2017 IEEE 23rd International Conference on Parallel and Distributed Systems

Blockchain-Based Government Information Resource Sharing

Liang Wang, Wenyuan Liu, Xuewei Han
School of Information Science and Engineering
Yanshan University
Qinhuangdao, China
lyonwong@126.com, wyliu@vip.163.com

***Abstract*—Government information resource (GIR) sharing, an important means for efficient government working, requires new technology to enhance its reliability and security. Blockchain, as an emerging technology for building decentralized applications, is gradually penetrating various fields. In this paper, we present a technical combination that consists of Blockchain structure, network sharing model and consensus algorithms. With these techniques, we design and implement a Blockchain-based GIR sharing system (BGIRSS), which is decentralized, to make the sharing procedure more efficient. The results of a series of emulational experiments show that our system is securer and more reliable than conventional sharing schemes, and can effectively promote the sharing efficiency of GIRs with lower implementation cost.**

Keywords- Blockchain; government information resource; decentralization; consensus; BGIRSS

I. Introduction

By building trust in a decentralized way, Blockchain technology has contributed distinctive solutions to diversiform application areas, especially in finance [1]. Its fundamental is well introduced by Satoshi [2] in his paper on Bitcoin. Over the past decade, Blockchain has undergone many technological innovations, including Smart Contract and various Blockchain service platforms [3]. Blockchain has been breaking away from domanial restriction, and becoming a technical standard which can be fitted into more scenarios.

We consider a scenario outside financial realm. It is hoped that government information resources (GIRs) should be fully grasped and utilized among departments to make governments more efficient. In this paper, a GIR means the useful information expressed by a group of data that come from business information systems (BISs) during the processes of government performance. GIRs are commonly shared by administrative means, this centralized way may lead to concerns about sharing efficiency and accountability. Another more technical solution is to build a data center to collect the GIRs of all departments, and then to publish them. Obviously, this solution centralizes both management and data, even it eases sharing tasks. It also increases the development cost and gives attackers a chance to steal all the data through intruding into the data center. Actually, it is cost-effective and low security risk to leave GIRs where they belong to, and to share them only when needed.

The basis of sharing is trust, and trust is based on security. Blockchain built on a decentralized peer-to-peer network uses a series of cryptographic techniques to make it tamper-resistant and encrypted, and is shared in the network by consensus. The network is not controlled by any single node, and this allows each participant to share data without having to establish trust with its counterparties. The scenario of GIR sharing, which differs from that of payment or exchange, exactly provides a distributed circumstance and application requirements for Blockchain. Reconstructing of the GIR sharing model using Blockchain technology is a beneficial attempt in a nonfinancial field.

The main contributions of this paper are as follows:

- We propose a decentralized sharing model to solve the problems in GIR sharing, such as efficiency, reliability and security.
- We design and implement a Blockchain-based GIR sharing system (BGIRSS) with reconstructed Blockchain structure and matched consensus algorithms.
- We establish an emulational testbed to test and evaluate the effectiveness, reliability and security of our system.

The rest of the paper is organized as follows. Section II sketches the related work. Section III compares the GIR sharing models. Section IV elaborates the specially designed Blockchain structure. Section V describes the establishment of the network. Section VI gives details about system tests and evaluation. Section VII concludes this paper.

II. Related Work

There would be a lot of synergy among government departments if they shared their information resources efficiently. So, GIRs can be regarded as digital assets that are in the form of data stored in computers. The GIR sharing typically consists of three basic phases. The first phase is to carry out a survey on GIRs in every department. The outcome of the survey is a catalog of GIRs (GIRC), by which departments can send their sharing requests to others. They will really get the data they needed by online or offline ways in the second phase, while the last phase is to keep the GIRC up-to-date and to maintain the sharing records. Our work focuses on solving technical problems in the latter two phases. According to the layered architecture of Blockchain application described by Yuan [4], we provide a concrete implementation for the Blockchain-based sharing model to make sharing tasks more efficient. In addition, research on other issues raised by this model will complement this work.

III. Sharing Model

Two common GIR sharing models are currently available without regard to trust, accountability, and security. One is the managerially centralized sharing model (MCSM) (Fig. 1), and the other is the data-specific centralized sharing model (DCSM) (Fig. 2). The problem with MCSM is that the sharing processes are controlled by a few specific nodes, such as SSd in Fig. 1. This often leads to inefficient sharing because of too much coordination.

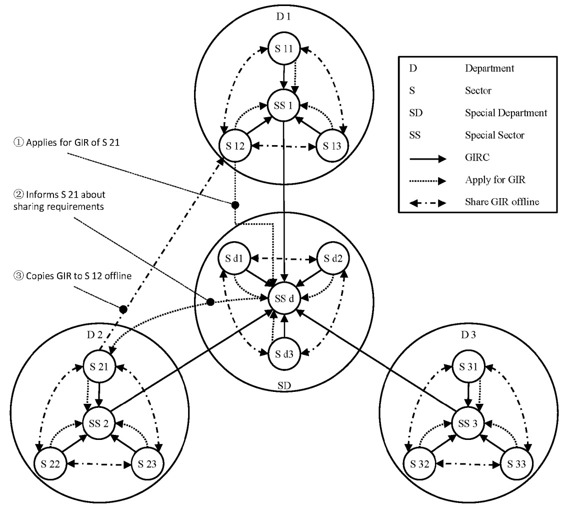

Figure 1. MCSM

Integrating all the sharable data into a data center can indeed make the sharing more efficient, but increase the risk of the single node attacking. Fig. 2 illustrates how the data centers work. Meanwhile, construction and maintenance of the data center require a large investment.

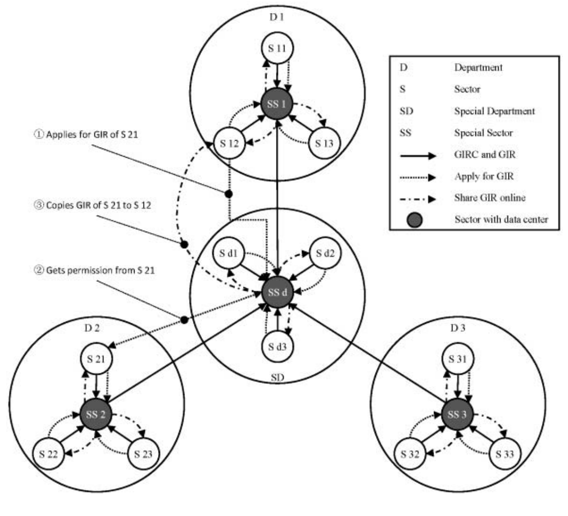

Figure 2. DCSM

Now we consider a decentralized sharing model (DSM). As shown in Fig. 3, nodes are interlinked without any center. In this model, each node only holds its own GIRs, but all nodes maintain the GIRC and a same version of Blockchain. The workflow of the model is demonstrated below.

i. S12 looks up its own GIRC for the data it wanted and sends a sharing request directly to S21.

ii. If S21 accepts the request of S12, the two parties will establish a connection, through which S21 will copy the data directly to S12.

iii. Write this sharing process to Blockchain as a transaction, and make consensus on it in the whole network.

Compared with MCSM and DCSM, the virtues of this model are as follows:

- The topology of the network is decentralized.
- Liability accidents can be partially avoided as the sharing events recorded on Blockchain are tamper-proof and traceable.
- A few broken-down nodes will not affect the normal operation of the whole network.

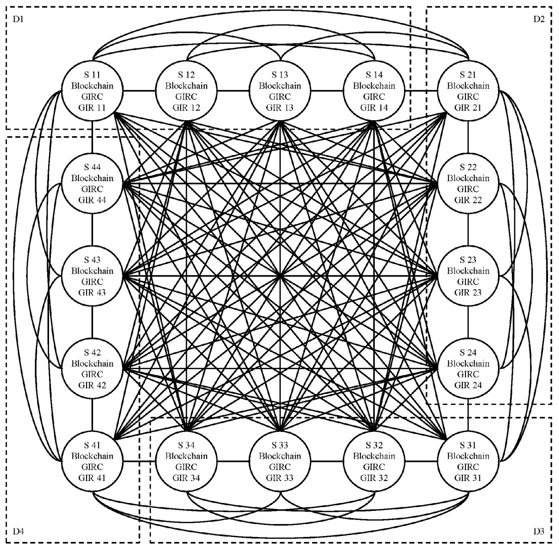
D1
D2
D3
D4
S 11 Blockchain GIRC GIR 11
S 12 Blockchain GIRC GIR 12
S 13 Blockchain GIRC GIR 13
S 14 Blockchain GIRC GIR 14
S 21 Blockchain GIRC GIR 21
S 22 Blockchain GIRC GIR 22
S 23 Blockchain GIRC GIR 23
S 24 Blockchain GIRC GIR 24
S 31 Blockchain GIRC GIR 31
S 32 Blockchain GIRC GIR 32
S 33 Blockchain GIRC GIR 33
S 34 Blockchain GIRC GIR 34
S 41 Blockchain GIRC GIR 41
S 42 Blockchain GIRC GIR 42
S 43 Blockchain GIRC GIR 43
S 44 Blockchain GIRC GIR 44

Figure 3. DSM

IV. Blockchain Structure

Each node in the network is equipped with a node-side software referred to as ShareClient we design to implement the following two interfaces:

- Catalog: for storing and maintaining GIRC.
- Wallet: for storing and maintaining GIRs owned or shared.

Generally, a GIR is a structured data set whose summary is composed of GIR's primary metadata as shown in TABLE I. When a GIR is registered, its summary will be appended to the GIRC, while the GIR will be divided into several parts, each of which is denoted by GIRPart. Fig. 4 demonstrates the registration of a GIR.

Unlike Bitcoin [2], there is no double-spending [5] problem with GIR sharing. Therefore, we define a concatenate structure for the GIR as shown in Fig. 5. This structure keeps the owners' actual control over the direction of the transactions. Each sharer only stores the handle of the latest transaction in its Wallet, and this leaves it with no chance to share the same data to a third party through ShareClient.

TABLE I. GIR SUMMARY

Metadata	Description	Options	Default
GIRCode	The unique identification of this GIR.	--	--
Title	The title of this GIR.	--	--
Depiction	The description of this GIR.	--	--
Owner	The possessor of this GIR.	--	--
UpdateRate	The frequency of update to this GIR.	--	real time
SharingMode	The condition for sharing this GIR.	limited, unlimited, forbidden	limited
DataAttributes	The data attributes of the elements in this GIR including Id, name, meaning, datatype and sharingmode.	--	--

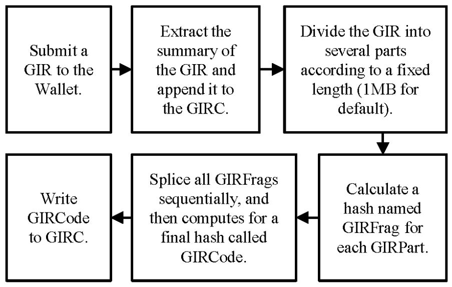
Submit a GIR to the Wallet.
Extract the summary of the GIR and append it to the GIRC.
Divide the GIR into several parts according to a fixed length (1MB for default).
Calculate a hash named GIRFrag for each GIRPart.
Splice all GIRFrags sequentially, and then computes for a final hash called GIRCode.
Write GIRCode to GIRC.

Figure 4. GIR registration

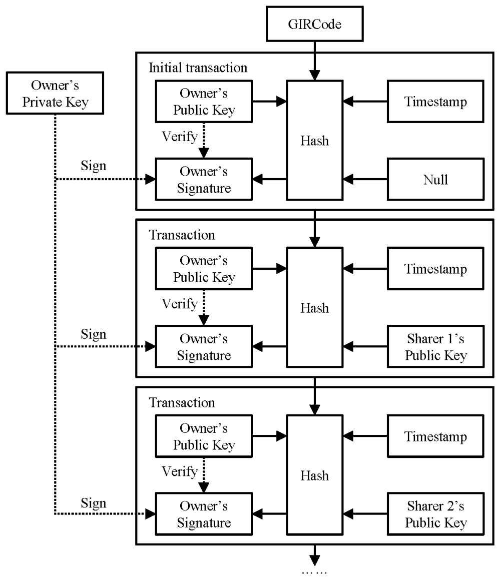

Figure 5. GIR structure

Transactions cannot be removed for the reason of releasing disk space, so that they can be traced. Instead of using a Merkle tree to map multiple transactions, we let a block only mount one transaction. Fig. 6 shows the structure of our designed Blockchain.

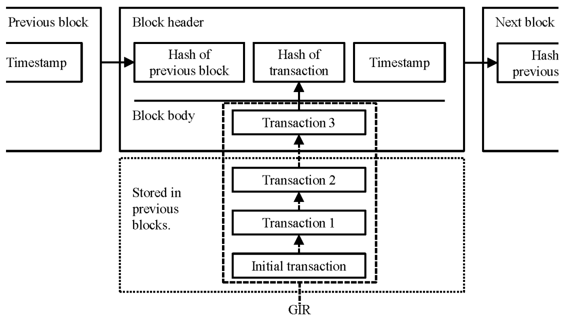

Figure 6. Blockchain structure

V. NETWORK

A. Communications

To build a peer-to-peer network among different intranets requires Network Address Translator (NAT) traversal through UDP Hole Punching techniques [6] [7] [8]. Fig. 7 demonstrates how two allopatric nodes communicate.

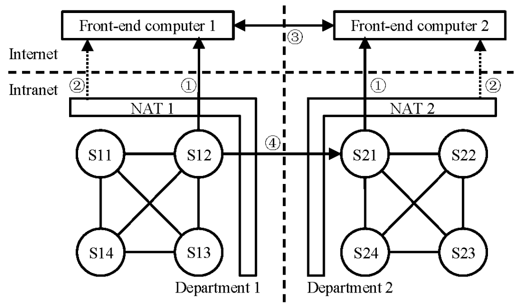

Figure 7. Communicating between intranets

ShareServer, the server-side program we install on the front-end computers, is integrated with STUN Server [9] [10]. It takes on the address registration of nodes and maintains an address table with all other ShareServers through PBFT [11]. TABLE II is an illustration of an address table.

TABLE II. ADDRESS TABLE

Node Id	Inner IP	Inner Port	Outer IP	Outer Port	Org. Code	Status
(Node's public key)	192.168. 0.24	3200	123.180. x.x	5000	Tax number	online
(Node's public key)	192.168. 1.143	2700	56.134. x.x	6000	Tax number	online
…	…	…	…	…	…	…

B. Consensus

The following steps are the consensus on an initial transaction.

i. GIR owner bundles the initial transaction, GIRC, GIRFrags and GIRCode into a packet and broadcasts it to all nodes.

ii. Each node completes a verification with the GIRCode and GIRFrags in the packet, and then broadcasts a tag.

iii. Once receiving the tags from more than half nodes of the network, GIR owner would create a new block with the initial transaction and broadcast it together with GIRC to all other nodes. If enough other tags are received prior, GIR owner must wait for that block first.

iv. Each node chains the new block by hash order, and updates the GIRC.

The qualification to record sharing transactions depends on a broad consensus across the network. The steps are as follows:

i. GIR owner sends GIRParts to the sharer one by one.

ii. Once receiving a GIRPart, the sharer would broadcast the matched GIRFrag to all nodes.

iii. GIR owner broadcasts the sharing transaction, GIRCode and GIRC to all nodes.

iv. GIR sharer receives GIRCode and then also broadcasts it to all nodes.

v. Each Node verifies the sharing transaction with GIRFrags and GIRCode sent by GIR owner, and broadcasts a tag to the entire network only when it receives the correct GIRCode from GIR sharer.

vi. When a node other than GIR owner and sharer is the first to receive more than half tags of the network, it will create a new block with the sharing transaction and broadcast it together with GIRC to all nodes.

vii. Each node chains the new block by hash order, and updates the GIRC. (The earliest version of the block will be reserved.)

VI. Evaluations

According to the principle shown in Fig. 7, we build an emulational testbed to evaluate the reliability and security of our designed system in several measures. The experimental environment is two local networks comprising ten PCs and two routers, as shown in Fig. 8. The routers simulate the NATs, while eight PCs act as the nodes, and the other two PCs direct connect to the Internet to serve as the front-end computers.

A. Storage and Load

A block is about 220 bytes, with transactions taking up close to 150 bytes. The GIR sharing businesses have no fixed frequency, so we assume an extreme case where GIR sharing would happen every one minute, i.e. one block per minute, 220bytes * 60 * 24 * 365 ≈ 110MB per year. This is not an issue for modern computer storage capacity.

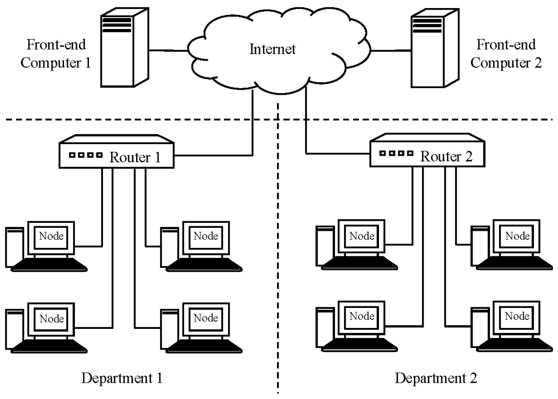

Figure 8. Emulational testbed

On our testbed, we first allocate a fixed number of GIRs (2MB per GIR) for each node. Then, the initial transactions are concurrently executed by multithreaded programs to simulate the simultaneous work of multiple nodes. The result is shown in Fig. 9.

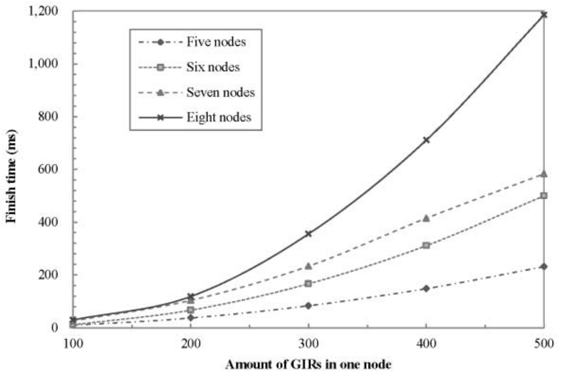

Figure 9. System load of initial transactions

The average speed of block creation (denoted by *asbc*) with initial transactions can be estimated by

$$asbc = b\delta \,/\, \omega \,, \tag{1}$$

where b is the bit rate of the network (1000Mbps), ω denotes the total among of data in a transaction, and δ is a constant equal to 131072. The ω of an initial transaction is about 1950 bytes. Theoretically, *asbc* can be calculated to be 67216 blocks per second. The decrease from theoretical rate to actual rate in Fig. 9 results from the time consuming on multithread computing, transaction verifying, and communication delay brought by adding nodes.

For sharing transactions, we can effectively avoid network congestion because we split a GIR into several smaller parts and send them one by one. In the experiment, each node sends GIRs to other nodes, but the recipients do not send GIRs to the sender in the meantime. Thus, the total of transactions can be calculated by

$$trans = g\sum(\, n-1 \,), \tag{2}$$

where n denotes the total of nodes and g represents the number of GIRs sent by a node concurrently. The result is shown in Fig. 10.

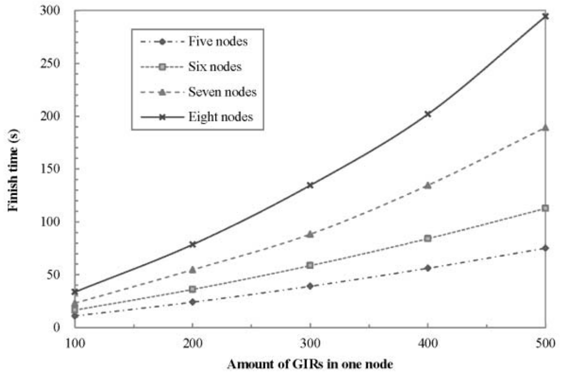

Figure 10. System load of sharing transactions

Now the ω of a sharing transaction is about 1050590 bytes. According to (1), the *asbc* is about 124 blocks per second. The system is still up to this extreme load.

B. Running Stability

1) Fault tolerance: Subjected to the consensus algorithms detailed in Section V, a few of failed nodes will not affect the running of the network. Let n be the total of nodes and f, which is called fault tolerance, be the number of failed nodes that can be allowed on the network, then the following equation holds:

$$f = (n - 2) - \lceil n / 2 \rceil . \tag{3}$$

With calculation, four online nodes are the minimum for start-up, but only if that number reaches six, the failed node may be allowed to exist.

2) Scalability: Let t be the longest confirming time of a transaction for one to tolerate, and n be the number of involved nodes the network can afford to. Assume that the network is in the case of concurrent execution, then n should satisfy

$$\omega \sum (n - 1) < \delta b t, \tag{4}$$

where ω is 1050590 bytes of s sharing transaction, b and δ has been defined in (1). If t is 60 seconds, then n should not exceed 122. We conduct an experiment on our testbed to verify the above calculation. The result is shown in Fig. 11. As seen, the theoretical values and observed values are well fitted.

3) Transaction reachability: There is only one transaction in one block, and a block is version identified by its creation time. These make every transaction not invalidated by the rejection of a block, and meet the high expectation of the reachability of the GIR sharing tasks.

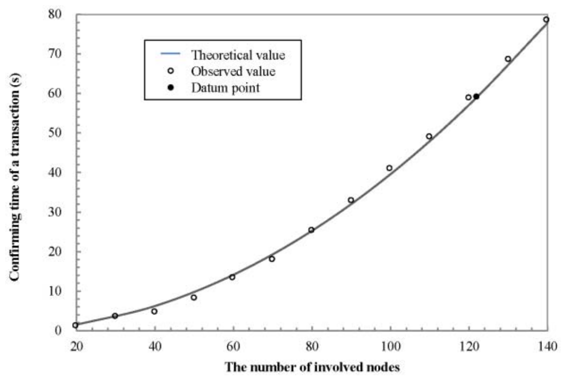

Figure 11. System scalability

C. Data Analysis

By traversing the Blockchain, the sharing history of each GIR can be calculated, on the one hand, to show the utilization of GIRs, and on the other hand, to understand the needs of different departments for GIRs. The algorithm below describes the traversal process on Blockchain.

```
Algorithm: BlockchainScan
Input:
  b, handle of the latest block in the chain.
Output:
  ratio, a set of key-value pairs, each of which
holds the number of sharing times of a GIR.
  req, a group of arrays, each of which lists
the GIRs a node has shared from other nodes.
Declaration:
  t, pointer to a transaction in block.
  trans, a field of block refers to a
transaction data structure.
  r, counter of GIR sharing times.
  toString(), a function used to convert the
content of a variable to a string.
  s, string of public key.
  sharer, a field of transaction refers to the
public key of sharer.
  prev, a field of block points to the hash of
previous block.
Method:
  while b != null do
    t = b.trans;
    r = 0;
    while t.hash != null do
      r = r + 1;
    endwhile
    if ratio(t.toString()) == null then
      ratio.add(t.toString(), r);
    endif
    s = b.trans.sharer;
    if s != null then
      req(s).add(t.toString());
    endif
    b = b.prev;
  endwhile
  return ratio, req;
```

D. Tamper-resistance

Our designed Blockchain is anti-fake relies on mutual supervision of all nodes, rather than comparing the lengths of the chains. So if one tampered the transactions on its own chain, it would have to download a true chain from a near node as it could never chain a new block.

E. Anti-attack

1) Node masquerading: The way our system defending a simulate node is revealed in the following attacking steps of an attacker:

i. Incurs front-end computer to get the address table.

ii. Selects a node A in the address table, downloads its GIRC, and makes it disconnected.

iii. Masquerades node A's IP address and port number, and sends a sharing request to another node B according to GIRC.

iv. Node B seeks the address table for node A's Id to verify the signature of the sharing request sent by the attacker.

v. The attack failed due to invalid authentication.

2) DDoS attack: As mentioned earlier in this section, if you want to create more than one minute of network congestion, you need to turn at least 122 nodes into puppets [12], otherwise the network will digest these malicious packets.

3) 51% attack: Same as Bitcoin, the ability to defend against the 51 % attack comes from the motivation rather than a technical way. That is, if one can master 51% of the nodes in the Intranet, he would have already got what he wants. Therefore, our system cannot replace firewall and antivirus software.

VII. Conclusions

This paper designs and implements a Blockchain-based government information resource sharing system. In view of the shortcomings of current GIR sharing strategies in efficiency, reliability and security, we put forward a decentralized data sharing model. The establishment of peer-to-peer network fully considers the features of government Intranet, and our proposed Blockchain data structure with matched consensus algorithms satisfy the three main reliability requirements of GIR sharing: sharing events tamper-resistance, sharing history traceability and sharing source purity. The elaborate evaluation experiments based on emulational testbed demonstrate our proposed model and system are effective, reliable and secure, especially take on good stability of the network and capability of data analysis.

Summing up the above, taking Blockchain as an independent technology to solve GIR sharing issues provides a new thinking for involved developers and practitioners. The future work will focus on further improving of the stability, security, scalability and applicability of our model and system such as:

- developing new fault-tolerant hash algorithms to identify the similarity of two GIRs,
- addressing the issues of UDP packets dropping and disordering,
- making further efforts to optimize the performance of the system load,
- reinforcing the defense of the system against more attacking ways,
- trying to improve the RSA algorithms to prevent fake nodes from faking the public keys,
- and so on.

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China (61672448, 61772453) and the Technology Foundation for Selected Overseas Chinese of Hebei Province (CL201625).

References

[1] Jesse Yli-Huumo, Deokyoon Ko, Sujin Choi, Sooyong Park, and Kari Smolander, “Where is current research on Blockchain technology?-A systematic review,” PLOS One, vol. 11, no. 10, e0163477, 2016.

[2] Satoshi Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," Consulted, www.bitcoin.org, 2008.

[3] M. Swan, “Blockchain: blueprint for a new economy,” O'Reilly Media, Inc., 2015.

[4] Yong Yuan and Feiyue Wang, "Blockchain: the state of the art and future trends," Acta Automatica Sinica, vol. 42, no. 4, pp. 481-494, 2016.

[5] G. O. Karame, E. Androulaki, and S. Capkun, “Double-spending fast payments in bitcoin,” in Proceedings of the ACM Conference on Computer & Communications Security, pp. 906-917, 2012.

[6] Arno Wacker, Gregor Schiele, Sebastian Holzapfel, and Torben Weis, "A NAT traversal mechanism for peer-to-peer networks," in Proceedings of the Eighth International Conference on Peer-to-Peer Computing, pp. 81-83, 2008.

[7] P. Srisuresh, B. Ford, and D. Kegel, "State of Peer-to-Peer (P2P) communication across Network Address Translators (NATs)," RFC-5128, 2008.

[8] Y. Takeda, "Symmetric NAT traversal using STUN," Internet Engineering Task Force, Internet Draft, 2003.

[9] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, et al., "SIP: Session Initiation Protocol," RFC-3261, 2002.

[10] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) through Network Address Translators (NATs)," RFC-3489, 2003.

[11] Miguel Castro and Barbara Liskov, "Practical Byzantine fault tolerance," in Proceedings of the Third Symposium on Operating System Design and Implementation, pp. 1-14, 1999.

[12] R. K. C. Chang, "Defending against flooding-based distributed denial-of-service attacks: a tutorial," IEEE Communications Magazine, vol. 40, no. 10, pp. 42-51, 2002.