is, the lower the failure probability for accessing VBG will be. The bigger the number of full VBG is, the lower the failure probability for accessing blocks will be.

5.5. Security

Security is particularly important in blockchain systems. There are many types of attacks during the block data storage in blockchain systems. We list several possible attack vectors and their solutions adopting this model.

Fabricating VBG data. When a node requires block data not stored locally, neighbor nodes may send wrong or false block data. As described in Section 4.3, when receiving block data, the node verifies these block data. For the block data that fails in the validation, the node will delete the data-sending node from its neighbor nodes, and the node receives correct block data from other neighbor nodes.

DDoS attacks, which initiates a great number of requests to acquire a certain full VBG for network attack. On the one hand, the node identity ID is random, attack nodes will be dispersed into different network routes, thus an attacker cannot guarantee that attack nodes are within the same network route. On the other hand, according to Section 4.2 Storage and query, because of the incentive mechanism of block data storage, there will be many nodes (three at least) storing full VBG, and the request of acquiring a certain full VBG will be distributed to different block storage nodes. The defense mechanisms have been adopted to counteract the DDoS attacks. For example, the node does not respond to requests within the requested time interval, and malicious nodes once identified will be blacklisted locally like in ETH network. Some techniques are currently being introduced in the protocol layer. A small amount of PoW in the block data request greatly increases difficulty of DDoS attacks, which has been tested in a cryptocurrency named Nano (RaiBlocks) [35]. The alternative way is paying very low fees with micropayments channels such as lightning network [10] or Raiden network [36] when requesting block data, which can also largely increase the DDoS attacks costs, but exert negligible impact on normal nodes.

Sybil attacks, in which an attack node has only one copy or deletes all copies, and deceives other nodes, as if it possesses multiple copies of block data. Each full VBG storage is signed by the node's private key and is unique, so an attacker cannot pass the verification unless there is a unique storage backup. According to Section 4.3 Verification and audit, the identity ID of a node at the time of joining the network is sole and random and cannot be forged; the regular audit of each node guarantees that block data must be stored validly, and each block data copy is dispersively stored by a node with an independent ID.

Long range attack, which means an attacking node rewrite many blocks in the ledger at once. If a node controls more than 51% of computing power in POW [1], this node can tamper ledger, but it is very difficult to reach 51% of computing power. The blockchain system depends on the lack of constraints on computing power in POS [37]. In this model, when the number of blocks is twice as large as n which is configured in the blockchain system, a mining node packages earliest n blocks to form a new VBG. This method effectively reduces the possibility of a long-range attack.

Google attack is a coordinated attack on the network, initiated by a major company or entity controlling enormous computing power or storage space. According to Section 4.2 Storage and query, the blockchain system configures that each node must store the minimum value of full VBG to prevent

TABLE 5. VBG model compared to other schemes.

	Metrics	Sidechain	Sharding	VBG model
Reliability	$R_{\text{fileFailure}}$	No change	Reduction	No change
Security	Number of nodes	Reduction	Reduction	No change
Ease of Implementation	Degree of difficulty	High	High	Low
Complexity	Cost	High	High	Low

the extreme condition that all nodes in the P2P network do not store block data. As the identity ID of nodes when joining the network is random, the major company or entity cannot enable multiple nodes to store all copies of the same full VBG. Even if the major company or entity enable them to possess the offline or stop services for all nodes, the audit function can back up full VBG with the number of storage copies smaller than that of configured in the blockchain system to a new node.

Spoofing attack, collusion attack and slander attack between storage nodes and audit nodes. Spoofing attack: as an audit node requires a storage node to offer data of a random block for each audit, if this storage node does not store block data, this storage node cannot pass data verification of the audit node. Collusion attack: a final audit result is determined by multiple audit nodes jointly and thus a storage node and an audit node cannot collude. Slander attack: a storage node actually stores block data, but an audit node submits a false audit result, as the audit is completed by multiple nodes jointly, the final result is determined by multiple audit nodes jointly, and an audit node cannot slander storage nodes.

5.6. Scheme comparison

The scalability, reliability, security and complexity are fully considered at the time of design for this model. The comparisons with other scalability schemes are shown in Table 5.

$R_{\text{fileFailure}}$ is one of the most important indexes to the reliability of blockchain systems. If a greater number of nodes are needed, the scheme reduces blockchain security. A lower degree of difficulty shows the implementation of blockchain systems is easier. The development and operation costs indicate the complexity of blockchain system.

Sidechain adopts an independent blockchain to reduce the storage pressure of the main chain, but more nodes are needed to operate the side chain; otherwise, it is difficult to ensure enough security. The side chains need to mutually shift among independent and asynchronous blockchains; thus, the cost of implementation is high, and it increases the complexity of the blockchain system.

Sharding partitions the P2P network into more and smaller fragmentations and reduces the node number of each fragmentation, thus affecting the reliability and reducing security to some extent. In aspects of design and realization, to avoid attacks by entities that control a large number of fragments, it is necessary to ensure that nodes are kept in the fragments in a secure manner. The complexities of the network fragmentation, the transaction fragmentation and status fragmentation are both relatively high.

The scalable blockchain storage model adopting VBG does not change the consensus mechanism, the encryption algorithm, the block data structure, etc., of the original blockchain system; thus, it is easy to realize, and furthermore, it does not change the reliability and security of the original blockchain system.

6. CONCLUSION

In this paper, we put forward a model of combining multiple successive blocks as a VBG, and each node only needs to save part of block data adopting this model. With the incentive mechanism of block data storage, and the storage verification and audit mechanism of block data, the security and reliability of block data storage are ensured. Through analysis and calculation, in the premise of ensuring secure and reliable block data storage, this model saves the hard drive storage space of nodes to a greater extent with a shorter time of requesting block data and improves the blockchain storage scalability. This model is easy to realize, and furthermore, it does not change the reliability and security of the original blockchain system. The proposed method is not only applicable to the scalable storage of block data but can also be utilized to store non-block transaction data, which need to be uploaded to blockchain and stored distributively.

The future work involves cutting down the time for nodes to request block data, for example, to simultaneously download part of full VBG block data from multiple neighbor nodes in multiple threads and to splice them into complete block data. Besides, we plan to make use of smart contracts [37] to complete the audit of VBG block data storage nodes and simplify the audit work of block data storage.

Acknowledgements

This study was supported with the National Natural Science Foundation of China (No. 61602435) and Natural Science Foundation of Anhui Province (No. 1708085QF153).

REFERENCES

[1] Nakamoto, S. (2008) Bitcoin: a peer-to-peer electronic cash system. https://bitcoin.org/en/bitcoin-paper (6,5,2018).

[2] He, Z., Li, X., Zhan, L. and Zhongcheng, W. (2015) Data integrity protection method for microorganism sampling robots based on blockchain technology. *Huazhong Univ. of Sci. &. Tech. Natural Science Edition*, 43, 216–219.

[3] Bitcoin Developer Guide. https://bitcoin.org/en/developer-guide (15,5,2018).

[4] Bitcoin Cash – Peer-to-Peer Electronic Cash. (2017) https://bitinfocharts.com (25,3,2019).

[5] Ethereum (ETH) Blockchain Explorer. https://etherscan.io (25,3,2019).

[6] Bitcoin Unlimited. https://www.bitcoinunlimited.info (8,5,2019).

[7] Bitcoin Cash – Peer-to-Peer Electronic Cash. https://www.bitcoincash.org (8,5,2019).

[8] Zyskind, G., Nathan, Oz., Pentland, A. (2015) Enigma: Decentralized Computation Platform with Guaranteed Privacy. https://arxiv.org/abs/1506.03471 (20,6,2019).

[9] Slepak, G. and Petrova, A. (2018) The DCS theorem. https://arxiv.org/abs/1801.04335 (20,6,2019).

[10] Poon, J. and Dryja, T. (2016) The Bitcoin lightning network: scalable off-chain instant payments. https://lightning.network/lightningnetworkpaper.pdf (20,6,2019).

[11] Decker, C. (2016) *On the Scalability and Security of Bitcoin*. Createspace Independent Publishing Platform, South Carolina.

[12] Burchert, C., Decker, C. and Wattenhofer, R. (2017) Scalable funding of bitcoin micropayment channel networks. In *International Symposium on Stabilization, Safety, and Security of Distributed Systems, Boston, MA, USA, 5–8 November*, pp. 361–377. Springer, Cham.

[13] Home – Hyperledger. (2019) https://www.hyperledger.org (2,3,2018).

[14] Back, A., Corallo, M., Dashjr, L., Friedenbach M., Maxwell, G., Miller, A., Poelstra, A., Timón J., Wuille, P. (2014) Enabling blockchain innovations with pegged sidechains. https://blockstream.com/sidechains.pdf (22,3,2018).

[15] Bano, S., Al-Bassam, M. and Danezis, G. (2017) The road to scalable blockchain designs. *USENIX;login*, 42, 31–36.

[16] Clifford, A., Rizun, P. R., Suisani, A., Stone, A., Tschipper, P. (2016) Towards massive on chain scaling: presenting our block propagation results with xthin. https://medium.com/@peter_r/towards-massive-on-chain-scaling-block-propagation-results-with-xthin-da54e55dc0e4 (16,1,2018).

[17] Buterin, V. (2019) Sharding document. https://github.com/ethereum/sharding/blob/develop/docs/doc.md (10,8,2019).

[18] Benet, J. (2014) IPFS - content addressed, versioned, P2P file system (DRAFT 3). https://arxiv.org/abs/1407.3561 (5,8,2018).

[19] Vorick, D. and Champine, L. (2014) Sia: simple decentralized storage. https://sia.tech/sia.pdf (10,8,2018).

[20] Wilkinson, S. (2014) Storj - decentralized cloud storage. https://storj.io/whitepaper (17,8,2018).

[21] Wilkinson, S. and Lowry, J. (2014) Metadisk: blockchain-based decentralized file storage application. http://metadisk.org/metadisk.pdf (20,8,2019).

[22] Loewenstern, A. and Norberg, A. (2017) DHT protocol. http://www.bittorrent.org/beps/bep_0005.html (8,5,2018).

[23] Ratnasamy, S., Francis, P., Handley, M., Karp, R. and Shenker, S. (2001) A scalable content-addressable network. *ACM SIGCOMM Comp. Comm. Rev.*, 31, 161–172.

[24] Rowstron, A. and Druschel, P. (2001) Pastry: Scalable, Decentralized Object Location and Routing for Large-Scale Peer-to-Peer Systems. In *IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*, pp. 329–350. Springer, Berlin, Heidelberg 12-16 November.

[25] Karp, B., Ratnasamy, S., Rhea, S. and Shenker, S. (2004) Spurring Adoption of DHTs with OpenHash, a Public DHT Service. In *International Workshop on Peer-to-Peer Systems 2004*, pp. 195–205. Springer, Berlin, La Jolla 26-27 February.

[26] Wang, L. and Kangasharju, J. (2013) Measuring Large-Scale Distributed Systems: Case of Bittorrent Mainline DHT. In *Peer-to-Peer Computing (P2P), IEEE Thirteenth International Conference on Peer-to-peer Computing, Trento, Italy, 9–11 September*, pp. 1–10. IEEE, Piscataway.

[27] Blockmeta. (2019) https://blockmeta.com/btc-stat (22,5,2018).

[28] Harrison, D. (2016) Peer ID conventions. http://www.bittorrent.org/beps/bep_0020.html (19,8,2018).

[29] Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F. and Balakrishnan, H. (2003) Chord: a scalable peer-to-peer lookup service for Internet applications. *IEEE/ACM Transaction on Networking*, 11, 17–32.

[30] Merkle, R.C. (1980) Protocols for Public Key Cryptosystems. In *1980 IEEE Symposium on Security and Privacy*, pp. 122–133. IEEE Computer Society Press, U.S, Oakland, CA, USA 14-16 April.

[31] Merkle, R.C. (1988) A Digital Signature Based on a Conventional Encryption Function. In Pomerance C. (eds), Advances in Cryptology — CRYPTO '87. CRYPTO 1987. Lecture Notes in Computer Science, vol 293. Springer-Verlag, Berlin, pp. 369–378.

[32] Why Every Bitcoin User Should Understand SPV Security (2017) https://medium.com/@jonaldfyookball/why-every-bitcoin-user-should-understand-spv-security-520d1d45e0b9 (10,10,2019).

[33] Maymounkov, P. and Mazieres, D. (2002) Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In Druschel P., Kaashoek F., Rowstron A. (eds) Peer-to-Peer Systems. IPTPS 2002. Lecture Notes in Computer Science, vol 2429. Springer, Berlin, Heidelberg,, pp. 53–65.

[34] etherchain.org – The Ethereum Blockchain Explorer. (2017) https://www.etherchain.org (14,10,2019).

[35] LeMahieu, C. (2017) Nano: a feeless distributed cryptocurrency network. https://content.nano.org/whitepaper/Nano_Whitepaper_en.pdf (5,11,2019).

[36] Raiden network 0.100.3 Documentation. (2019) https://raiden-network.readthedocs.io/en/stable/ (13,12,2019).

[37] Ethereum White Paper. (2015) A next-generation smart contract and decentralized application platform. https://github.com/ethereum/wiki/wiki/WhitePaper (17,10,2018).