J Grid Computing (2020) 18:629–638
https://doi.org/10.1007/s10723-020-09535-x

AirCargoChain: A Distributed and Scalable Data Sharing Approach of Blockchain for Air Cargo

Gejun Le · Qifeng Gu · Qiang Qu · Qingshan Jiang · Jianping Fan

Received: 5 August 2019 / Accepted: 31 July 2020 / Published online: 5 December 2020

Abstract Air cargo involves large-scale data and multiple stakeholders, i.e., airports, airlines, agents, and clients. How to enable stakeholders to share data in a secure way is essential, since it improves efficiency for various processes among stakeholders. This paper proposes AIRCARGOCHAIN, a blockchain-based data sharing approach for air cargo, which has the following advantages: (a) secure: we propose a blockchain-based cooperative network architecture, Cooperative Network (CN), to allow mutually distrusted stakeholders to manage data collaboratively. (b) scalable: we design a storage scheme, Off-Chain Storage (OCS), based on IPFS to support large-scale data storage; (c) user-friendly: we provide effective communication mechanism and convenient contract name service in Node Communication (NC) and Contract Management (CM), respectively. A comprehensive evaluation offers insight into the applicability and effectiveness of AIRCARGOCHAIN.

Keywords Blockchain · Smart contract · Data sharing · Air cargo

G. Le
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518055, China

G. Le · J. Fan
University of Chinese Academy of Sciences, Beijing, 100049, China

Q. Gu
School of Economics and Management, Beijing University of Posts and Telecommunications, Beijing, 100867, China

Q. Qu (✉)
Guangdong Provincial R&D Center for Blockchain, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518055, China
e-mail: qiang@siat.ac.cn

Q. Jiang
Shenzhen Key Laboratory for High Performance Data Mining, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518055, China

1 Introduction

Air cargo involves multiple distributed stakeholders and goods, which requires efficient data sharing to improve business process, e.g. freight checking. In freight transport, stakeholders need to check freight according to waybills that contain large-scale sensitive data, such as identity information, address, and telephone. Secure data sharing among multi-stakeholders is important, since it accelerates business process. But the multiple challenges are still existing: (a) for stakeholders, they have to perform freight checking in information asymmetry, which leads to low efficiency; (b) for sensitive data, the privacy preservation of data decides we cannot use central information system to manage data simply; (c) for storage, ever increasing volumes of data demand a scalable solution.

Blockchain technology is initially used in financial sector [1]. In recent years, blockchain has found

applications in a variety of other non-financial fields [2–4] as it is a peer-to-peer network that enables mutual distrusting nodes to maintain a distributed, consistent, and tamper-proof ledger collaboratively. However, using blockchain technology as data sharing systems in air cargo may bring other issues, for example, (a) data stored in a ledger is shared among nodes, which may expose sensitive information if we do not consider an access control method; (b) storing large-size data in a blockchain that uses up storage resources as every node maintains a complete ledger; (c) some business communications do not require irrelative stakeholders participate; (d) account address is too complicated for users to remember, since it is a fixed-length hexadecimal string.

This paper presents AIRCARGOCHAIN, a distributed and scalable data sharing approach for air cargo. AIRCARGOCHAIN proposes a cooperative network architecture, Cooperative Network (CN), based on blockchain to manage data collaboratively among mutual distrusted stakeholders. Each airport and airline has an Externally Owned Account (EOA) that is able to create Contract Account (CA) for each agent and client. CA records identity information and waybills. As for privacy-preserving, an access control method is proposed to ensure that only airports, airlines and CA owners can access CA and view details. Considering the limited storage capacity of each blockchain node, we design a storage scheme, i.e., Off-Chain Storage (OCS). OCS utilizes InterPlanetary File System (IPFS) [5] to store waybills information and then stores hash values generated by IPFS in CA. IPFS is a content-addressable storage that distributes data without redundancy and generates a unique hash string as Uniform Resource Identifier (URI) to retrieve the data. Last but not least, we provide a user-friendly service based on Advanced Messages Onchain Protocol (AMOP) and Contract Name Service (CNS). AMOP enables two nodes to efficiently communicate without being noticed by others. CNS allows users to assign a simple name instead of complicated contract address.

The advantages of AIRCARGOCHAINare

- **Secure**: we propose a data sharing approach for air cargo based on blockchain technology, which not only enables mutual distrusted stakeholders to manage data collaboratively but also ensures user privacy and data security.
- **Scalable**: we design a high scalable storage scheme to support large-scale data storage that reduces the consumption of storage resources.
- **User-friendly**: we provide user-friendly functionalities that contain an effective communication mechanism and a convenient contract name service.

The rest of this paper is structured as follows. Section 2 reviews blockchain technology and the related work of blockchain-based data sharing. Section 3 presents the problem description of data sharing in air cargo. Section 4 describes AIRCARGOCHAIN, including system model, main procedures and security analyses, which is followed by the implementation and experimental reports in Section 6. Section 7 concludes the paper.

2 Related Work

This section first reviews blockchain technology, and then discusses the existing projects of blockchain-based data sharing.

2.1 Blockchain technology

Blockchain is a shared ledger that stores append-only data. It is originally derived from Bitcoin [6], then has emerged a large number of applications in the financial domain [7–10]. In recent years, much attention has been attracted to applications of blockchain in the non-financial sectors such as healthcare, internet of things, and supply chain [11–13], for the following four reasons:

- **Decentralization:** A blockchain is a peer-to-peer network that consists of multiple nodes which do not fully trust each other. It enables nodes to maintain a set of global states without a trust third-party or central administrator. Each node keeps an entire replicated ledger and no Single Point of Failure (SPoF) [14] can affect the availability of the stored information [29].
- **Tamper-proofing:** A blockchain provides integrity protection for all data recorded in the ledger via its append-only linked block structure. This structure makes adversary hard to modify data because any modification needs to modify all the previous blocks in the chain.

- **Fraud prevention and consistency:** Nodes in a blockchain network work collaboratively by using a consensus protocol, such as Proof of Work (POW) [6], Proof of Stake (POS) [15], and Practical Byzantine Fault Tolerance (PBFT) [16]. A consensus protocol manages the right of creating new blocks and determines the next block, which not only ensures consistency but also makes fraud impossible.
- **Smart contract:** Smart contract [17] is a modular, reusable, and automatically executed script that runs on a blockchain, which improves efficiency of business processes. When a transaction is sent to the contract address and trigger the predetermined condition, all nodes execute the operation code generated by the script compilation of this contract, and finally write the execution result to the blockchain.

2.2 Blockchain-based Data Sharing

Plentiful solutions of blockchain-based data sharing [18–24] have been proposed during the last decade. Here, research trends pertaining to blockchain-based data sharing for supply chain are outlined.

Lei Xu et al. [25] presented a set of protocols for blockchain-based efficient information sharing and exchange for maritime transportation. They focused on how to maintain information consistency between the ledger and reality. However, they neglected the data privacy and ownership, exposing sensitive information in transactions to all participants. Quansi Wen et al. [26] proposed a blockchain-based data sharing scheme for supply chain. They stored real-time data in a blockchain and set access policies by smart contract, ensuring the reliability and privacy of data sharing in supply chain. But they did not take into the scalability, using up a large number of storage resources. Ingo Weber et al. [27] designed a blockchain-based system to address the lack-of-trust issue in collaborative business processes. It was the first to use blockchain for collaborative process execution and monitoring. Although they discussed data privacy and off-chain data storage in their paper, they did not demonstrate the details of their scheme. The studies [28–31] introduced spatial and temporal blockchain methods to enable blockchain applicable for broad scenarios.

In summary, privacy concerns and storage limitations should be carefully addressed when employing a blockchain system as a data sharing tool. Our work not only proposes a role-based access control, but also designs a high scalable storage scheme to reduce the consumption of storage resources.

3 Problem

This section describes the problem of secure data sharing among multiple stakeholders in air cargo, including potential threats and design goals.

3.1 Potential Threats

Table 1 shows multiple potential threats in air cargo. Our work needs to avoid these risks.

- **DT** denotes *Data Tampering*, i.e., data (e.g., waybills data and identity data) is maliciously modified.
- **IL** denotes *Information Leakage*, i.e., agents/clients' information is leaked to unauthorized third-party.
- **MSA** denotes *Message Spoofing Attack*, i.e., a malicious stakeholder may broadcast fake messages to other stakeholders.
- **SPOF** denotes *Single Point of Failure*, i.e., a node failure stops the entire system from working.
- **Waybills-based Solutions** represent original manual solutions without digital system for data sharing. They rely on a large quantity of human resources and paper-based documents (e.g., waybills) to execute business processes.
- **Centralized Systems** represent traditional air cargo systems that utilize centralize services and store data in a central database.
- **AirCargoChain** is a proposed scheme in the paper.

Table 1 Potential risks

	Waybills-based solutions	Centralized systems	AirCargoChain
DT	•	•	–
IL	•	–	–
MSA	•	•	–
SPOF	–	•	–

Examples of potential risks are as follows:

- Waybills-based Solutions: In waybills-based solutions, data is recorded in paper documents. The data is easy to obtain and modify once a malicious user gets paper documents (i.e., **DT**, **IF**). There is no message verification mechanism, causing waybills-based solutions are vulnerable to fake messages (i.e., **MSA**).
- Centralized Systems: A centralized system has an administrator with the highest authority. First, an attacker invades the system and acquires administrator privileges. Next, the attacker is able to modify waybills data to maximize his/her profits or broadcast fake messages about the information of cargo to swindle profits (i.e., **DT**, **MAS**). Finally, any centralized servers or databases fail, which interrupts the entire business processes and results in significant economic losses (i.e., **SPOF**).
- AirCargoChain: we present the security analysis of AIRCARGOCHAINin Section 4.

3.2 Design Goals

This paper focuses on exploring a distributed and scalable data sharing method, aiming at business processes acceleration and information security protection for air cargo. Therefore, the design of AIRCARGOCHAIN should achieve the following seven goals:

- **Privacy protection:** Waybills contain sensitive data such as identity, address and telephone. To protect the privacy of users, it is necessary to design a rule to prevent unauthorized third parties from accessing data.
- **Decentralization:** Air cargo involves multiple distributed stakeholders and storing a large amount of sensitive data in a centralized database increases security risks. Thus, AIRCARGOCHAINought to take full advantages of distributed nodes, ensuring the security of stored data.
- **Consistency:** Multiple stakeholders often execute operations concurrently. Thus, it is essential to maintain data consistency among decentralized stakeholders.
- **Fraud prevention:** Air cargo is intricacy and distrusting. Any role has the probability to broadcast fake messages that disturbs business processes. Therefore, how to prevent fraud is worth exploring.
- **Tamper-Proofing:** A tamper-proof design is considerable because data tampering impairs the reliability and results in economic losses.
- **Scalability:** With the growth number of cargo volume, the data continues to increase. Due to the limited resources, it is not suitable for storing all data in blockchain.
- **User-friendliness:** It is important to consider a user-friendly design, which improves user experiences.

Table 2 Symbols and Definitions

Symbols	Definitions
A_{user}	User's address
I_{user}	User's identity information
N_{user}	User's contract name
$D_{waybill}$	Waybill data
$H_{waybill}$	URI of waybill data stored in IPFS
CN	Cooperative Network
OCS	Off-Chain Storage
NC	Node Communication
CM	Contract Management

4 Proposed Method

This section presents system model and four main processes. Table 2 gives the list of symbols we use.

4.1 System Model

As shown in Fig. 1, AIRCARGOCHAIN has four roles (i.e., airport, airline, agent and client) and an AirCargo Information Platform (ACIP). The ACIP contains four modules: Cooperative Network (CN), Off-Chain Storage (OCS), Node Communication (NC), and Contract Management (CM), the detailed design of ACIP is presented as follows.

- **CN:** The revised version of Ethereum[1] is used to implement Cooperative Network. It is a blockchain network consisting of decentralized

[1] https://github.com/FISCO-BCOS

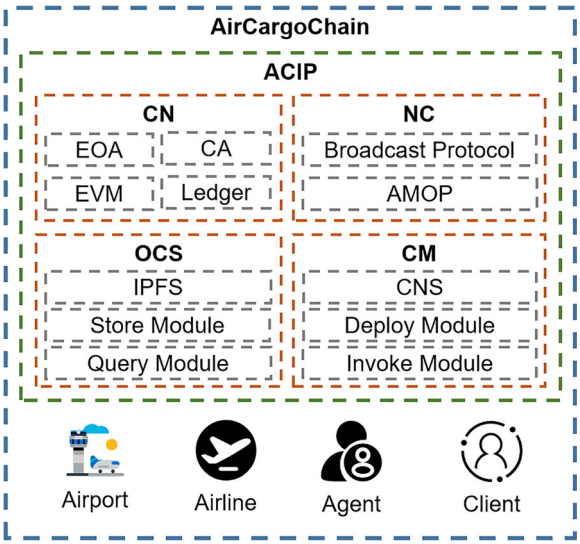

Fig. 1 The module design of the proposed system

peers that can communicate with each other and maintain a consistent ledger collaboratively. We utilize the features of blockchain itself to ensure data is tamper-proofing. In CN, Airport/Airline creates CA for Agent/Client by EOA. CA, which is run in EVM, records Agent/Client's identity information and the URI of its waybills in Ledger. We use PBFT to manage block generation in case of fraud messages and inconsistency among nodes.

- **CM:** All smart contracts are managed by CM. A fixed-length hexadecimal contract address is generated once a smart contract is deployed. A contract address is used to invoke the smart contract. We employ a Contract Name Service (CNS) to make a mapping between contract address and contract name. In other words, when a smart contract is deployed, the CNS allows the deployer to assign a string to the smart contract. Users just need to provide the string when they want to invoke the contract. Next, CNS confirms the contract address corresponding to the string, and invoke this contract. On the other hand, to protect privacy, each smart contract includes a declarative access control that can determine which users can read, edit, and update elements in network.
- **NC:** NC involves communication over the network. The broadcast protocol of the blockchain network is similar to Gossip: for newly generated transactions, they are broadcast to every node; for transactions from other nodes, they are broadcast again to 25% nodes which are randomly selected; a transaction is broadcast only once on a node. The Advanced Messages Onchain Protocol(AMOP) provides message channel on the blockchain network where all nodes can communicate once the blockchain is deployed. The AMOP messages do not rely on consensus and do not be stored in the blockchain.
- **OCS:** Traditional storage schemes store all data in a blockchain. With the rapid increase of cargo data, traditional storage schemes are impractical because of the limited disk spaces. Leveraging the distributed features of a blockchain network, we build an InterPlanetary File System to store large-size waybills data. The IPFS provides a high-throughput block storage model with content-addressed hyperlinks. A file is broken down into many smaller blocks when adding the file to IPFS. The address of blocks is resolved to a hash value that links to those blocks. The hash value of the file is generated according to file content, so the hash value changes whenever the file is modified. In short, it not only improves the security of data storage but also saves storage resources.

4.2 Main Processes

This section presents three main processes in AIRCARGOCHAIN: (1) registration; (2) data storage; (3) data access.

4.2.1 Registration

First, agents/clients need to register CA for joining the blockchain. Algorithm 1 illustrates a user's operations in the registration process.

Assuming that $user_1$ is a user who wants to register CA. In order to avoid repetitive registrations, we need to verify whether the identity information (I_{user_1}) or the contract name (N_{user_1}) is used (Line 1). $checkInfo$ denotes a function of verification. If I_{user_1} and N_{user_1} are not occupied, CA is registered for the user (Line 3). $register()$ denotes a function of deploying CA. A_{user_1} denotes the contract address of $user_1$'s CA. After successful deploy, a mapping is set between I_{user_1} and N_{user_1} (Line 4). Otherwise, users are warned of registration failed (Line 6).

Algorithm 1: Registration.

Input: I_{user_1}, N_{user_1}
Output: $result$

```
1 isExist ← checkInfo(I_user1, N_user1);
2 if isExist == flase then
3 |   A_user1 ← register(I_user1);
4 |   result ← CNSS(I_user1, N_user1);
5 else
6 |   result ← false;
7 end
```

4.2.2 Data Storage

After owning CA, a agent/client is able to enjoy air cargo services. Algorithm 2 analyzes processes and functions required to create/update waybills in detail.

Algorithm 2: Data storage.

Input: N_{user_1}, A_{user_2}, $D_{waybill}$
Output: $result$

```
1  isValid ← checkPermission(A_user2);
2  if isValid == true then
3  |   A_user1 ← CNSM(N_user1);
4  |   operate ← confirmOperate();
5  |   if operate == addition then
6  |   |   D_waybill ← verifyWaybill(D_waybill);
7  |   |   H_waybill ← OCSS(D_waybill);
8  |   |   result ← CAS(A_user1, H_waybill);
9  |   else if operare == update then
10 |   |   H_waybill ← selectWaybill();
11 |   |   H_waybill ← OCSU(H_waybill, D_waybill);
12 |   |   result ← CAU(A_user1, H_waybill);
13 |   end
14 |   else
15 |   |   result ← false;
16 |   end
17 else
18 |   result ← false;
19 end
```

Assuming $user_2$ is an airport/airline that undertakes the transportation. A_{user_2} denotes $user_2$'s public key address. $checkPermission()$ is used to check user's permissions (Line 1). If a user is legal, $confirmOperate()$ gets user's contract address and confirms operation type currently (Line 3-4). $CNSM()$ is a mapping function between contract address and contract name. Addition operation includes three steps: first, $verifyWaybill()$ allows communication between carriers (the airport and the airline) to verify waybill information (Line 6); second, $D_{waybill}$ is stored in IPFS and $H_{waybill}$ is received form IPFS (Line 7); third, $H_{waybill}$ is stored in Ledger (Line 8). $OCSS()$ and $CAS()$ denote functions of storing data in IPFS and Ledger, respectively. Update operation also contains three steps: first, $selectWaybill()$ selects which waybill needs to be updated (Line 10); second, $OCSU()$ updates data in IPFS (Line 11); third, $CAU()$ updates data in Ledger (Line 12). Otherwise, a user is warned of storage failed.

4.2.3 Data Access

In AIRCARGOCHAIN, waybills query is a process of data access, which is exhibited in Algorithm 3.

Algorithm 3: Data access.

Input: N_{user_1}, A_{user_2}
Output: $result$

```
1 isValid ← checkPermission(A_user2);
2 if isValid == true then
3 |   H_waybill ← selectWaybill();
4 |   result ← OCSQ(H_waybill);
5 else
6 |   result ← false;
7 end
```

In the process of data access, user's permission first needs to be verified (Line 1). If a user is legal, user's CA is invoked and then the user selects which waybills to query (Line 3-4). Otherwise, a user is warned of query failed (Line 6). $QCSQ()$ is a function used to query data from OCS.

5 Security Analysis

This section presents the security analysis of the four potential threats described from Table 1 in Section 4, including Information Leakage (IL), Data Tampering

(DT), Message Spoofing Attack (MSA) and Single Point of Failure (SPOF).

- **DT:** We store $D_{waybill}$ in OCS and store $H_{waybill}$ in Ledger. A URI of waybill data stored in OCS is content-based. A new URI is generated once waybill data is modified. However, $H_{waybill}$ is unaltered because of blockchain features. Thus, the falsify is meaningless because tampered data is inaccessible owing to the difference of URIs.
- **IL:** We design an access control method in CA and store both I_{role} and $H_{waybill}$ in Ledger. Only Airports/Airlines and CA owners can access CA and acquire data (i.e., I_{role}, $H_{waybill}$, $D_{waybill}$).
- **MAS:** In AirCargoChain, all nodes run a Byzantine Fault Tolerance (BFT) protocol to agree on a concerted strategy. Therefore, AirCargoChain is able to verify the credibility of the received messages, since the number of malicious nodes is limited.
- **SPOF:** SPOF is a serious concern in traditional systems because both data storage and services are centralized. However, AirCargoChain is a blockchain-based distributed system. Every node stores a complete ledger, and each node can be a duplication and provide the same services if any other node fails.

6 Evaluation

This section deploys a 4-node blockchain network and performs experiments to verify the feasibility of AIRCARGOCHAIN. Our experiment environment is described in Table 3.

We first report experimental results and present discussions to answer the following questions.

1. **Scalability**: How well does AIRCARGOCHAIN store large-scale data?
2. **User-friendliness**: Why does AIRCARGOCHAIN conveniently work on real-world business?
3. **Effectiveness**: can AIRCARGOCHAIN improve efficiency for various processes among stakeholders in air cargo?

6.1 Q1 - Scalability

For the experiment, we compare the usage of disk space with original blockchain-based solutions by using a 98.8MB dataset. Original blockchain-based solutions store all data in a blockchain. The dataset includes 1,288 records where each record contains departure address, arrival address, timestamp, and figure of goods. Space consumption is shown in Fig. 2.

In original blockchain-based solutions, the disk space usage of each blockchain node consumes average 596.88MB, and 2387.52MB usage is consumed in total. In AIRCARGOCHAIN, IPFS storage nodes increase 146.36MB space usage, but only 36.79MB space usage is consumed in each blockchain node. To sum up, the total disk space usage of AIRCARGOCHAINis 293.52MB, and AIRCARGOCHAINdecreases storage resources consumption 0.1x of original blockchain-based solutions.

6.2 Q2 - User-Friendliness

In real-world business, stakeholders need communication to confirm the details of waybills. However, the process does not require the irrelated others to participate. AMOP is suitable for the process because it not only provides peer-to-peer communication, but also ensures packets are not acquired by other nodes. In addition, a contract address is usually a fixed-length hexadecimal data identifier, such as *b83261efa42895c38c6c2364ca878f43e77f3cddbc922bf57d0d48070f79feb6, which is too intricate to remember for users. CNS allows users to use a simple name instead of an*

Table 3 The Details of Implements

Parameters	Values
CPU	Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz × 40
Operating System	Ubuntu 18.04.1 LTS 64-bit
Memory	64GB
Software	Solidity: v0.4.24 Tomcat: v6.0.35 JDK: v1.8.0 Mysql: v5.7.11 Revised version of Ethereum: v2.4.0 IPFS: v0.4.23

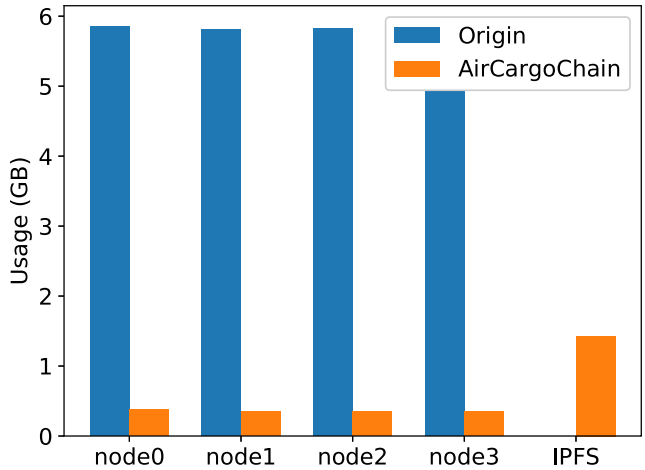

Fig. 2 Storage Resources Consumption: origin vs AIRCARGOCHAIN

intricate contract address, for example $user_1$. *Thus,* AIRCARGOCHAIN*is user-friendly. We conducted a survey for the real practice with 80 users, 96% of the returned questioner are likely to use CNS.*

6.3 Q3 - Effectiveness

We evaluate three main processes of AIRCARGOCHAINand the results are presented in Fig. 3.

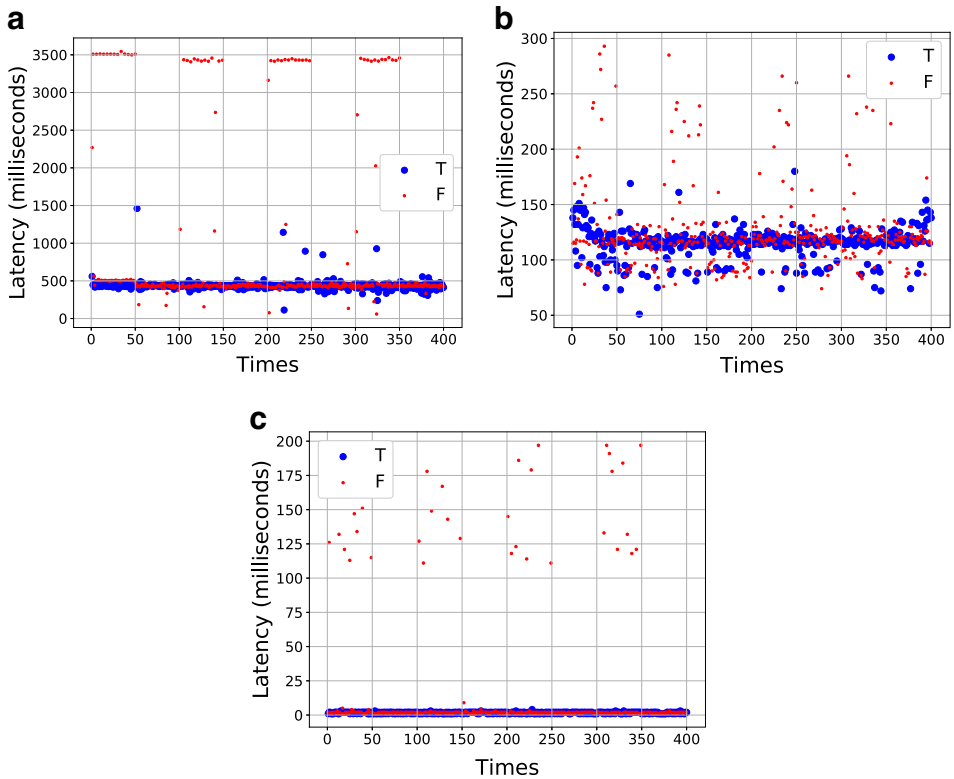

Fig. 3 Performance Evaluation. **a** Latency for Registration **b** Latency for Data Storage **c** Latency for Data Access

T and F denote two situations where all nodes are working and some node fails, respectively. We perform each process 400 times in each situation. In F, we first fail one node at the experiment beginning and recover the fault node at the 51st execution. Then we fail next node and recover the new fault node after the next 50st execution.

Most latency in F is similar to that in T. Figure 3a exhibits that a little latency is approximately 3471.21ms/tx in F. The average latency for registration is 437.37ms/tx and 831.97ms/tx in T and F, respectively. Figure 3b shows a little latency increases but all latency still less than 300ms/tx in F. The average latency for data storage is 115.15ms/tx and 126.31ms/tx in T and F, respectively. Figure 3c describes a little latency increases but no more than 200ms/tx in F. The average latency for data storage is 1.65ms/tx and 13.95ms/tx in T and F, respectively.

In summary, AIRCARGOCHAINis still working even if any node fails. Although there is a little increase of latency when some node fails, the increments do not disturb business processes and latency also recovers after failed nodes return to work. The millisecond AIRCARGOCHAINis more efficient than traditional schemes that are hours or days. Stakeholders can acquire data and execute various processes, such as freight checking, faster than traditional schemes. Thus, AIRCARGOCHAINhas no SPOF and improves efficiency for air cargo.

7 Conclusion

This paper proposes AIRCARGOCHAIN, which addresses the distrusted problem of data sharing for air cargo. The main idea is to build a cooperative network among stakeholders based on a blockchain, and make up for the deficiencies of the blockchain through an IPFS-based storage scheme and additional user-friendly design.

The advantages of AIRCARGOCHAINinclude

1. **Security**: security analyses in Section 5 proves that AIRCARGOCHAINis able to avoid multiple potential risks in air cargo.
2. **Scalability**: AIRCARGOCHAIN saves storage resources is presented in Section 6.1 with Figure 2.
3. **User-friendliness**: AIRCARGOCHAIN provides convenient services discussed in Section 6.2.

We also conduct evaluation of AIRCARGOCHAINin Section 6.3. The experimental results demonstrate the effectiveness and applicability of AIRCARGOCHAINfor air cargo.

Acknowledgements This work was partially supported by National Natural Science Foundation of China (No. 61902385), the Natural Science Foundation of Guangdong Province of China (No. 2019A1515011705, 2018A030313943), Shenzhen Basic Research Foundation (No. JCYJ20180302145645821, JCYJ20180302145633177).

References

1. Crosby, M., Pattanayak, P., Verma, S., Kalyanaraman, V., et al.: Blockchain technology: Beyond bitcoin. Appl. Innov. **2**(6-10), 71 (2016)
2. Dorri, A., Kanhere, S.S., Jurdak, R., Gauravaram, P.: Blockchain for iot security and privacy: The case study of a smart home. In: 2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops), pp. 618–623. IEEE (2017)
3. Nasrulin, B., Muzammal, M., Qu, Q.: Chainmob: Mobility analytics on blockchain. In: 2018 19th IEEE International Conference on Mobile Data Management (MDM), pp. 292–293. IEEE (2018)
4. Yue, X., Wang, H., Jin, D., Li, M., Jiang, W.: Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control. J. Med. Syst. **40**(10), 218 (2016)
5. Benet, J.: Ipfs-content addressed, versioned, p2p file system. arXiv:1407.3561 (2014)
6. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Technical Report, Manubot (2019)
7. Wood, G. et al.: Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper **151**(2014), 1–32 (2014)
8. Schwartz, D., Youngs, N., Britto, A., et al.: The ripple protocol consensus algorithm. Ripple Labs Inc White Paper **5**(8) (2014)
9. Brown, R.G., Carlyle, J., Grigg, I., Hearn, M.: Corda: an introduction. R3 CEV **1**, 15 (2016)
10. King, S.: Primecoin: Cryptocurrency with prime number proof-of-work. **1**(6) (2013)
11. Angraal, S., Krumholz, H.M., Schulz, W.L.: Blockchain technology: applications in health care. Circ. Cardiovasc. Quality Outcomes **10**(9), e003800 (2017)
12. Yumna, H., Khan, M.M., Ikram, M., Ilyas, S.: Use of blockchain in education: A systematic literature review. In: Asian Conference on Intelligent Information and Database Systems, pp. 191–202. Springer (2019)
13. Panarello, A., Tapas, N., Merlino, G., Longo, F., Puliafito, A.: Blockchain and iot integration: A systematic survey. Sensors **18**(8), 2575 (2018)
14. Noveck, B.S.: The single point of failure. In: Innovating government, pp. 77–99. Springer (2011)
15. Kiayias, A., Russell, A., David, B., Oliynykov, R.: Ouroboros: A provably secure proof-of-stake blockchain

protocol. Annual International Cryptology Conference, pp. 357–388. Springer (2017)
16. Castro, M., Liskov, B., et al.: Practical byzantine fault tolerance. In: OSDI, vol. 99, pp. 173–186 (1999)
17. Buterin, V. et al.: A next-generation smart contract and decentralized application platform. White Paper **3**(37) (2014)
18. Singh, M., Kim, S.: Blockchain based intelligent vehicle data sharing framework. arXiv:1708.09721 (2017)
19. Liang, X., Zhao, J., Shetty, S., Liu, J., Li, D.: Integrating blockchain for data sharing and collaboration in mobile healthcare applications. 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), pp. 1–5. IEEE (2017)
20. Fan, K., Wang, S., Ren, Y., Li, H., Yang, Y.: Medblock: Efficient and secure medical data sharing via blockchain. J. Med. Syst. **42**(8), 136 (2018)
21. Xia, Q., Sifah, E.B., Smahi, A., Amofa, S., Zhang, X.: Bbds: Blockchain-based data sharing for electronic medical records in cloud environments. Information **8**(2), 44 (2017)
22. Zhang, G., Li, T., Li, Y., Hui, P., Jin, D.: Blockchain-based data sharing system for ai-powered network operations. J. Commun. Inf. Netw. **3**(3), 1–8 (2018)
23. Azaria, A., Ekblaw, A., Vieira, T., Lippman, A.: Medrec: Using blockchain for medical data access and permission management. In: 2016 2nd International Conference on Open and Big Data (OBD), pp. 25–30. IEEE (2016X)
24. Wang, S., Zhang, Y., Zhang, Y.: A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. IEEE Access **6**, 38437–38450 (2018)
25. Xu, L., Chen, L., Gao, Z., Chang, Y., Iakovou, E., Shi, W.: Binding the physical and cyber worlds: A blockchain approach for cargo supply chain security enhancement. In: 2018 IEEE International Symposium on Technologies for Homeland Security (HST), pp. 1–5. IEEE (2018)
26. Wen, Q., Gao, Y., Chen, Z., Wu, D.: A blockchain-based data sharing scheme in the supply chain by iiot. In: 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), pp. 695–700 (2019)
27. Weber, I., Xu, X., Riveret, R., Governatori, G., Ponomarev, A., Mendling, J.: Untrusted business process monitoring and execution using blockchain. In: International Conference on Business Process Management, pp. 329–347. Springer (2016)
28. Qu, Q., Nurgaliev, I., Muzammal, M., Jensen, C.S., Fan, J.: On spatio-temporal blockchain query processing. Futur. Gener. Comput. Syst. **98**, 208–218 (2019)
29. Muzammal, M., Qu, Q., Nasrulin, B.: Renovating blockchain with distributed databases: An open source system. Fut. Gener. Comput. Syst. **90**, 105–117 (2019)
30. Nasrulin, B., Muzammal, M., Qu, Q.: A robust spatio-temporal verification protocol for blockchain. In: International Conference on Web Information Systems Engineering, pp. 52–67. Springer (2018)
31. Nurgaliev, I., Muzammal, M., Qu, Q.: Enabling blockchain for efficient spatio-temporal query processing. In: International Conference on Web Information Systems Engineering, pp. 36–51. Springer (2018)