Towards Secure Cross-Organizational Data Transfer: A Blockchain-Enabled Message Broker Approach

Gleb Slepenkov[0009−0001−9978−396X] and Vladimir Korkhov[0000−0003−2458−3194]

Saint Petersburg State University, 7-9 Universitetskaya Embankment, St Petersburg, Russia, 199034
gslepenkov@gmail.com, v.korkhov@spbu.ru

Abstract. Secure data transfer (SDT) is crucial in modern information societies, particularly for cross-organizational data exchange. This paper investigates the application of blockchain technology to create robust SDT systems, combining the benefits of using blockchain as both a data access interface and a direct data transfer tool. We propose a novel architecture featuring a message broker-like design that overcomes data size limitations present in existing blockchain-based SDT solutions. The architecture incorporates blockchain, internal data storage within organizations, and connectors facilitating interaction between these components. The SDT process relies on smart contracts for managing data access, transfer, and confirmation, while the actual data is stored off-chain. We advocate for a private, two-layered blockchain-based system, employing PBFT consensus within smaller clusters for efficient message transfer and RAFT consensus for system management and data replication. This layered approach enhances both fault tolerance and scalability capabilities in comparison to existing blockchain approaches and traditional message brokers. Furthermore, we explore the potential of integrating Artificial Intelligence (AI) for anomaly detection in smart contract execution and dynamic load balancing across blockchain nodes in order to optimize the security and performance characteristics of the system. We present a system architecture overview and describe key implementation details of proposed system, including potential AI-driven enhancements.

Keywords: Blockchain · Secure Data Transfer (SDT) · Data Security · Distributed Ledger Technology (DLT) · Message Broker · Data Access Control · Fault Tolerance · Artificial Intelligence (AI)

1 Introduction

Nowadays, secure data transfer (SDT) is essential for maintaining data confidentiality and integrity. SDT systems are critical infrastructure components across diverse sectors, including financial institutions, governmental agencies, and enterprises. Implementing robust SDT mechanisms becomes particularly challenging when facilitating cross-organizational data exchange or connecting geographically distributed departments within a single organization, i.e., scenarios which require the use of public networks. This fact significantly increases the risk of data loss, data corruption, or unauthorized access to data during the transfer process. Consequently, data integrity verification and counterparty authentication protocols are required. Furthermore, confirmation of data reception is often a critical requirement. The heterogeneity of data transfer technologies utilized by different counterparties further complicates the SDT process.

Blockchain technology has the potential to enhance existing SDT mechanisms. This technology, by design, provides core features essential for SDT, such as fault tolerance, ledger data immutability, and zero trust between blockchain nodes. Moreover, many blockchains are designed to operate in unreliable public networks with nodes running in different, heterogeneous clusters owned by separate stakeholders, which makes this technology especially suitable for cross-organizational data transfer. Furthermore, blockchain platforms often provide smart contracts, which allow implementing custom data transfer logic directly on the blockchain.

The reasons mentioned above have stimulated significant research interest, as evidenced by comprehensive surveys exploring blockchain applications for data sharing and exchange [1] and for specific domains like smart transport [2]. These surveys highlight the potential of blockchain to address the challenges of SDT in diverse and demanding environments.

This study explores existing blockchain application paradigms for the SDT process and proposes a novel approach that integrates their advantages. The result is a blockchain-based SDT system that maintains key features of established approaches while adopting a message broker-like architecture with relaxed data size limitations. Furthermore, we investigate the potential of integrating Artificial Intelligence (AI) to enhance the security and efficiency of this system through anomaly detection and dynamic load balancing. This article presents an architectural overview and describes key implementation details of the proposed system, including potential AI-driven enhancements.

The study is organized in the following way. Section 2 describes existing data transfer paradigms in detail. Section 3 represents the proposed system architecture. Section 4 describes implementation details of the proposed system. Section 5 explores the potential AI-driven enhancements to the system. Finally, Section 6 concludes this article and denotes further research directions, including future work on AI integration.

2 Related Work

The integration of blockchain technology into data transfer systems is implemented according to two primary paradigms: blockchain as a data access interface and blockchain as a direct data transfer tool. This section examines the existing literature, classifying studies according to these two distinct roles of blockchain in facilitating data exchange.

2.1 Blockchain as a Data Access Interface

This approach leverages blockchain as a secure and transparent mechanism for managing and controlling access to data that is typically stored off-chain. The blockchain records metadata about the data, access permissions, and transaction histories, thereby ensuring data integrity, auditability, and secure access control. The data itself is transferred using conventional methods.

Wang et al. introduced BBS, a big data sharing system utilizing blockchain for access control and data integrity management [3]. Similarly, Wang et al. proposed a blockchain-based model for sharing big data in the oil and gas sector, employing blockchain to enable secure data access and provenance tracking [4]. Yang et al. developed a sharing platform for wild bird data based on blockchain and IPFS, where blockchain governs access rights to data residing in IPFS [5]. A cross-organizational data sharing framework that uses blockchain probes to orchestrate and audit data access across various entities was presented by Jia et al. [6]. Gupta et al. explored the application of blockchain for securing data access within e-healthcare applications [7].

2.2 Blockchain as a Data Transfer Tool

This approach employs the blockchain directly to transfer data between participants. While data may occasionally be stored on-chain, it is more common for the blockchain to facilitate the transfer of data stored off-chain, often in conjunction with technologies such as peer-to-peer networks or distributed file systems.

Lin et al. proposed a multi-level blockchain architecture for secure data transfer in the Internet of Vehicles (IoV), directly leveraging blockchain for data exchange [8]. Peng et al. demonstrated the feasibility of building a peer-to-peer file storage and sharing system on a consortium blockchain, where the blockchain assists in discovering and securely transferring file chunks [9]. Priyadarshini et al. introduced a system for secured data transfer between fog nodes utilizing blockchain to enhance the security of the data transfer process itself [10].

2.3 Enabling Technologies and Security Considerations

Regardless of the chosen paradigm, several enabling technologies and security considerations are universally relevant. Kim et al. presented a hybrid decentralized PBFT blockchain framework for OpenStack message queues, aimed at improving fault tolerance and scalability, crucial aspects for reliable data transfer systems [11]. A publish-subscribe architecture to foster interoperability among different blockchain networks, thus facilitating data exchange across heterogeneous systems, was introduced by Ghaemi et al. [12]. Both solutions represent the architecture where blockchain acts as a message broker, data transfer process is organized in publish-subscribe way and data flow is separated into topics. Such an architecture is traditional for message brokers like Apache Kafka [13].

That's why it will be referenced as "message broker-like" architecture in the rest of the paper.

Bagga et al. and Xu et al. explored blockchain-based authentication and key agreement protocols for IoV, enhancing the security of data transfer in vehicular contexts [14, 15]. Bogdanov et al. focused on the consensus mechanism, exploring a combination of PBFT and Raft [16], while Ai et al. proposed a Proof-of-Transactions consensus protocol [17], both striving for efficient and scalable agreement in distributed ledgers.

2.4 Comparison of Approaches

The two approaches, blockchain as a data access interface and blockchain as a data transfer tool, offer distinct advantages and disadvantages depending on the specific use case and requirements.

Table 1. Comparison of Blockchain Approaches for Data Transfer

Feature	Blockchain as Data Access Interface	Blockchain as Data Transfer Tool
Data Storage	Primarily Off-Chain	Often Off-Chain, but metadata on-chain
Data Transfer	Traditional methods	Blockchain or Blockchain-assisted
Scalability	Higher, as data transfer is off-chain	Potentially Lower, dependent on blockchain throughput
Complexity	Lower	Higher
On-Chain Footprint	Smaller (Metadata only)	Larger (Metadata and potentially data chunks)
Suitability	Large datasets, access control focus	Smaller datasets, secure and direct transfer focus

As illustrated in Table 1, the choice between these two approaches hinges on factors such as data size, scalability requirements, security priorities, and the level of on-chain transparency desired. The data access interface approach is generally more suitable for scenarios involving large datasets and a primary focus on secure access control, while the data transfer tool approach is better suited for scenarios demanding secure and direct data transfer, even if it potentially introduces limitations in terms of scalability.

3 System Architecture Overview

3.1 System Components

This section describes the architecture of the proposed system. The architecture aims to combine two dominant blockchain integration paradigms for data transfer systems, as identified in Section 2. By integrating these approaches, our system seeks to achieve a message broker-like architecture that facilitates convenient data transfer while mitigating the data size limitations inherent in individual solutions.

Our architecture leverages the following trends observed in recent research:

1. **Blockchain as a message broker:** (e.g., [11], [12]). This approach utilizes the blockchain ledger for direct message transfer.
2. **Blockchain as an interface to off-chain data:** (e.g., [3], [6]). This paradigm employs the blockchain to manage access and pointers to data stored off-chain.

While each of these trends offers unique advantages, they also present limitations:

1. **Limitations of Blockchain as a Message Broker:** Relying on the blockchain ledger for message transfer significantly restricts system scalability and the size of transmissible data due to the inherent constraints of blockchain technology.
2. **Limitations of Blockchain as an Interface to Off-Chain Data:** Solutions implementing this approach often lack native notification mechanisms, making efficient message retrieval a challenge and requiring external polling or other complex notification strategies.

The proposed system addresses these individual limitations by combining the strengths of both paradigms. This integrated approach allows us to leverage the blockchain for both message brokering and off-chain data referencing, effectively relaxing the constraints on data size and enabling efficient notification mechanisms.

The proposed architecture assumes that the system should contain the following components:

- Blockchain as the core system component responsible for data transfer process orchestration. Blockchain nodes act as interfaces to data storages, while smart contracts implement all data transfer mechanisms like data transformation (e.g. message encryption / signing), data verification and message notifications.
- Data storages internally used by counterparties (organizations) to store the data to be sent.
- Connectors — some application used by SDT process counterparties in order to create a connection between their data storages and blockchain nodes owned by the organization (mainly inspired by blockchain probes described in [6]).

A sample of such an architecture for SDT between two organizations is presented in Figure 1.

3.2 SDT Process Scheme

The SDT process scheme is illustrated in Figure 2 and involves data transfer between two organizations (Org1 and Org2) using a blockchain-mediated system.

The process can be separated into following stages:

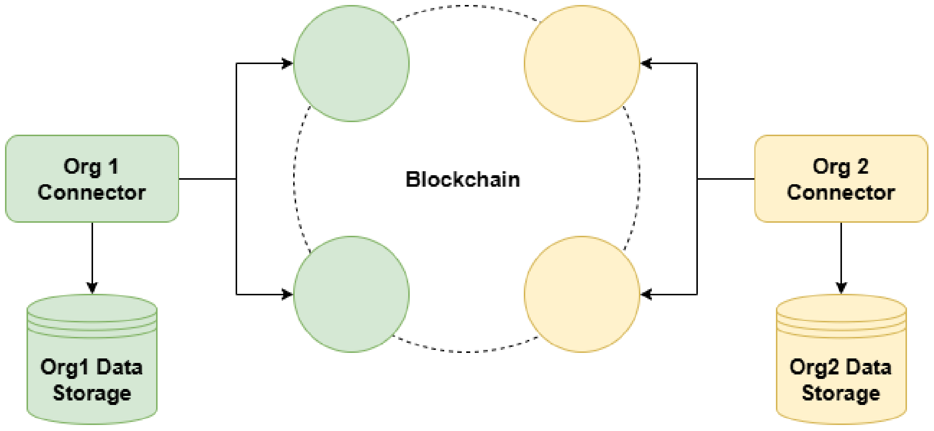

Fig. 1. System architecture for two organizations

Data Sending (Org1)

1. **Initiate Transfer (1):** The data sender (Org1) initiates a data transfer request to its connector.
2. **Data Persistence and Metadata Collection (2):** The Org1 connector persists the data in Org1's storage and gathers associated metadata.
3. **Smart Contract Invocation (3):** The Org1 connector queries the blockchain to initiate the execution of the SDT smart contract.
4. **Metadata Validation and Recording (4):** The smart contract validates the provided metadata and records it on the ledger.
5. **Notification (5-6):** Upon successful metadata validation, the smart contract notifies the Org2 connector about the availability of a new message and acknowledges message acceptance to Org1.

Data Retrieval (Org2)

1. **Data Request and Smart Contract Execution (7):** The Org2 connector queries the blockchain nodes and executes the SDT smart contract.
2. **Receiver Authentication (8):** The smart contract verifies Org2's permissions to the requested data.
3. **Data Retrieval from Org1 (9-12):** If authorized, the smart contract requests the data from the Org1 connector. Upon receiving the request, the Org1 connector retrieves the data from Org1's storage and transmits it back to the smart contract.
4. **Data Preparation (13):** The smart contract prepares the data for transfer, potentially involving encryption and digital signing.
5. **Data Delivery to Org2 (14-16):** The prepared data is returned to the Org2 connector, which then stores the received data in Org2's storage.

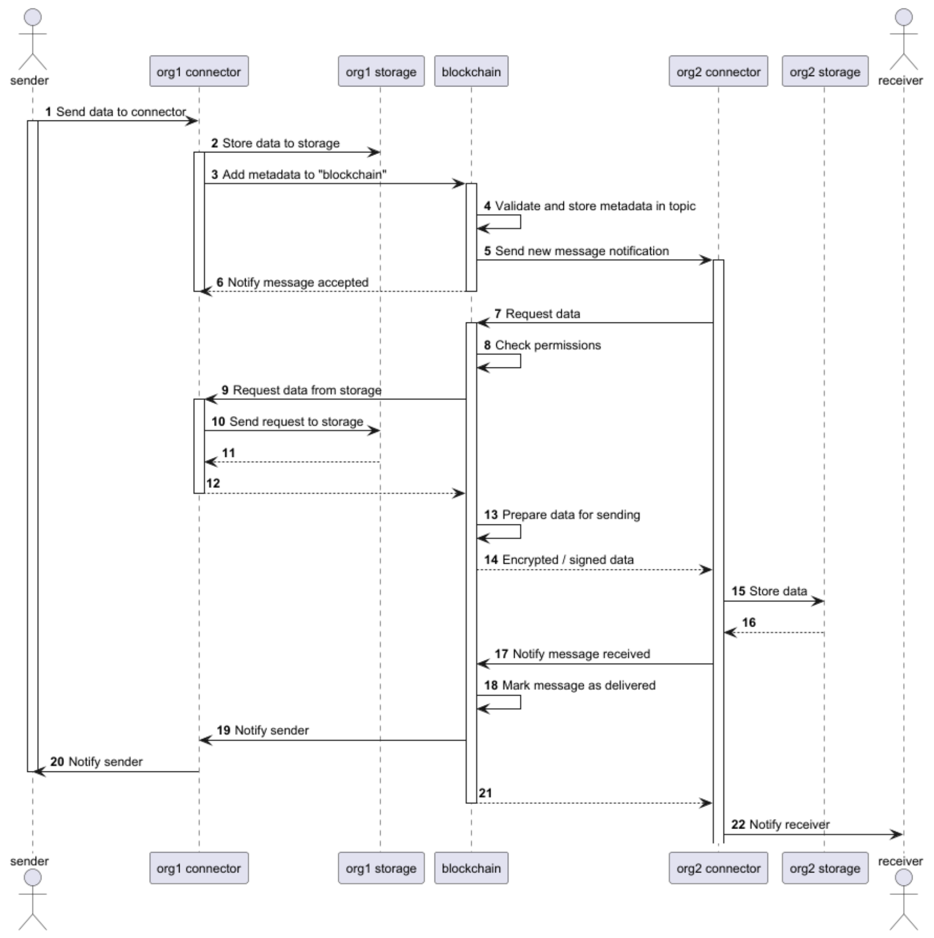
sender
org1 connector
org1 storage
blockchain
org2 connector
org2 storage
receiver
1 Send data to connector
2 Store data to storage
3 Add metadata to "blockchain"
4 Validate and store metadata in topic
5 Send new message notification
6 Notify message accepted
7 Request data
8 Check permissions
9 Request data from storage
10 Send request to storage
11
12
13 Prepare data for sending
14 Encrypted / signed data
15 Store data
16
17 Notify message received
18 Mark message as delivered
19 Notify sender
20 Notify sender
21
22 Notify receiver

Fig. 2. SDT process scheme

Process Finalization

1. **Message Reception Notification (17):** The Org2 connector notifies the blockchain about the message reception via the smart contract.
2. **Message Registry Update (18):** The smart contract updates the message registry, marking the message as delivered.
3. **Confirmation Notifications (19-22):** The smart contract notifies the Org1 and Org2 connectors about the successful completion of the message delivery process, and the connectors inform the data sender and data receiver about the completion of the data transfer process respectively.

Such an algorithm has the following important features:

1. Loose coupling between sender and receiver due to blockchain usage as a message broker.

2. Better fault tolerance compared to regular message brokers.
3. More convenient message broker-like SDT interface compared to existing blockchain solutions.
4. Greater SDT process customization capabilities due to smart contract usage.

4 Implementation details

4.1 Blockchain Selection

Blockchain technology constitutes the core component of the SDT system. Consequently, the specific blockchain implementation exerts a significant influence on key system parameters, including:

– **Throughput:** The volume of data the system can transfer within a defined time interval.
– **Fault Tolerance:** The types of faults the system can withstand and the maximum number of node failures the system can accommodate while maintaining functionality.
– **Scalability:** The maximum permissible number of nodes within the system.

The parameters delineated above are primarily governed by the blockchain type and the employed consensus algorithm.

A review of the existing literature reveals a prevalent trend favoring private blockchain solutions. This preference arises from the necessity for controlled access, enhanced privacy, and the potential for higher throughput in many SDT scenarios, particularly within applications such as IoT and vehicular networks. While several studies mention the potential use of public blockchains, they are not typically presented as the primary option. Specific examples include:

– Peng et al. [9] propose a peer-to-peer file storage and sharing system based on a consortium blockchain.
– Jia et al. [6] present a cross-organizational data sharing framework based on blockchain-probes, indicative of a permissioned setting.
– Numerous papers addressing IoT applications (e.g., [7, 17]) often implicitly or explicitly assume a permissioned blockchain context to address security and scalability constraints.

Therefore, this paper posits that a private blockchain is the most suitable option for SDT system implementation.

In addition to selecting the appropriate blockchain type, employing the correct consensus algorithm is crucial. To enhance the SDT system's fault tolerance, this paper proposes the utilization of the Practical Byzantine Fault Tolerance (PBFT) algorithm, aligning with observed trends in the literature. However, PBFT is known to have inherent performance limitations.

Firstly, PBFT implementations necessitate the transfer of substantial data volumes between blockchain nodes to achieve consensus. For instance, the BFT-SMaRT algorithm, employed by Hyperledger Fabric [18], requires each node to transmit 80 MB of data to achieve a performance of 2500 transactions per second. This implies a total traffic volume of 720 MB within a cluster of only 10 nodes during all consensus algorithm stages.

Furthermore, the throughput of such a consensus algorithm is contingent upon the number of nodes within the system. Research [18] indicates that BFT-SMaRT achieves maximum throughput when the system comprises 7 nodes (Fig. 3).

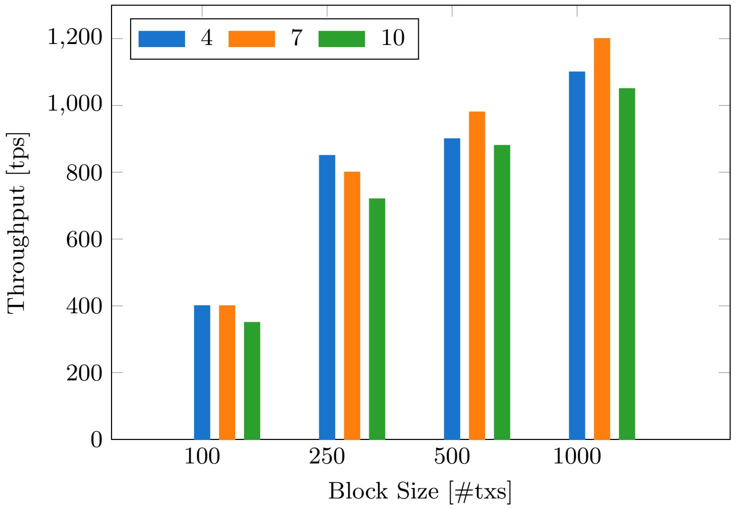

Fig. 3. BFT-SMaRT performance in global networks (measured by Barger et. al. [18])

Increasing the number of nodes beyond this optimum leads to a degradation in throughput. The maximum permissible number of nodes, according to research [19], is limited to approximately 100.

For the reasons articulated above, a pure PBFT blockchain is not ideally suited for the creation of scalable and fault-tolerant systems. Fortunately, this limitation can be addressed by leveraging the multi-layered blockchain architecture described in [8] and [16], coupled with the data flow separation into topics and partitions, a common practice in message brokers such as Apache Kafka [13].

This paper proposes the use of a two-layered blockchain architecture. The first layer employs the PBFT consensus algorithm. Given the algorithm's known limitations, this layer will be constructed from multiple, relatively small clusters, each consisting of up to 7 nodes. This layer embodies the concept of a partition, representing the smallest system component directly responsible for the execution of SDT smart contracts.

The subsequent layer leverages the RAFT consensus algorithm [20]. This layer is designed for two primary functions: system management (detailed in

Section 4.2) and asynchronous data replication following PBFT consensus rounds (described in Section 4.3). Consequently, this layer is analogous to the topic concept found in message brokers.

This architecture allows for the preservation of all fault-tolerance characteristics inherent in the PBFT consensus algorithm while simultaneously enhancing system scalability. Scaling the system does not necessitate an increase in the PBFT cluster size. Instead, a new, smaller PBFT layer-1 cluster is created and connected to the existing RAFT layer-2 cluster, which has a less stringent limit on the maximum number of nodes.

4.2 Data Flow Separation

This section delineates the proposed data flow separation scheme. As previously mentioned, the data flow of the SDT system is partitioned into topics and partitions.

A topic is defined as a data stream unified by a common theme or a list of consumers. To address the limitations outlined earlier, each topic is further subdivided into a list of partitions. An illustrative example of such a data flow separation into two partitions is depicted in Fig. 4.

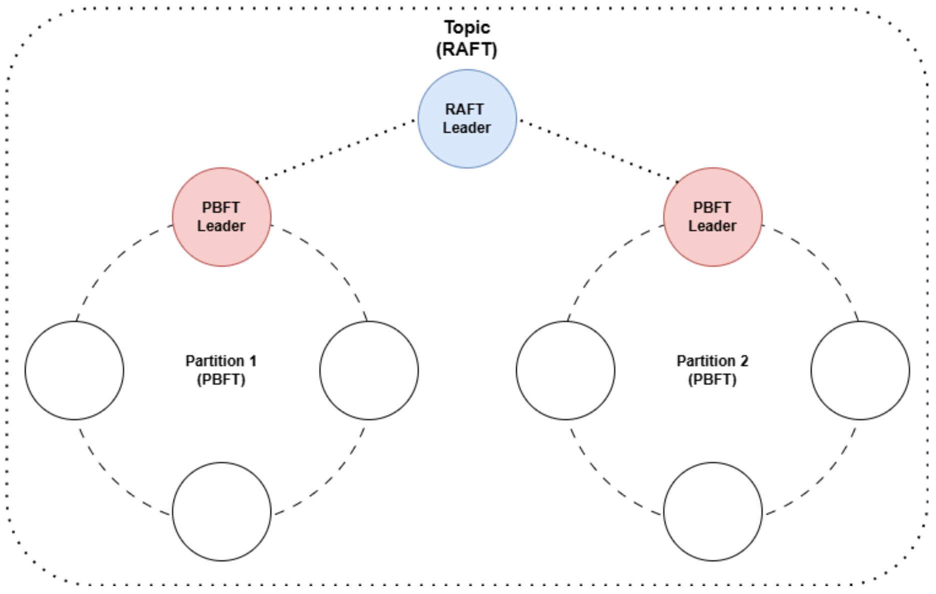

Fig. 4. Structure of a topic with two partitions

Two primary challenges must be addressed to implement this data flow separation:

- How to select a specific blockchain node for SDT process initialization while ensuring effective load balancing?
- How to select the appropriate partition for a specific message?

The selection of a specific load balancing algorithm constitutes a separate area of research and, therefore, falls outside the scope of this article.

Concerning the second challenge, this paper proposes the utilization of hash partitioning, a technique employed by Apache Kafka, for partition selection. The partition is determined by calculating the remainder of dividing the key's hash value by the total number of partitions.

To store the topic configuration data, the layer-2 ledger of the blockchain is proposed. Each node within this blockchain layer stores a list of partitions and a corresponding list of nodes for each partition. Any structural modifications within a topic, such as the creation of a new partition, are to be executed via a dedicated smart contract.

A key advantage of this approach lies in the replication of configuration data across all blockchain nodes. Consequently, data senders can retrieve this data from any topic node. This eliminates the need for dedicated control nodes, such as Apache Zookeeper nodes in Apache Kafka, thereby enhancing system fault tolerance by mitigating the risk of a single point of failure.

4.3 Data Replication

As previously discussed, a primary advantage of utilizing blockchain technology in a data transfer system is its ability to provide verifiable evidence of message delivery by immutably recording the data transfer history within its ledger. Given the system's separation into multiple partitions, verifying message delivery necessitates identifying the specific partition used for the message's transfer. This task is complicated by the fact that the system configuration may evolve over time, potentially requiring an iterative search across all existing partitions. To streamline the message delivery verification process, data replication to all topic nodes is essential.

To facilitate this data replication mechanism, this article introduces the concepts of global and local ledgers. The global ledger, maintained at the topic level, essentially represents the union of all partition-specific ledgers, which we term local ledgers. The global ledger is updated asynchronously with data from the local ledgers through a dedicated periodic task. This task implements the following algorithm:

1. Retrieve the list of partitions from the topic configuration ledger.
2. For each partition:
   (a) Identify the key of the last replicated block.
   (b) Query the local partition ledger to ascertain the presence of new data.
   (c) If new data is found, copy all new data blocks to the global ledger.

An illustrative example of this algorithm's execution for two partitions is presented in Fig. 5. In the figure, the last replicated blocks are highlighted in yellow, while the blocks slated for replication are depicted in green. Prior to the algorithm's execution, there are three blockchain nodes awaiting replication: one from Partition 1 and two from Partition 2. Following the replication process, these blocks are appended to their corresponding blockchains within the global ledger.

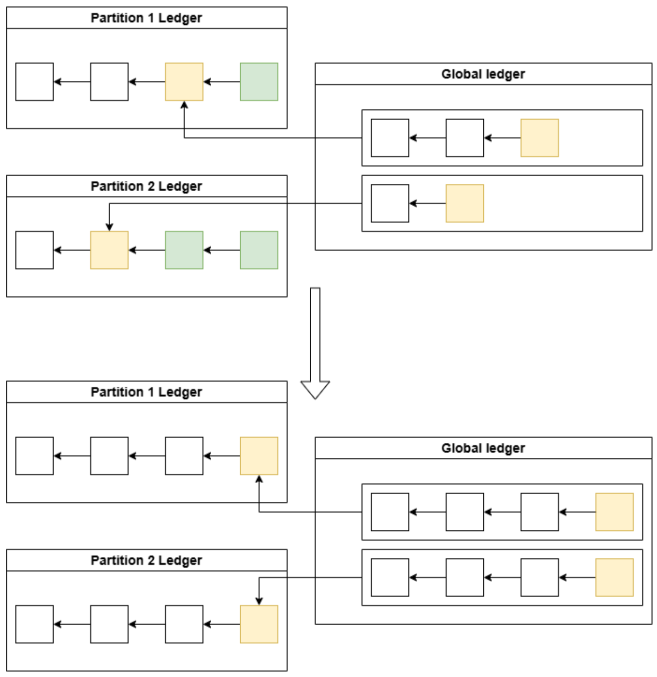

Fig. 5. Data replication to global ledger

Given that the algorithm operates on the second blockchain layer, all modifications to the global ledger are, again, governed by the RAFT consensus algorithm.

5 Potential AI-Driven Enhancements

Building upon the architecture and implementation details described in previous sections, the integration of Artificial Intelligence (AI) offers substantial opportunities to further optimize and enhance the SDT system. Specifically, AI can be instrumental in improving the efficiency and security of smart contract execution, as well as in dynamically managing the workload across the blockchain nodes.

5.1 AI-Powered Anomaly Detection

As outlined in Section 3, smart contracts play a central role in managing data access, transfer, and message delivery confirmation within the SDT process. However, vulnerabilities in these contracts can compromise the entire system. Drawing inspiration from [21], AI, and particularly deep learning techniques, can be employed to create a real-time anomaly detection system tailored to the nuances of the SDT process.

This AI system would continuously monitor the execution of SDT smart contracts, learning the expected behavior patterns for each stage of the data transfer process (as depicted in Figure 2). Deviations from these established patterns, such as:

- **Unvalidated Metadata Changes:** Detecting attempts to alter metadata recorded on the ledger without proper authorization.
- **Unexpected Data Transformation:** Identifying unauthorized or incorrect data encryption/signing operations.
- **Abnormal Access Patterns:** Recognizing unauthorized data retrieval attempts or suspicious data requests from connector applications.

These deviations would be flagged as potential anomalies, triggering alerts and potentially halting the SDT process until the issue is investigated. This proactive approach significantly strengthens the security of the SDT system by mitigating risks associated with malicious or compromised smart contracts.

5.2 AI-Driven Dynamic Load Balancing

As described in Section 4, the two-layered blockchain architecture, coupled with data flow separation into topics and partitions, is designed to enhance the scalability and throughput of the SDT system. However, the efficient allocation of smart contract execution requests across the cluster nodes is crucial to maximizing performance.

Inspired by the work of Singh et al. [22] and Tsang et al. [23], we propose integrating an AI-powered dynamic load balancing mechanism that considers a multitude of factors to intelligently route SDT smart contract execution requests:

- **Node Availability and Capacity:** Monitoring the real-time availability and computational capacity of each node within the PBFT clusters in layer 1.
- **Network Latency:** Assessing the network latency between the requesting connectors and the potential execution nodes.
- **Current Workload:** Evaluating the current workload of each node to avoid overloading.
- **Data Localization:** Prioritizing the selection of nodes that have already replicated relevant metadata from the topic, as described in Section 4.3, minimizing data transfer overhead.

The AI algorithm, potentially employing reinforcement learning techniques, would continuously learn from past performance data to optimize the routing decisions, ensuring that SDT smart contract execution requests are directed to the most suitable node based on the prevailing conditions. This dynamic load balancing would lead to improved network throughput, reduced latency, and enhanced overall system efficiency.

6 Conclusions and Future Work

Blockchain technology is, by design, a promising solution for enhancing the security and performance characteristics of SDT systems. However, existing blockchain application paradigms have either strict transferred data size limitations or interfaces inconvenient for SDT process implementation. That's why we propose a novel approach, which is basically a combination of existing paradigms enhanced by a message broker-like architecture. Such an approach allows us to combine the strengths of existing solutions in order to achieve better security and performance characteristics, combined with familiar data transfer interfaces and data flow separation into topics and partitions. The core component of the proposed system is a two-layered blockchain. The first blockchain layer utilizes small PBFT clusters (named partitions) directly for data transfer. These small clusters are then combined into a second layer blockchain with a RAFT consensus algorithm (named topics), utilized for system management and asynchronous data replication across topic nodes. Such a two-layered architecture allows us to overcome known PBFT cluster size limitations while preserving fault tolerance capabilities. For further system improvements, we explored the potential integration of Artificial Intelligence (AI) for anomaly detection and dynamic load balancing. To summarize, the proposed system architecture provides loose coupling between sender and receiver, improved fault tolerance compared to traditional message brokers, a more convenient message broker-like SDT interface, greater SDT process customization capabilities through smart contract usage, and the prospect of AI-driven enhancements, which makes it a practical and secure solution for cross-organizational data sharing.

Future research will focus on several directions. The main research direction is the implementation and evaluation of the proposed system's functional prototype in order to measure its performance characteristics. The list of additional research directions includes the selection of specific, optimized consensus algorithm implementations, the development of a dynamic load balancing mechanism for smart contract execution node selection, and the integration of AI-driven SDT process anomaly detection.

Acknowledgments. Supported by Saint Petersburg State University, project ID: 95438429.

References

1. Song, R., et al.: A survey of blockchain-based schemes for data sharing and exchange. IEEE Transactions on Big Data (2023)
2. Bagga, P., Das, A.K.: Blockchain for Smart Transport Applications, pp. 125–154. Springer International Publishing, Cham (2022)
3. Wang, S., et al.: Bbs: A secure and autonomous blockchain-based big-data sharing system. Journal of Systems Architecture **150**, 103133 (2024)
4. Wang, Y.Y., Huang, S., Yu, X.: An oil and gas big data sharing model based on blockchain technology. IOP Conference Series: Earth and Environmental Science **651**(3), 032105 (2021)
5. Yang, H., et al.: A research on the sharing platform of wild bird data in yunnan province based on blockchain and interstellar file system. Sensors **22**(18), 6961 (2022)
6. Jia, X., et al.: Cross-organisational data sharing framework based on blockchain-probes. IET Networks **12**(2), 77–85 (2023)
7. Gupta, S., Yadav, B., Gupta, B.: Security of IoT-based e-healthcare applications using blockchain, pp. 79–107. Springer International Publishing, Cham (2022)
8. Lin, H.Y.: Secure data transfer based on a multi-level blockchain for internet of vehicles. Sensors **23**(5), 2664 (2023)
9. Peng, S., et al.: A peer-to-peer file storage and sharing system based on consortium blockchain. Future Generation Computer Systems **141**, 197–204 (2023)
10. Priyadarshini, R., Malarvizhi, N.: Secured data transfer between fog nodes using blockchain. In: Proceedings of the 2nd International Conference on Computational and Bio Engineering: CBE 2020. pp. 417–422. Springer Singapore (2021)
11. Kim, Y., Park, J.: Hybrid decentralized pbft blockchain framework for openstack message queue. Human-centric Computing and Information Sciences **10**(1), 31 (2020)
12. Ghaemi, S., et al.: A pub-sub architecture to promote blockchain interoperability. arXiv preprint arXiv:2101.12331 (2021)
13. Foundation, A.S.: Apache kafka documentation. https://kafka.apache.org/documentation/ (nd), accessed: 2025-03-09
14. Bagga, P., et al.: Blockchain-based batch authentication protocol for internet of vehicles. Journal of Systems Architecture **113**, 101877 (2021)
15. Xu, Z., et al.: A blockchain-based roadside unit-assisted authentication and key agreement protocol for internet of vehicles. Journal of Parallel and Distributed Computing **149**, 29–39 (2021)
16. Bogdanov, A., et al.: Combining pbft and raft for scalable and fault-tolerant distributed consensus. Physics of Particles and Nuclei **55**(3), 418–420 (2024)
17. Ai, Z., Cui, W.: A proof-of-transactions blockchain consensus protocol for large-scale iot. IEEE Internet of Things Journal (2022). https://doi.org/10.1109/JIOT.2021.3108621
18. Barger, A., et al.: A byzantine fault-tolerant consensus library for hyperledger fabric. In: 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). pp. 1–9. IEEE (2021)
19. Ke, Z., Park, N.: Performance modeling and analysis of hyperledger fabric. Cluster Computing **26**(5), 2681–2699 (2023)
20. Ongaro, D., Ousterhout, J.: The raft consensus algorithm. Lecture Notes CS **190**, 2022 (2015)

21. Demertzis, K., Iliadis, L., Tziritas, N., Kikiras, P.: Anomaly detection via blockchained deep learning smart contracts in industry 4.0. Neural Computing and Applications **32**(23), 17361–17378 (2020)
22. Singh, A.R., Kumar, R.S., Madhavi, K.R., Alsaif, F., Bajaj, M., Zaitsev, I.: Optimizing demand response and load balancing in smart ev charging networks using ai integrated blockchain framework. Scientific Reports **14**(1), 31768 (2024)
23. Tsang, Y.P., Lee, C.K.M., Zhang, K., Wu, C.H., Ip, W.: On-chain and off-chain data management for blockchain-internet of things: a multi-agent deep reinforcement learning approach. Journal of Grid Computing **22**(1), 16 (2024)