Heliyon 9 (2023) e13186

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Research article

Building operation and maintenance scheme based on sharding blockchain

Jinlong Wang [a,*], Xu Wang [a,b], Yumin Shen [a,c], Xiaoyun Xiong [a], Wenhu Zheng [a,b], Peng Li [b], Xiaoxue Fang [b]

[a] School of Information and Control Engineering, Qingdao University of Technology, Qingdao 266525, China
[b] Qingdao E-Link Information Technology Co.,Ltd., Qingdao, 266033, China
[c] Hangzhou Ancun Internet Technology Co.,Ltd., Hangzhou 311100, China

ARTICLE INFO

Keywords:
Building operation and maintenance (BOM)
Blockchain expansion
Sharding blockchain
Transaction handling strategy

ABSTRACT

Blockchain can ensure data security and reliability during the stage of building operation and maintenance (BOM), provide reliable data for decision-making. However, existing schemes based on single-chain architecture have the problems of storage limitation and scalability, and ignore the impact of event's priority and real-time on blockchain transaction. Therefore, for BOM, this paper provides a BOM framework based on sharding blockchain (SBC-BOMF), which constructs two-layer architecture based on master-chain and multiple shards, relieves the storage pressure of blockchain nodes and improves the concurrency capability. Priority-based transaction handling strategy is designed to achieve reasonable and rapid response for multi-level transactions. Finally, an actual BOM project is taken as example to illustrate the effectiveness of proposed scheme; experiments are conducted for performance testing and evaluation. Results show that proposed scheme can effectively solve the scalability problem caused by the application of blockchain in BOM, reduce storage overhead, and realize efficient handling for blockchain transactions.

1. Introduction

The lifecycle of building can generally be divided into four stages: planning and design, construction, operation and maintenance (O&M), and demolition. Among the above stages, O&M is the longest stage in the whole lifecycle of building, accounting for the vast majority of the total cost [1]. Efficient management work in the stage of building operation and maintenance (BOM) is necessary, as management methods and work efficiency play a crucial role in saving costs [2,3].

The emergence of the Internet of Things (IoT) technology and its rapid development in the field of construction has promoted the transformation of O&M mode from manual to automated and intelligent, effectively improving the efficiency and quality of BOM, and reducing O&M costs [4–6]. Literature [4] developed an IoT-based BOM system, using IoT technology to realize the interconnection and efficient collaboration between sensors in the building, and improve the efficiency of O&M. Literatures [5,6] used data generated in BOM to build an energy model to predict and reduce energy consumption. However, since the IoT-based BOM system is often constructed in the centralized architecture (i.e., store O&M-related data through a central server), it faces the single point of failure [7]. In addition, trust and security issues make the authenticity of O&M data unable to be guaranteed [8]. The false data may lead O&M personnel to make incorrect decisions, adversely affecting the efficiency of O&M and the effectiveness of decisions [9].

* Corresponding author.
E-mail address: qdwangjinlong@163.com (J. Wang).

https://doi.org/10.1016/j.heliyon.2023.e13186
Received 16 September 2022; Received in revised form 13 January 2023; Accepted 19 January 2023
Available online 26 January 2023

Blockchain [10] is a distributed ledger technology which can solve the single-point failure, trust and security problems in BOM. Some works stored BOM data in the blockchain to avoid third-party trust problems in the process of using data and prevent data from being tampered with [11,12]. Literatures [13,14] designed BOM system based on blockchain to avoid single-point failure, and used smart contracts to monitor the running status of IoT devices. The above studies effectively solve the problems of trust and single-point failure in the stage of BOM [15]. However, due to the scalability bottleneck and full-copy storage of blockchain [16], these schemes will encounter challenges in concurrency and storage capacity [17] when applied to BOM system. Moreover, different O&M events have different priorities and real-time requirement, which also brings new challenges to the strategy design of blockchain transactions, but the existing schemes have not taken this into account.

As the mainstream method of blockchain expansion, sharding technology can offload the workload of handling transactions and storing data to different shards [18,19], thereby reducing the storage pressure of nodes and improving the throughput of blockchain system. However, it is rare to apply sharding technology to BOM scenario.

For BOM, this paper designs a BOM scheme based on sharding blockchain, which includes:

- Proposing a BOM framework based on sharding blockchain (SBC-BOMF), and further constructing the two-layer blockchain architecture of "master-chain & multiple shards" in proposed SBC-BOMF. O&M data are stored in different shards to reduce the storage overhead of nodes; each shard processes O&M-related intra-shard transactions in parallel, while cross-shard transactions are recorded and forwarded by master-chain. Since transactions in different shards can be processed in parallel, the throughput performance of system will improve with the expansion of blockchain network, thereby solving the scalability bottleneck of blockchain.
- Designing priority-based handling strategy, which can dynamically adjust the handling order of transactions based on priority and urgency, to achieve reasonable and rapid response for multi-level transactions.

The rest of the paper is organized as follows. Related literatures are reviewed in Section 2. In Section 3, design goals of proposed scheme are described. Section 4 is to describe the scheme in detail. Section 5 is to illustrate the effectiveness of proposed scheme by an actual project, and evaluate the performance of the scheme. Conclusions and future works are drawn in Section 6.

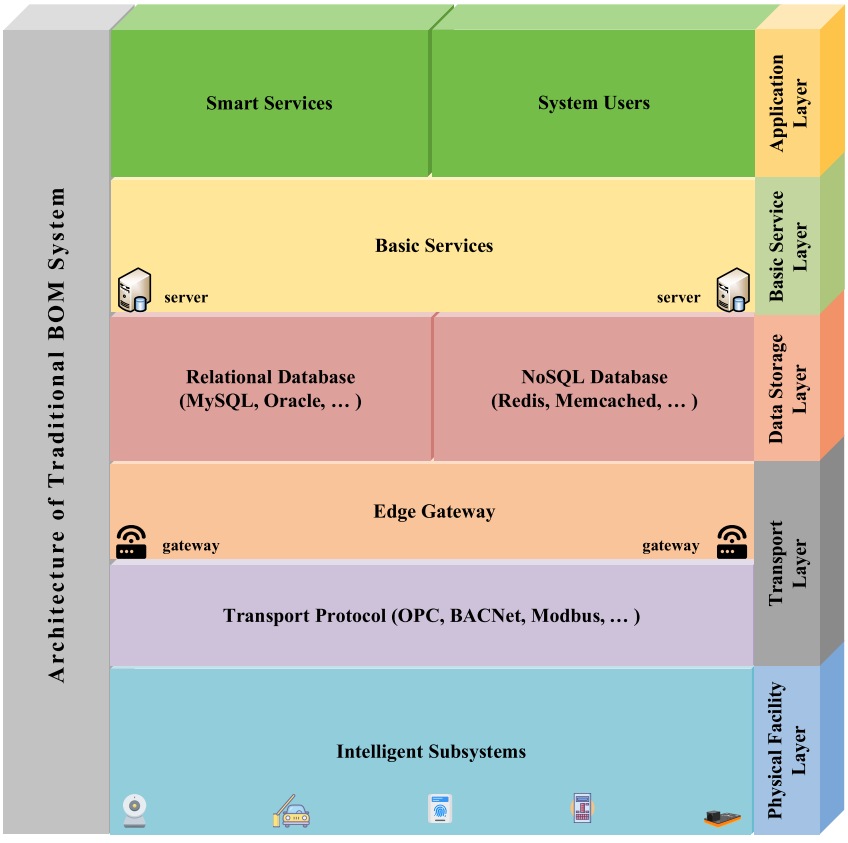


Fig. 1. BOM model based on IoT.

2. Related works

2.1. Intelligent building operation and maintenance

Efficient BOM is of great significance to reduce costs [1]. El-Ammari et al. [20] studied a facilities management (FM) model based on IFC standard, which improved the efficiency of FM by applying the data generated in the stages of design and construction to FM. Zhang et al. [21] proposed a building management optimization system to maximize the service life of building and improve O&M efficiency. These early works mainly adopt the inefficient manual management mode, which are difficult to effectively reduce the cost in BOM.

The application of IoT technology in the field of construction has promoted the transformation of O&M mode from manual to automated and intelligent [22]. Traditional IoT-based BOM model can be expressed as a five-layer architecture as shown in Fig. 1, including physical facility layer (PFL), transport layer (TL), data storage layer (DSL), basic service layer (BSL) and application layer (AL). (1) PFL is composed of IoT devices that constitute various intelligent subsystems (fire alarm, video surveillance, access control, etc.), and it is the main data source of BOM system. (2) TL provides a series of communication protocols for edge gateways to gather real-time data from IoT devices. (3) DSL utilizes relational databases and non-relational databases to store IoT data generated by PFL, O&M-related data after secondary processing by BSL, and application data generated by AL. (4) BSL builds knowledge models by processing the data from DSL layer. (5) AL binds the data (in DSL) with the basic services (in BSL), and provides the smart services (visualization, energy consumption analysis, fault diagnosis of devices, etc.). Based on the monitoring data generated by IoT devices, Bottaccioli et al. [5] designed a building energy model to simulate, predict, and reduce unnecessary energy consumption. Tragos et al. [23] built an IoT-based BOM system to reduce O&M risks and costs for enterprises.

Through the organic combination of IoT and building information modeling (BIM), the whole process of visual BOM management from data perception, acquisition, integration, analysis to decision-making can be realized, so as to improve O&M efficiency and reduce O&M costs [24–26]. Oti et al. [24] integrated BIM into BOM system to reduce management difficulty of building facilities and improve FM efficiency. Chen et al. [25] applied BIM and IoT technology to the inspection and maintenance of fire safety equipment to improve O&M efficiency and indirectly reduce fire losses. Based on BIM and IoT real-time data, Wang et al. [26] designed a dynamic fire escape path planning method to provide evacuation path planning in fire scenarios and meet firefighting needs of complex buildings.

Applying IoT technology to BOM can effectively improve O&M efficiency and reduce O&M costs. However, in traditional IoT-based BOM platform, DSL and BSL that serve the AL usually rely on the centralized cloud server, which may cause single-point failure and data security issues. In addition, the problems of third-party trust and reliability exist in BOM, which bring great challenge to the effectiveness of O&M decisions [8].

2.2. Blockchain in building operation and maintenance

Blockchain is a trustless and decentralized distributed ledger technology jointly maintained by multiple nodes [27]. Once the data is verified by nodes and uploaded to blockchain, it will be permanently stored and cannot be tampered with. Blockchain achieves partition tolerance for traditional applications through decentralized storage, enables data to be temper-proof and traceable based on cryptography and chain structure, thereby enhancing data security from the perspective of confidentiality, integrity and availability. The emergence of blockchain technology provides new means for solving the problems of data security and third-party trust faced by existing fields [28]. Once the concept of blockchain is proposed, it has attracted the attention of experts and researchers from all walks of life, and has been applied to finance [29], Industry 4.0 [30], electronic medical records [31] and other fields.

Similarly, in construction, especially in the stage of BOM, blockchain is used to replace the traditional database within DSL to ensure system security and data reliability. Xu et al. [12] stored devices data generated from BOM system in the blockchain, avoided third-party trust problems in the process of using data, and ensured data safety and reliability. Van et al. [13] designed a blockchain-based decentralized architecture which used smart contracts to achieve building energy management and ensure data reliability. Agarwal et al. [14] proposed a blockchain-based BOM framework for safety management and FM. to achieve efficient O&M. Siountri et al. [32] proposed a BOM system integrating BIM, IoT and blockchain. The framework (1) realized the storage and sharing of O&M data through cloud-edge collaboration; (2) used blockchain to ensure data security and prevent data from being tampered with.

With the popularity of IoT applications, the number of IoT devices deployed in building continues to increase, resulting in an increasing amount of IoT data [33]. Application of the BOM scheme based on single-chain architecture will bring great storage challenges [17]. In addition, the increase in the number of IoT devices will generate more blockchain transactions related to data storage and O&M, which puts forward higher requirement for the throughput performance of blockchain.

2.3. Sharding blockchain

Sharding blockchain [34] divides the entire nodes into several relatively independent shards. Each shard only handles transaction and store data related to themselves, and can be combined with other shards to form a complete global ledger. Since nodes in different shards can handle disjoint transactions in parallel, the overall throughput of blockchain network can increase approximately linearly with the expansion of network scale. Fragmented storage of global ledger can also effectively relieve the storage pressure of blockchain nodes [35].

As one of the mainstream methods for on-chain scaling [36], sharding technology has attracted extensive attention of scholars.

Considering that the throughput of mainstream blockchain systems (e.g., Ethereum, Hyperledger Fabric) cannot meet the needs of large-scale data storage in multi-domain IoT systems, Tong et al. [18] divided blockchain nodes into different shards according to the relationship between nodes and domains. Each shard handled transactions related to its domain in parallel, thereby improving overall throughput of blockchain system. Wang et al. [19] proposed a distributed secure storage scheme based on sharding blockchain, which randomly and evenly divides the nodes into shards. On this basis, they designed secret sharing scheme without trusted third party, effectively reducing the overhead of storage and communication.

The above researches provide the technical idea for solving scalability and storage limitations in traditional blockchain-based BOM applications.

3. Problem formulation

In order to determine the goals of blockchain-based BOM scheme and carry out the design, this section analyzes the special requirements of BOM scenario for deploying blockchain.

The deployment of blockchain environment has requirements for storage and computing capabilities of nodes: for storage capability, the hard disk space reserved for running blockchain is different according to the network architecture and storage mode, which is not be discussed here; for computing capability, taking Hyperledger Fabric as an example, running peer node requires an additional 300 MB of memory [37].

In BOM, installations that can deploy the blockchain environment mainly include the following three categories:

1) Gateway (GW): placed inside the building and connected to IoT devices in a certain subsystem to realize data collection, preprocessing, forwarding and control.
2) Subsystem server (SS): usually placed inside the building (computer room), equipped with the management software provided by device manufacturer to assist O&M personnel to manage subsystem devices. According to the scale of building and devices, software of multiple subsystems may be deployed in the same server.
3) Cloud server (CS): various servers deployed in the cloud to provide data backup, integrated smart O&M or other services.

Existing schemes usually deploy the above installations as blockchain nodes in the single chain. However, due to the full-copy storage characteristics of blockchain, as well as the dynamic growth of transactions (caused by the expansion of devices scale), these schemes face the performance issues of concurrency and storage.

Furthermore, O&M events generated in BOM can be divided into different priorities [38] (e.g., urgent, important, ordinary), for example, the priority and real-time requirements of fire alarm event are significantly higher than ordinary device failure. Different O&M events with the same priority also have different real-time requirements for response (most events are controlled within 2–10 s). The response to an event may involve the consensus of relevant blockchain transaction. Therefore, it is necessary to formulate a handling strategy for blockchain transactions according to the priority and real-time of O&M events.

In summary, design goals of blockchain-based BOM scheme can be summarized as the following three points:

Goal 1: Trusted storage. Blockchain is used to enhance data confidentiality, integrity and availability for BOM applications. This is the most basic goal to be achieved when designing blockchain-based scheme.

Goal 2: Low storage overhead and high concurrency capability. The scheme can alleviate storage pressure of blockchain nodes and improve throughput performance of blockchain system.

Goal 3: Efficient capability for handling transaction. The scheme needs to provide a transaction handling strategy to achieve reasonable and rapid response for multi-level blockchain transactions.

4. Proposed scheme

Combined with the previous description of blockchain-based application requirements and the definition of design goals, this paper proposes a BOM scheme based on sharding blockchain. The specific details of proposed scheme can be summarized as follows:

1) Designing BOM framework based on sharding blockchain (SBC-BOMF) to achieve the trusted storage (**Goal 1**). Sharding technology is used to disperse the storage pressure of blockchain nodes and improve the throughput of blockchain system (**Goal 2**). SBC-BOMF is formed as:

$$\text{SBC-BOMF} = \left\{S_i - \text{MC} - S_j\right\}_{\text{PFL-TL-DSL-BSL-AL}}$$

where MC represents master-chain, S_i and S_j represent different shards in SBC-BOMF; $\{S_i - \text{MC} - S_j\}$ refers to achieve the cross-shard transaction through master-chain, thereby forming the two-layer blockchain network in SBC-BOMF (marked as BC); $\text{BC}_{\text{PFL-TL-DSL-BSL-AL}}$ refers to building the two-layer blockchain network and transaction logic based on the five-layer architecture to support BOM.

2) Developing transaction handling strategy for blockchain nodes to support efficient operation of SBC-BOMF, and achieve reasonable and rapid response for multi-level blockchain transactions (**Goal 3**).

4.1. Overview of SBC-BOMF

The architecture of SBC-BOMF is shown in Fig. 2. Compared with the traditional BOM model, in DSL, SBC-BOMF uses blockchain to store data to ensure system security and data reliability; in TL and DSL, interfaces of IoT-Blockchain and Blockchain-Services are added for GWs, SSs and CSs to realize data interaction with the blockchain.

Two-layer Blockchain network of the DSL is constructed based on sharding technology, and IoT data generated from PFL is stored in multiple shards to reduce the storage overhead of blockchain nodes. As shown in Fig. 3, considering subsystem is an indivisible minimum system that serves BOM applications, this paper simply integrates all subsystems according to the classification of "5A" (BA-building automation, CA-communication automation, FA-fire automation, OA-office automation, SA-security automation), and finally divides them into 5 shards and 1 master-chain:

·Shard: Nodes in each shard are composed of related GWs, SSs and CSs (for backup). CS instantiates multiple logical nodes and adds them into different shards to achieve full-copy backup. Each shard only handles intra-shard transactions related to O&M events (Section 4.2.2).

·Master-chain: Inspired by literatures [39,40] and "relay chain" technology [41], a high-performance node (i.e., SS, CS) is selected from each shard to join master-chain for atomically handling cross-shard transactions related to O&M (Section 4.2.3).

Blockchain, as a decentralized and trustless framework, can achieve the trusted storage of BOM data. The two-layer blockchain architecture enables blockchain transactions to be handled only by the nodes in relevant shards, without the participation of all nodes in the network, thereby effectively improving the overall concurrency of blockchain and reducing nodes' storage space occupied by ledger data.

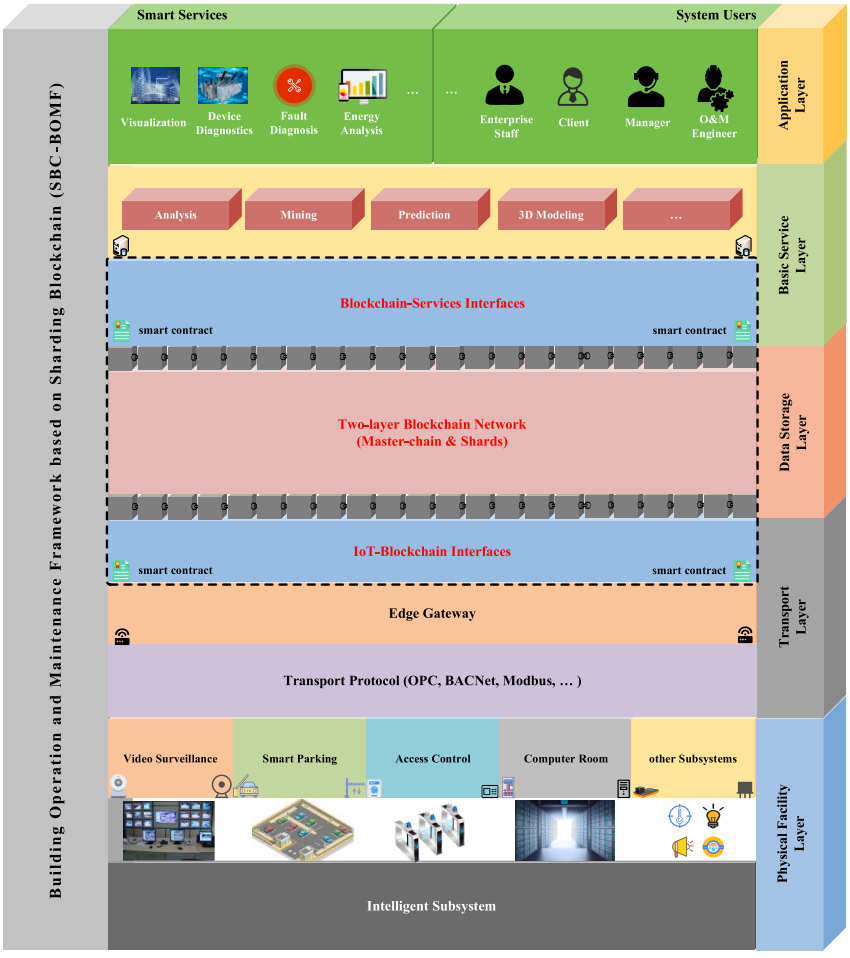


Fig. 2. Framework of SBC-BOMF.

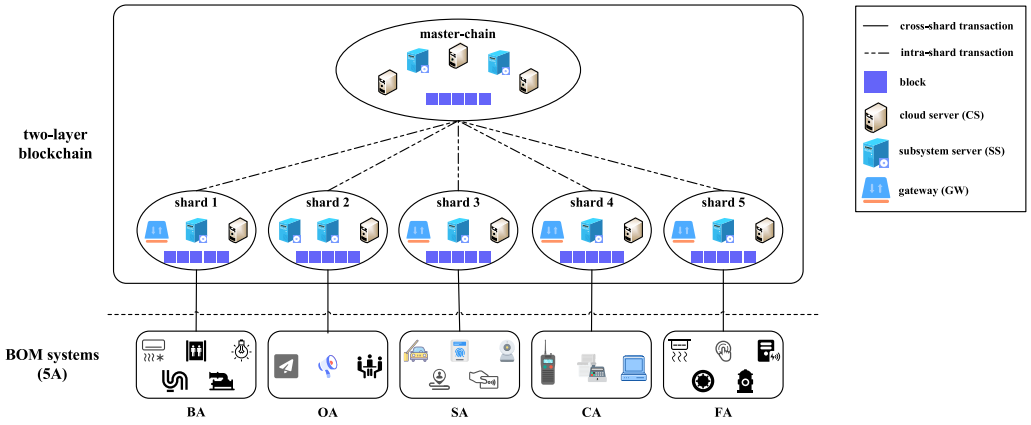


Fig. 3. Architecture of two-layer blockchain network.

4.2. *Design of sharding blockchain*

The process of shard management usually includes the following stages: shard configuration, intra-shard/cross-shard transaction consensus, and shard reorganization. BOM involves the joint cooperation of multiple organizations (building operator, property company, third-party service providers, resident enterprise, etc.) and has admission control for the joining of nodes, which can be classified as the scenario of consortium chain. In this permissioned environment, the identities of nodes are known and verifiable, and have a certain degree of trust, so there is no need to perform reorganization after sharding [34]. Therefore, this section elaborates the constructed two-layer blockchain architecture from three aspects: shard configuration, intra-shard transaction consensus and cross-shard transaction consensus.

4.2.1. *Shard configuration*

Division method of nodes is performed according to Fig. 3, and will not be repeated here. After dividing the blockchain network, the configuration is written into each shard. In TL and DSL, interfaces of IoT-Blockchain and Blockchain-Services are developed based on software development kit (SDK). These interfaces are used to call smart contracts to handle intra-shard/cross-shard transactions.

It should be added that for buildings (or parks) with different types and scales, there are great differences in the composition of subsystems, the scale of nodes/devices, and the amount of BOM data. Consequently, this paper only provides a general method of shard configuration. For a specific building, shards can be configured in combination with building's conditions.

4.2.2. *Intra-shard transaction*

The consensus process of intra-shard transactions is relatively simple and can be handled by the nodes in this shard. Take the collection and storage of air-conditioner data in BA system as an example to describe:

1) Gateway GW_01 executes a new round of data collection. Data generated from m air-conditioners are collected and integrated to $dataList = \{data_1, data_2, \ldots, data_m\}$. *data* includes: running parameters of air-conditioner, various sensor values (temperature, humidity, air volume, vibration, etc.).
2) GW_01 assembles *dataList*, timestamp into intra-shard transaction (marked as *TX*). Then, according to the configuration of shards, *TX* is submitted to shard 1 through IoT-Blockchain interface: *shard1.SubmitTransaction*("StoreDataOfIoT", *TX*).
3) *TX* is distributed to the nodes in shard 1, and the nodes call the function *StoreDataOfIoT*() of smart contract to execute *TX* and endorse the result respectively. GW_01 gathers endorsements and sends them to the leader.
4) The leader gathers multiple *TX*s and assembles them into a block, and then leads the nodes (in shard 1) to reach consensus. After the consensus, each node put the block into local ledger.

In BOM scenario, each node has a certain degree of trust, and its honesty can be guaranteed (exclude external malicious attacks). Therefore, this paper uses Raft algorithm [42] to conduct the consensus of intra-shard/cross-shard transactions and the election of leader. Raft is one of the mainstream consensus algorithms in consortium chains, which can tolerate half of non-Byzantine fault nodes and promote stable consensus among blockchain nodes.

4.2.3. *Cross-shard transaction*

In BOM scenario, the response to an O&M event may involve collaboration between multiple subsystems. For example, OA system processes client's request of visitor appointment, while SA system needs to store authorization logs and identification information to provide parking and access rights for client. In SBC-BOMF, processing such events will generate cross-shard transactions, which need to be jointly handled by relevant shards.

Taking the visitor appointment as an example, this section introduces the workflow of handling cross-shard transaction. As shown

in Fig. 4, assuming that SS_{OA} and SS_{SA} have joined master-chain, the workflow includes:

Step **1–2**: The client intends to enter the building to negotiate business with an enterprise, and then he uses App to input his identity information and submit the request (marked as *req*) which will be approved by the enterprise administrator.

Step **3–5:** After the approval, SS_{OA} (that serves OA system) converts *req* into an intra-shard transaction (marked as TX_{OA}), and submits TX_{OA} to shard 2 through Blockchain-Services interface: *shard2.SubmitTransaction*("StoreAppointment_CTX", TX_{OA}). Nodes in shard 2 make consensus for TX_{OA} and store it.

Step **6–7:** SS_{OA} creates cross-shard transaction $CTX_{OA\text{-}SA} = \{TX_{OA}, TX_{park}, TX_{access}\}$. TX_{park} and TX_{access} only contain the filed "TxID", which is used to identify and retrieve itself. $CTX_{OA\text{-}SA}$ is submitted to master-chain through SDK-based interface for intra-shard consensus (i.e., the execution status of TX_{OA} is set to "success", while TX_{park} and TX_{access} are set to "wait").

Step **8–10:** Master-chain throws event "cross_shard_relay_event" (included $CTX_{OA\text{-}SA}$) which is listened by SS_{SA} (that serves SA system). SS_{SA} obtains client's information from $CTX_{OA\text{-}SA}$ and import it into the subsystems of access control and smart parking.

Step **11–12:** SS_{SA} fills authorization logs into TX_{park} and TX_{access}, submits them to shard 3 for intra-shard consensus and storage.

Step **13–14:** SS_{SA} constructs $TX_{ctx\text{-}update}$, invokes master-chain SDK to update the status of TX_{park} and TX_{access} to "success".

Step **15–17:** Master-chain throws event "cross_shard_success_event" (means successful execution) which is listened by SS_{OA}. $CTX_{OA\text{-}SA}$ is successfully executed. SS_{OA} sends the result of authorization back to the client's App. At this point, the process of cross-shard transaction ends. The client can use parking space and get access right for entering building within the appointed time.

In addition, to effectively manage the load of cross-shard transactions, the following settings are made:

1) A counter and a timer are set inside each blockchain node. In any step, if the operation fails for 3 times consecutively or does not response for more than 30 s, the node will create a rollback transaction CTX_{rb} to undo previous operations. The above setting is used to ensure the atomicity of cross-shard transactions, and prevent cross-shard transactions from excessive occupation of processors and resources.
2) A priority-based handling strategy (Section 4.3) is designed. When shards are busy, it can reasonably utilize the resources of nodes and prioritize the processing of more important transactions.

4.3. Transaction handling strategy

The O&M events generated by BOM system have different priorities and real-time requirements. Reasonable and rapid responses (**Goal 3**) need to be achieved for relevant blockchain transactions. Oktian et al. [43] constructed a multi-priority request pool for

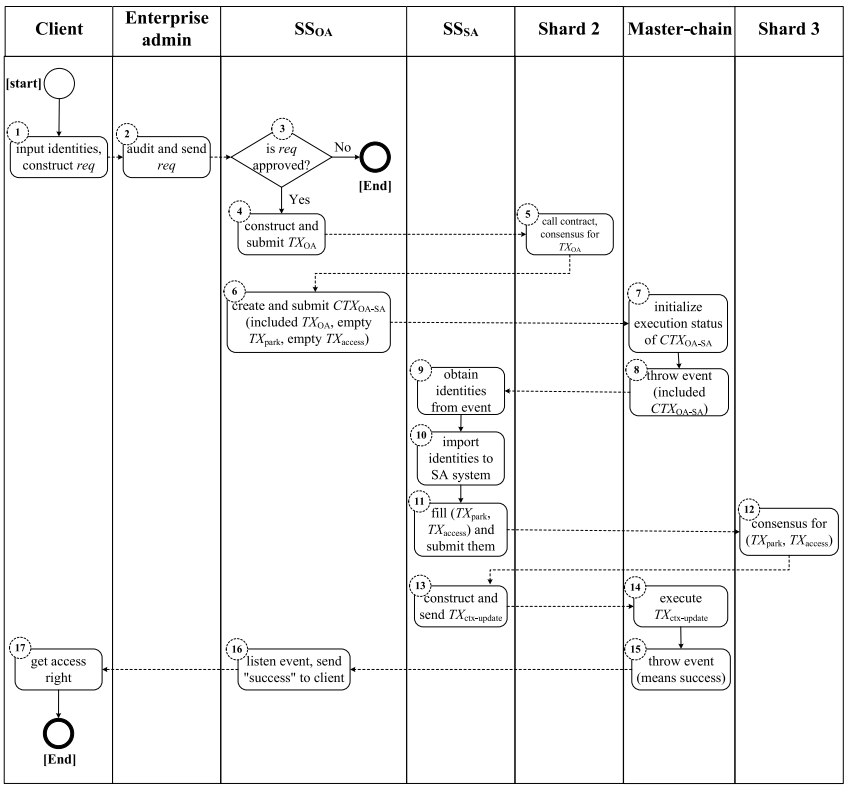


Fig. 4. Handling the cross-shard transaction.

blockchain nodes to achieve fast response to high-priority requests. However, the logic of their algorithm is relatively simple, and the real-time of events is not taken into account when handling blockchain transactions.

To this end, as shown in Fig. 5, this paper divides the transaction pending pool of blockchain nodes into sub-transaction pools with three priorities: high, medium, low (denotes as: h, m, l). Then, the blockchain transaction (intra-shard & cross-shard) is packaged into a new structure *P-TX*.

Premise assumptions: a) priority-based transaction pending pool has been constructed for each node; b) values "priority" and "timeOut" of various O&M events have been stored as the configuration of shard.

Taking the intra-shard transaction as an example, the workflow includes:

1) GW (or SS, CS) invokes SDK to constructs intra-shard transaction *TX*, and then submits *TX* to shard X for endorsing.
2) Endorsed *TX* is received and packaged into *P-TX* by the leader. The leader stores *P-TX* into local sub-transaction pool according to its priority.
3) According to **Algorithm 1**, the leader reorders transactions stored in local pending pool based on "priority" and "timeOut", takes out a certain number of transactions and assembles them into a block for intra-shard consensus.
4) After the consensus, nodes of shard X save the block containing *TX* into their ledgers to achieve the consistency of ledger data.

The following is a brief description of the pseudocode provided by **Algorithm 1**. Assumptions are:

a) gt is a timestamp when O&M event generated, rt is the maximum response time. gt and rt correspond to fields "timestamp" and "timeOut" in *P-TX* respectively.
b) n_h, n_m, n_l represent the total number of 3 priority-type transactions in the pool, which can be uniformly expressed as n_i ($i \in$ h, m, l).
c) max_{tx} is the maximum number of transactions in a block; 3 priority-type transactions account for v_h, v_m, v_l (uniformly expressed as v_i, and $v_h + v_m + v_l = 1$) respectively.
d) T_h, T_m, T_l respectively represent the number of 3 priority-type transactions pre-submitted to the block, which can be uniformly expressed as T_i (equals to the product of max_{tx} and v_i).

Algorithm 1: Transaction handling strategy

Input: max_{tx} is the maximum number of transactions in a block
v_h, v_m, v_l represent the ratio of transactions with three priorities ($v_h + v_m + v_l = 1$)
Output:
R is a list of transactions that will be assembled into a block for consensus
Pseudo code:
1) get transactions lists Q_h, Q_m, Q_l from local pending pool (sub-transaction pools)
2) get the number of transactions n_h, n_m, n_l in Q_h, Q_m, Q_l
3) //quantity of transactions in the pending pool exceeds max_{tx}
4) **if** $n_h + n_m + n_l > max_{tx}$ **then**
5) calculate the number of pre-submitted transactions in Q_h, Q_m and Q_l:
$T_h = max_{tx} * v_h$, $T_m = max_{tx} * v_m$, $T_l = max_{tx} * v_l$
6) obtain the current timestamp ts
7) //calculate remaining response time $repT$ of each transaction and sort Q_h, Q_m and Q_l in ascending order based on $repT$. The formula is: $repT = rt - (ts - gt)$
8) Q_h, Q_m, $Q_l = Order(ts, Q_h, Q_m, Q_l)$
9) //compare n_i and T_i, if $n_i < T_i$, assign $|T_i - n_i|$ to higher priority

(*continued on next page*)

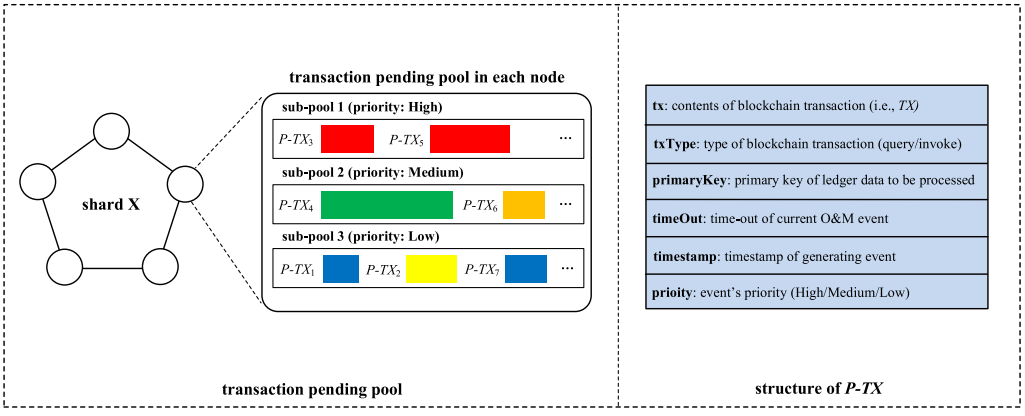


Fig. 5. Transaction pending pool based on priorities.

(*continued*)

Algorithm 1: Transaction handling strategy

10) //calculate number of actual submitted transactions T_h', T_m', T_l'
11) T_h', T_m', T_l' = *CalRealTxs*(T_h, T_m, T_l, n_h, n_m, n_l)
12) $R \leftarrow Q_h[1 \ldots T_h']$, $R \leftarrow Q_m[1 \ldots T_m']$, $R \leftarrow Q_l[1 \ldots T_l']$
13) **else**
14) $R \leftarrow Q_h$, $R \leftarrow Q_m$, $R \leftarrow Q_l$
15) **end if**
16) Remove the corresponding transactions stored in sub-transaction pools
17) **output** R

First, the leader calculates whether the total number of transactions in local pending pool is greater than max_{tx}:

1) If not, the leader reads all transactions stored in the pool and output the list R. At this point, the process ends. (Leader can parse *TX*s from R and assemble them into a block for consensus.)
2) Else, the leader calculates the number of pre-submission transactions for each sub-transaction pool (recorded as T_h, T_m, T_l) according to the parameters max_{tx} and ratio v_h, v_m, v_l. The leader judges whether the number of transactions in each sub-transaction pool (recorded as n_i) is greater than (or equal to) T_i:

- If n_i are all greater than (or equal to) T_i, then T_h, T_m, T_l represent the number of submission transactions for each priority (recorded as T_h', T_m' and T_l' respectively).
- Else, the leader assigns the difference quantity $|T_i - n_i|$ to higher priority, obtains the number of actual submission T_h', T_m', T_l'. The specific logic for calculating the actual submission of 3 priority-type transactions is shown in **Algorithm 2.**

After obtaining the actual number of submission, the leader respectively obtains the first T_h', T_m', T_l' transactions from three sub-transaction pools to form the list R. R is output and used to construct the block.

Algorithm 2: Function *CalRealTxs*()

Input:
T_h, T_m, T_l respectively represent the number of 3 priority-type transactions pre-submitted to the block. n_h, n_m, n_l respectively represent the total number of 3 priority-type transactions in the pool.
Output:
T_h', T_m', T_l' respectively represent the number of 3 priority-type transactions actually submitted to the block.
Pseudo code:
1) //judge the value of n_i and T_i, so as to adjust T_i' (i ∈ h, m, *l*)
2) **if** $n_h \geqq T_h$ **and** $n_m \geqq T_m$ **and** $n_l \geqq T_l$ **then**
3) //T_i' equals to the number of pre-submitted transactions T_i
4) $T_h' = T_h$, $T_m' = T_m$, $T_l' = T_l$
5) **else if** $n_h \geqq T_h$ **and** $n_m \geqq T_m$ **and** $n_l < T_l$ **then**
6) $T_l' = n_l$
7) **if** $T_h + (T_l - n_l) \geqq n_h$ **then**
8) //set T_h' equals to n_h, while T_m' equals to the minimum between n_m and the sum of differences
9) $T_h' = n_h$, $T_m' = \min\{T_m + (T_l - n_l) - (n_h - T_h), n_m\}$
10) **else**
11) //assign the difference quantity of $|T_l - n_l|$ to T_h'
12) $T_h' = T_h + (T_l - n_l)$, $T_m' = T_m$
13) **end if**
14) **else if** $n_h \geqq T_h$ **and** $n_m < T_m$ **and** $n_l \geqq T_l$ **then**
15) $T_m' = n_m$
16) **if** $T_h + (T_m - n_m) \geqq n_h$ **then**
17) $T_h' = n_h$, $T_l' = \min\{T_l + (T_m - n_m) - (n_h - T_h), n_l\}$
18) **else**
19) $T_h' = T_h + (T_m - n_m)$, $T_l' = T_l$
20) **end if**
21) **else if** $n_h \geqq T_h$ **and** $n_m < T_m$ **and** $n_l < T_l$ **then**
22) //set T_h' equals to the minimum between n_h and the sum of differences, while T_m' and T_l' equal to n_m and n_l respectively
23) $T_h' = \min\{T_h + (T_m - n_m) + (T_l - n_l), n_h\}$, $T_m' = n_m$, $T_l' = n_l$
24) **else if** $n_h < T_h$ **and** $n_m \geqq T_m$ **and** $n_l \geqq T_l$ **then**
25) $T_h' = n_h$
26) **if** $T_m + (T_h - n_h) \geqq n_m$ **then**
27) $T_m' = n_m$, $T_l' = \min\{T_l + (T_h - n_h) - (n_m - T_m), n_l\}$
28) **else**
29) $T_m' = T_m + (T_h - n_h)$, $T_l' = T_l$
30) **end if**
31) **else if** $n_h < T_h$ **and** $n_m \geqq T_m$ **and** $n_l < T_l$ **then**
32) $T_h' = n_h$, $T_m' = \min\{T_m + (T_h - n_h) + (T_l - n_l), n_m\}$, $T_l' = n_l$
33) **else if** $n_h < T_h$ **and** $n_m < T_m$ **and** $n_l \geqq T_l$ **then**

(*continued on next page*)

(*continued*)

Algorithm 2: Function *CalRealTxs*()

34) $T_h' = n_h$, $T_m' = n_m$, $T_l' = \min\{T_l + (T_h - n_h) + (T_m - n_m), n_l\}$
35) **else**
36) $T_h' = n_h$, $T_m' = n_m$, $T_l' = n_l$
37) **end if**
38) **output** T_h', T_m', T_l'

It should be added that the following situations may be encountered during the process of transactions: a) The same block contains 2 or more conflict transactions (i.e., conflicting read and write operations on data with the same primary key); b) A large number of transactions accumulate in leader's pending pool, which increases the time-cost of reordering.

In response to the above problems, the following settings are made in this paper:

·The fields "primaryKey" and "timestamp" are used to ensure the order of transactions with the same primary key. For conflict transactions in the same block, the transaction with an earlier timestamp is regarded as a valid transaction, and the others are regarded as invalid.

·Threshold $F = c * max_{tx}$ is set (c is set to 10 here) to control the sorting time within a certain range. Each sub-transaction pool only reorders the first F transactions.

5. Experiments and analysis

This section will evaluate proposed scheme in terms of effectiveness analysis, performance analysis, and security analysis. Firstly, in Section 5.1, an actual BOM project is used as the example to illustrate the effectiveness of proposed scheme. Secondly, in Section 5.2, performance experiments are conducted, and testing results are analyzed. Finally, security of proposed scheme is discussed in Section 5.3.

5.1. Use case

To verify the effectiveness of proposed blockchain-based scheme in BOM scenario, a prototype system was built and tested on real-world BOM project. The project is an actual building in Qingdao, China. According to the division of the Chinese National Standard [44], this building belongs to the "supertall" office building. It covers an area of about 63,000 m^2, with a height of 128.5 m and 30 floors, and can be rented by innovative companies and start-ups. The building is equipped with 1030 intelligent devices (e.g., smart meter, fresh air unit, access control controller, lighting circuit, alarm host). The project covers the "5A" system, and the gateways collect device data once a minute. Management of facilities and personnel can be realized through BIM visualization platform (see Fig. 6).

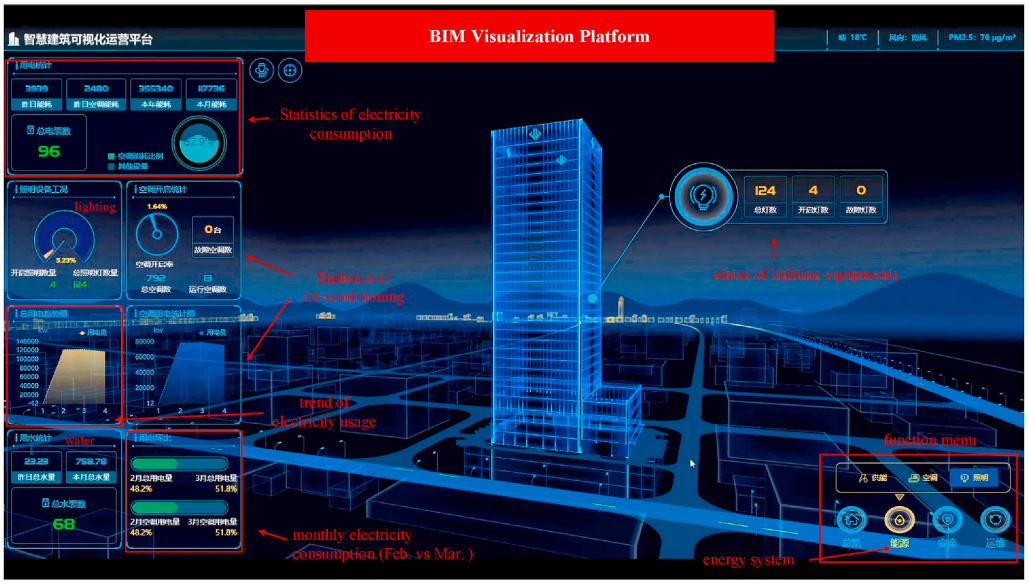


Fig. 6. BIM visualization platform.

5.1.1. Technical implementation

Restricted by economic conditions, Raspberry Pi 4B (4-core processor, 2 GB memory) are used to simulate gateways, and multiple virtual machines (4-core processor, 4 GB memory) are built in 2 computers to simulate the subsystem servers and cloud servers. Among them, the function and performance of Raspberry Pi 4B are similar to those of gateways in building, which can simulate the gateway well [45].

Since OA system does not contain the gateway, this paper sets the simulation environment as: 5 virtual machines (VM_{SS_BA}, VM_{SS_CA}, VM_{SS_FA}, VM_{SS_OA}, VM_{SS_SA}) and 4 Raspberry Pi (GW_{BA}, GW_{CA}, GW_{FA}, GW_{SA}) are used to simulate SSs and GWs in "5A″ respectively; 2 virtual machines (VM_{CS_1}, VM_{CS_2}) and 1 virtual machine (VM_{CS_OP}) are used to simulate CSs to provide data backup and integrated O&M services.

Considering that BOM can be classified into the scenario of consortium chain, this paper adopts Hyperledger Fabric as the blockchain platform. Fabric provides open source code for secondary development, supports multiple languages for smart contract and SDK (e.g., Golang, Java, Python).

Referring to Fig. 3, blockchain network consists of 1 master-chain and 5 shards. The network is initialized based on multi-channel technology of Fabric. Table 1 shows the mapping relationships of Fabric nodes. The two organizations are denoted as org1 and org2, representing the building operator and project developer respectively (not provide blockchain identity for enterprises and residents, they only obtain blockchain services through SDK). Table 2 shows the configuration of each channel (i.e., shard), select VM_{CS_*} that simulate the high-performance CSs to form the master-chain ("*" means the wildcard).

Subsequently, smart contracts are written based on Golang language (see Fig. 7) and installed to each channel in the form of chaincode. IoT-Blockchain and Blockchain-Services interfaces are developed using SDK to complete the preliminary construction of the prototype system.

5.1.2. Scenario: visitor appointment

Similarly, this section takes the scenario of visitor appointment as an example to describe the simple workflow of the prototype system. Meanwhile, since Fig. 4 has already provided the detailed steps of cross-shard scenario, this section will focus on describing the business data in the scenario.

1) Stage 1: Channel "oa" processes request and generates cross-shard transaction

As shown in Fig. 8, client "Li Hua" intends to enter building to negotiate business with a group, and then he uses App to make an appointment. Request of appointment is approved by this group and recorded in "oa" with the form of transaction (TX_{OA}). Visualization page (Fig. 9) provided by Fabric can be used to view the world state database of nodes and search the details of TX_{OA}. The "world state" is a specific term in Hyperledger Fabric, which represents the latest values for all keys included in the chain transaction log.

Subsequently, "oa" generates cross-shard transaction which is routed to the nodes of channel "sa" via master-chain (i.e., channel "master").

2) Stage 2: Handling cross-shard transaction

As shown in Fig. 10, "sa" handles cross-shard transaction, and imports client's identities into access control subsystem. TX_{access} is stored in the channel, and the relevant information can be seen in Fig. 11(this project is not linked with smart parking subsystem, so the prototype system omits the corresponding operation).

TX_{access} is handled by the nodes in "sa", the status of this transaction in channel "master" is updated to "success". Event is listened by "oa", and the authorization information is fed back to "Li Hua". After that, within the appointed time, "Li Hua" can enter the building by QR code or face recognition, and negotiate business with this group.

The prototype system provided a visualization page (Fig. 12) based on the ledger data of channel "master", showing the transaction ID and execution status of cross-shard transactions. Manager can retrieve the detailed information based on transaction ID to realize the traceability of blockchain transaction. Participants in this BOM project spoke highly of this prototype system, believing that the application of blockchain technology can significantly improve system stability and data security.

5.2. Performance test

This section conducts performance tests on concurrency, storage overhead, and the designed transaction handling strategy. BOM project (provided in Section 5.1) is used to evaluate the usability of prototype system in the actual BOM scenario. Some performance

Table 1
Mapping relationship of Fabric nodes.

Installation	Organization	Mapped node
GW*	org1	1 peer node
VM_{SS_*}		1 orderer node
VM_{CS_1} /VM_{CS_2}	org2	1 orderer node
VM_{CS_OP}		1 peer node

Table 2
Configuration of Fabric channels.

Channel ID	Nodes	Involved	Consensus
ba	2 peer, 3 orderer	GW_{BA}, VM_{SS_BA}, VM_{CS_*}	Raft
ca	2 peer, 3 orderer	GW_{CA}, VM_{SS_CA}, VM_{CS_*}	
fa	2 peer, 3 orderer	GW_{FA}, VM_{SS_FA}, VM_{CS_*}	
oa	1 peer, 3 orderer	VM_{SS_OA}, VM_{CS_*}	
sa	2 peer, 3 orderer	GW_{SA}, VM_{SS_SA}, VM_{CS_*}	
master	1 peer, 2 orderer	VM_{CS_*}	

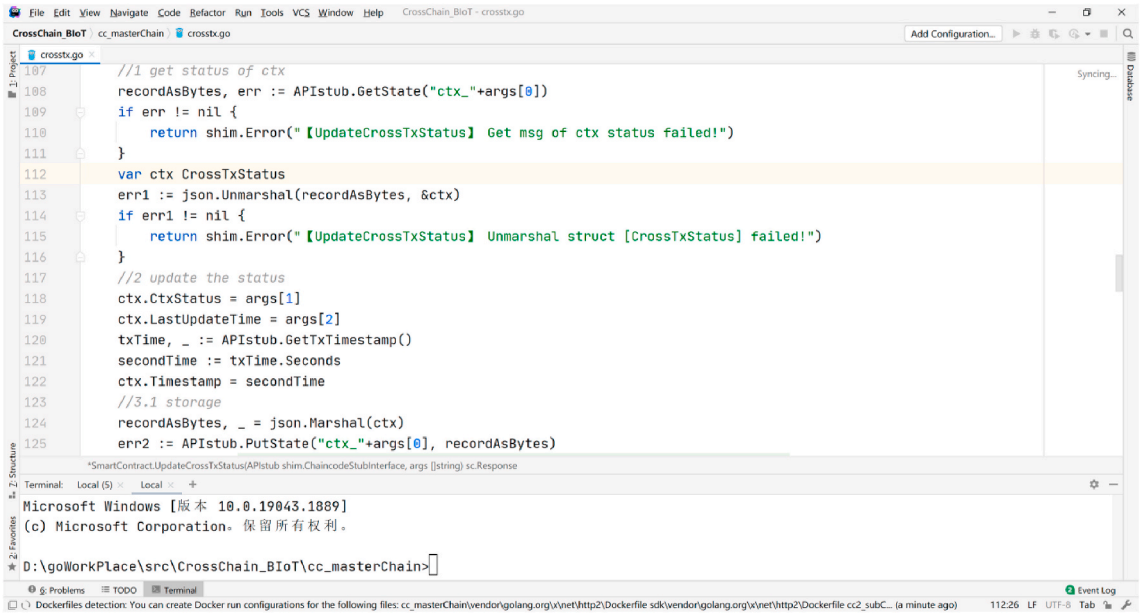


Fig. 7. Screenshot of the partial code of smart contracts.

indicators of this project are as follows (only on weekdays): BOM system handles about 30 transactions per second (recorded as 30tps), with a peak value of about 100tps; about 582 MB of data is added to the database every day (including IoT data and application data).

5.2.1. Test of concurrency

Events generated in BOM involve query and writing operations of the blockchain ledger. If the throughput of the blockchain is too low, it will cause network congestion and affect the use of system.

Firstly, throughput of the blockchain-based prototype system is tested. Literature [32] is a typical scheme for integrating blockchain into BOM application, which is used for comparison with proposed scheme in this paper. In addition to the setting of shards, prototype systems built based on proposed scheme and scheme [32] use the same environment. Environments are set as follows: a) use the default configuration of block in Fabric (maximum generation time is 2 s, max_{tx} is 10, and consensus algorithm is Raft), and construct the transaction pending pool in orderer; b) since the throughput of different writing operations is similar, the storage transaction of application data (about 0.4 KB) is used for the test; c) shard configuration of proposed scheme is set according to Table 2, while the prototype built based on scheme [32] uses single-chain architecture (i.e., all nodes deployed in one channel).

Continuously adjust the sending rate of blockchain transactions, send transactions to channels (exclude channel "master"), test the maximum throughput and compare with the non-shard scheme [32] (single channel). As shown in Fig. 13, maximum throughput of the prototype system based on proposed scheme is 283.2tps, which is significantly improved compared to the non-shard scheme [32] (78.3tps). After the peak is reached, the throughput decreases due to the large data volume and the high usage of memory.

The prototype system in this paper is built based on the BOM project described in Section 5.1, and its throughput can meet the requirements of project. However, for the medium or large scale of parks, the scale of system and blockchain nodes will be larger, and the concurrency requirements will also be higher. More shards may need to be built to meet the concurrency requirements of system. Therefore, in practical application, it is necessary to design reasonable sharding architecture according to the scale of project and requirements of performance.

5.2.2. Test of proposed strategy

This section tests the designed transaction processing strategy. Further, the superiority of strategy in this paper is verified by

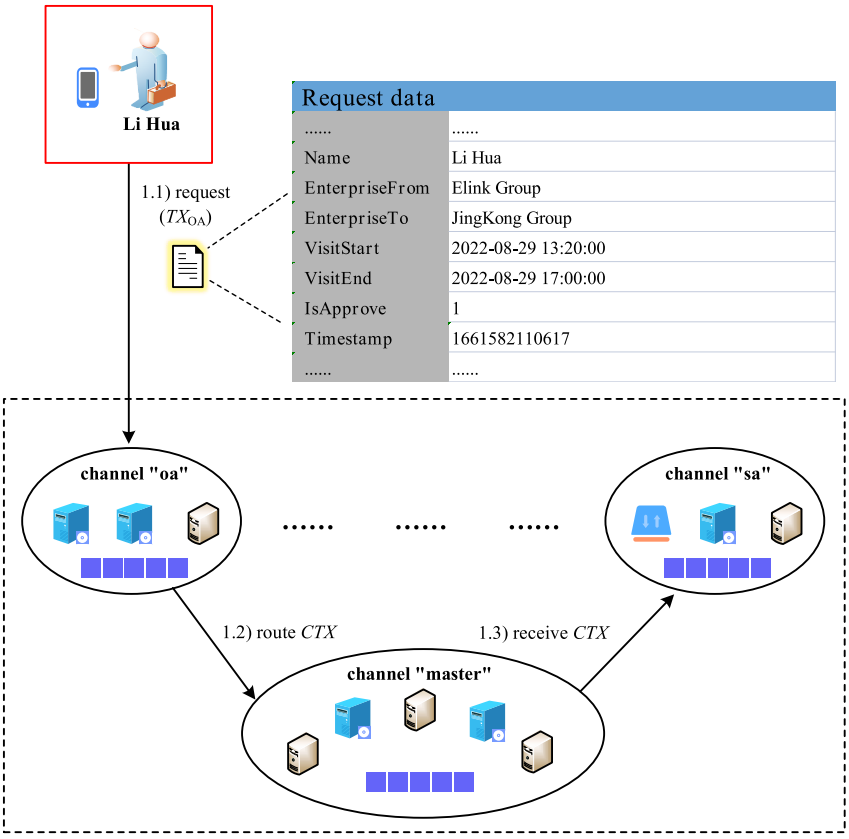


Fig. 8. Stage 1: Processing client's request and generating cross-shard transaction.

comparing the time-out rate with Oktian scheme [43] and non-priority scheme.

Firstly, n_h: n_m: n_l is used to represent the ratio of the total number of 3 priority-types transactions needed to be handled, while v_h: v_m: v_l represents the ratio of 3 priority-types transactions pre-submitted to one block. Then, the following three settings are used to test the effectiveness of proposed strategy under different dispersion of transactions:

(a) Test 1 — v_h: v_m: v_l = 5:3:2, n_h: n_m: n_l = 1:1:1
(b) Test 2 — v_h: v_m: v_l = 4:4:2, n_h: n_m: n_l = 1:1:1
(c) Test 3 — v_h: v_m: v_l = 5:3:2, n_h: n_m: n_l = 1:3:6

To make the results more intuitive, test is conducted in single-channel environment. The configuration of block generation remains unchanged; For one block, the number of transactions with three priorities is set to 5, 3, 2 respectively; use Autocannon [46], a HTTP benchmarking tool, to simulate the generation of O&M events and send them to VM_{CS_OP} (processes events and generates blockchain transactions).

The above three tests use the same configuration of Autocannon: a) to simulate the concurrency peak in this project, sending rate of O&M events is set to 100tps, and the duration is set to 30 s; b) to ensure the validity and comparability of results, three schemes uniformly use the pre-generated 3000 O&M events (shuffle these events, i.e., randomize the order of sending); c) take the sending time as the field "timestamp" of *P-TX*, the time of feedback to Autocannon as the completion time; d) combined with the real-time suggestions and requirements for each O&M event in standard "Code for acceptance of quality of intelligent building systems" [47] and its reference standards, the time-out of each event is set to 2–10 s.

According to the above settings, test the designed strategy when generation rate of events (100tps) is greater than throughput (78.3tps). First, the number of transactions of each priority recorded in each block is counted. Since the strategy in this paper and Oktian scheme contain the same number of transactions with different priorities in each block, they are uniformly described in Fig. 14. The statistical result of non-priority scheme is described in Fig. 15. For the non-priority scheme (Fig. 15), it is difficult to achieve fast response to high-priority transactions, since transactions are packaged into blocks according to the order in which they are stored in the pending pool. If the priority-based strategy is adopted (Fig. 14), the node handles the transaction with higher priority preferentially when multi-priority transactions exist in the pending pool. This behavior can be seen in the first 250 generated blocks.

Furthermore, the test data in Fig. 14 shows that the difference of dispersion will only impact the speed of handling different priority-type transactions (can be seen by comparing the number of 3 priority-type transactions in each block), but not change the

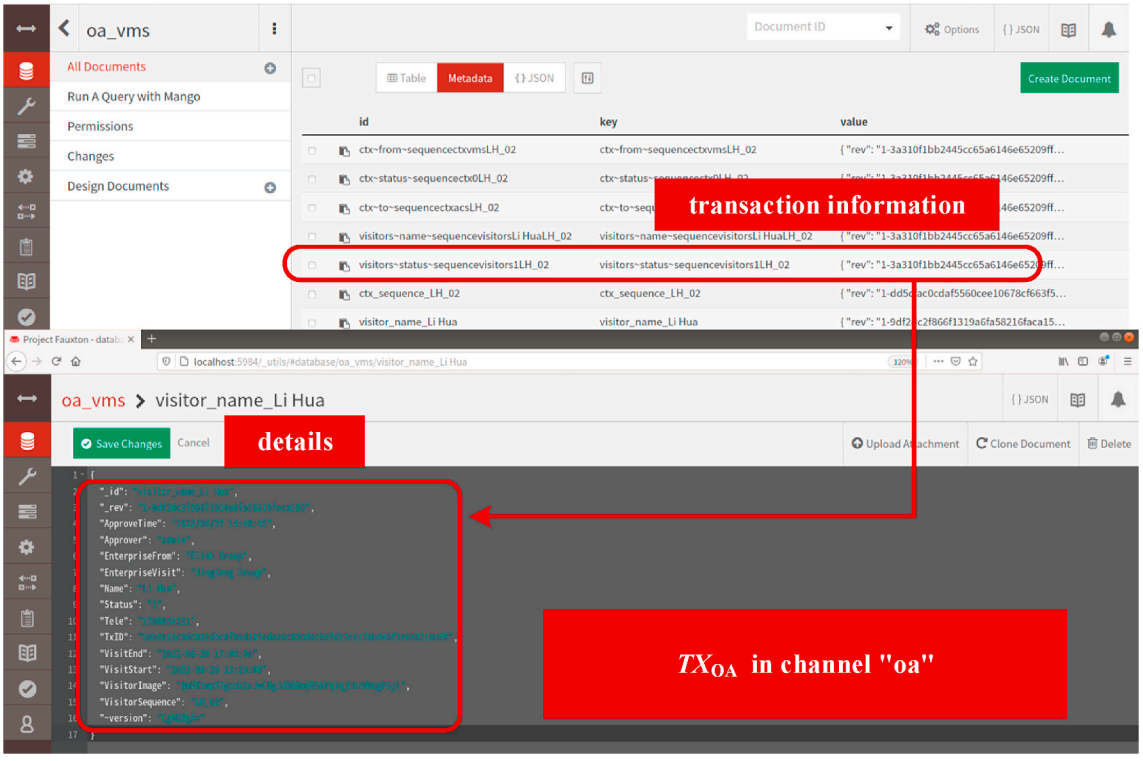


Fig. 9. Screenshot of world state database (in channel "oa").

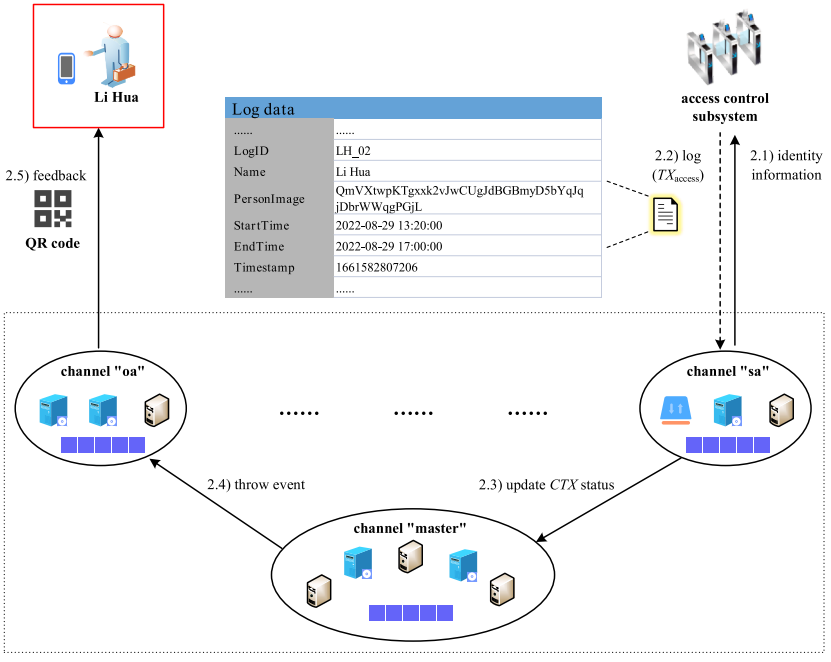


Fig. 10. Stage 2: Handling cross-shard transaction.

trend of "handling higher priority transactions preferentially". Therefore, the dispersion of transactions will not impact the effectiveness of proposed priority-based strategy.

Then, according to the generation time and completion time of each O&M event, the timeout rate under the three strategies is

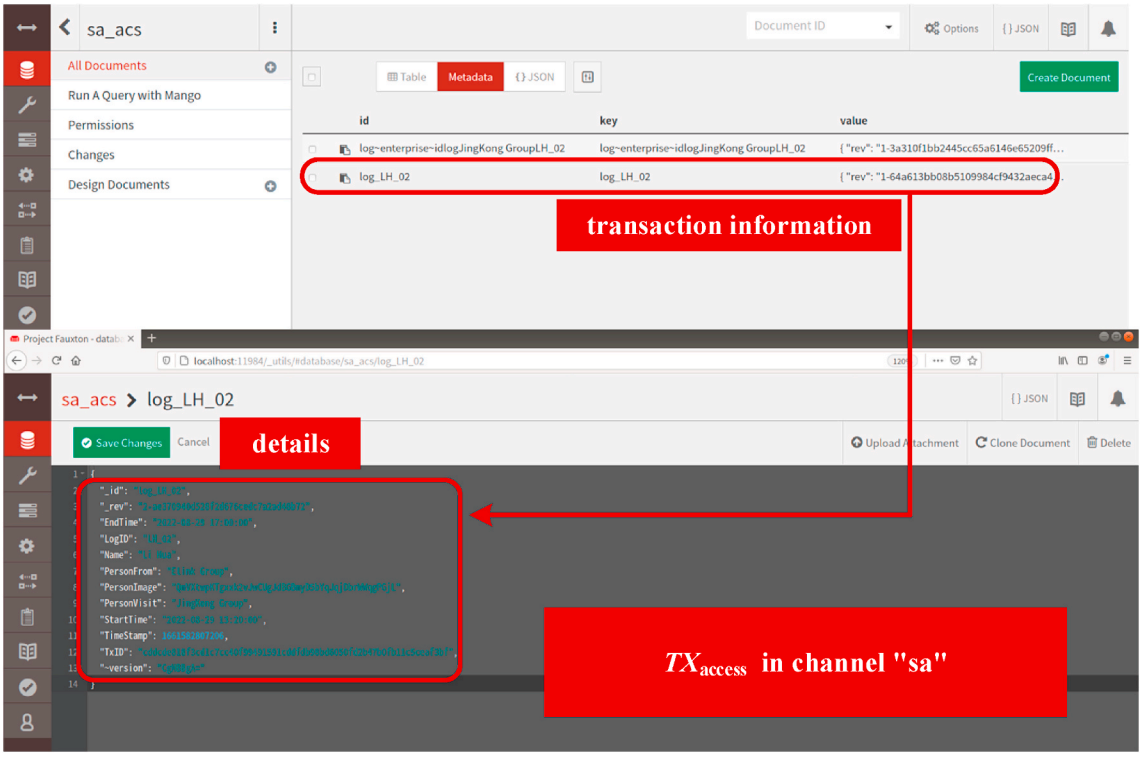


Fig. 11. Screenshot of world state database (in channel "sa").

FromSys	From_TxID	ToSys	To_TxID	Status	CreateTime	Options
	transaction in channel "oa"		transaction in channel "sa"			
OA	2eb2e11c3dc881dbc9f0ed187ede...	SA	cddcde818f3cd1c7cc40f9949159...	Success	2022-08-27 14:46:49	details
OA	dd4a716a2ac1d05216e347fbc277...	SA	003bf6b533c23d7abf67373285a4...	Success	2022-08-27 14:46:34	details
SA	71ae9e6a2a7c8657c313fb175be9...	FA	7af7b7d19059c9d68d16fc3941be...	Success	2022-08-27 14:45:51	details
OA	9c5c8bf0d38d16988c4860a079c2...	SA	c2d555efe00ca782f93208a33e30...	Success	2022-08-27 14:45:30	details

Fig. 12. Log of the cross-shard transaction.

calculated, and the results are shown in Table 3. The strategy in this paper takes both priority and real-time into consideration, and handles most urgent transactions with higher priority. Therefore, the timeout rate of events with different priorities is lower than that of Oktian strategy. No-priority scheme is equivalent to the FIFO mode of single-queue, so the timeout rate of high-priority events is significantly higher than the strategy in proposed scheme and Oktian scheme.

Moreover, in Test 1 and Test 2, the timeout rate of low-priority events in Oktian scheme is slightly higher than that in no-priority scheme. Since Oktian scheme only sorts events based on the priority, the handling of low-priority events will be delayed, and the urgent events with low-priority cannot be responded rapidly, resulting in an increase in the timeout rate. From Fig. 14, it can be known that only a few low-priority events are included in the first 250 generated blocks, which can prove the result in the first two tests.

By comparing the timeout rate of proposed strategy under the three settings, it can be known that the difference of dispersion will impact the efficiency for handling transactions. Take high-priority transactions as an example, the higher the value of ν_h or the lower the value of n_h, the lower the timeout rate. Therefore, in BOM application, the values of ν_h, ν_m, ν_l should be reasonably set according to

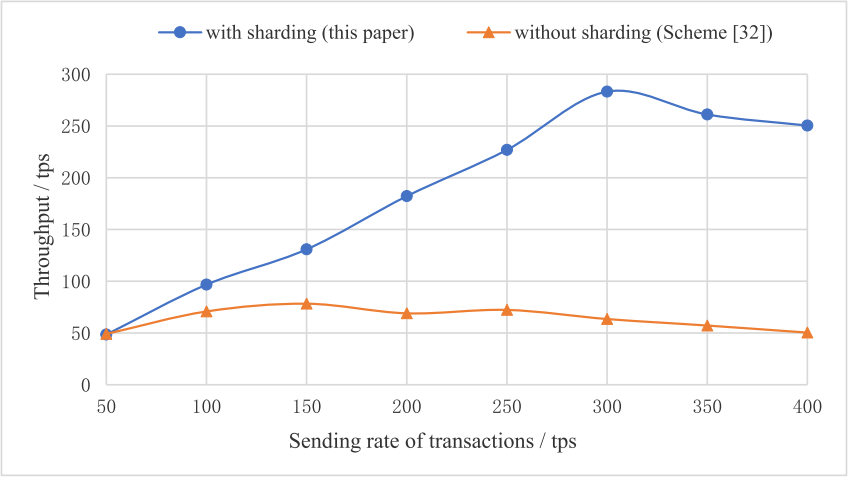


Fig. 13. Comparison of throughput.

the actual generation rate of different priority-type transactions.

Through the proportion statistics of each priority-type transactions in blocks (Figs. 14 and 15) and the comparison of timeout rate (Table 3), effectiveness and efficiency of the strategy in this paper can be verified. That is, achieving reasonable and rapid response for multi-level blockchain transactions (**Goal 3**).

5.2.3. Test of storage overhead

On the basis of the built prototype system, the data volume of the project in one day is simulated and stored in blockchain, and the ledger size of each channel is counted (Table 4). Since the channel "master" only stores the status related to cross-shard transactions, its ledger is the smallest (accounting for only 26.1 MB), while the channel "ba" has the largest ledger (accounting for 260.2 MB). Compared with the global ledger (870.9 MB, which can be obtained by accumulating the ledger size of each channel), it can be seen that the application of sharding technology can effectively reduce the hard disk space occupied by blockchain nodes to store ledger data.

In addition, different buildings have different O&M considerations (e.g., the scale of devices and usage frequency in each subsystem are different), which leads to a large difference in the data volume generated by each subsystem (Table 4). To fully utilize the storage space of each blockchain node, it is necessary to design the sharding and storage mode that conforms to the characteristics of current building in combination with the actual environment.

5.3. Analysis of security

According to the triad of information security [48], this section discusses the security of proposed scheme from the perspective of confidentiality, integrity and availability.

(1) Confidentiality

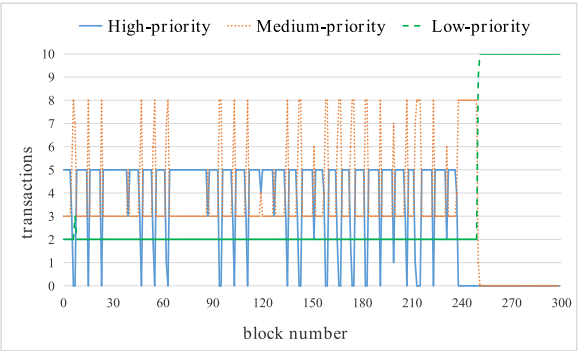
Since BOM belongs to the scenario of consortium chain, nodes need to be certified by certificate authority (CA) before joining the blockchain network. Nodes implement encrypted communication with each other based on the certificate provided by CA, thereby guaranteeing the confidentiality in the phase of data transmission.

In addition, users at AL layer do not directly join the blockchain network for participating the consensus of transaction. The application requests initiated by users are uniformly processed by cloud server that provides integrated smart O&M services. Only after user's identity is judged to be legitimate, the request will be converted into the corresponding blockchain transaction, thus preventing illegal users from obtaining or writing ledger data.

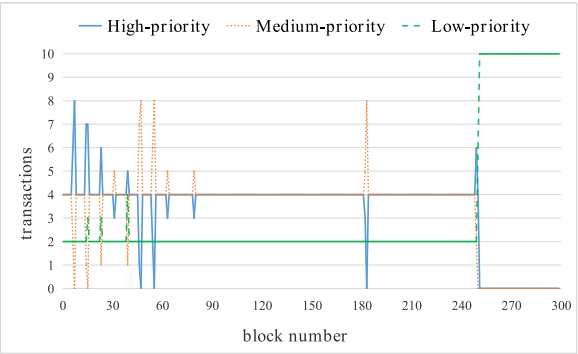
(2) Integrity

In this paper, blockchain technology is integrated into BOM to ensure the integrity of data by using hash function and digital signature. Data storage transactions need to be endorsed and signed by nodes. The leader node assembles multiple signed transactions into a block, constructs block hash in block header based on hash function. The block will be distributed among nodes for consensus and storage.

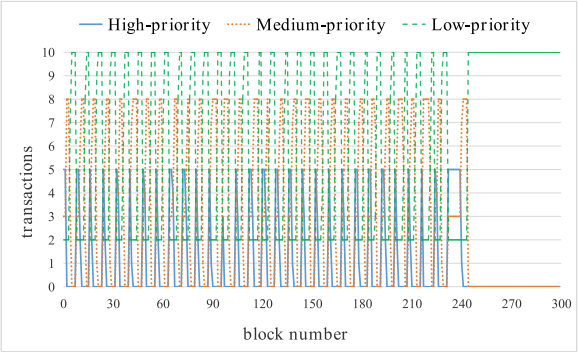
During the verification of data integrity, since intra-shard/cross-shard transactions are attached with transaction ID (Fig. 12), a specific block can be located, the correctness and integrity of block data can be verified through block header, signatures, etc.



(a) Test 1 — v_h: v_m: v_l = 5:3:2, n_h: n_m: n_l = 1:1:1



(b) Test 2 — v_h: v_m: v_l = 4:4:2, n_h: n_m: n_l = 1:1:1



(c) Test 3 — v_h: v_m: v_l = 5:3:2, n_h: n_m: n_l = 1:3:6

Fig. 14. Number of 3 priority-type transactions in each block with different dispersions (based on strategy with priority).

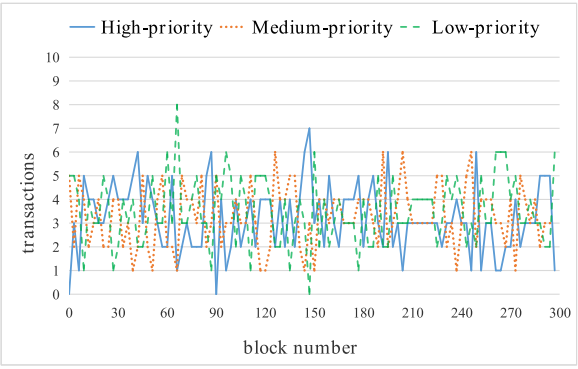
(3) Availability

In SBC-BOMF, IoT data and application data generated in BOM are stored in the distributed ledger, which can provide partition tolerance for the BOM system. If one node fails, system can still obtain O&M data from other nodes, which improves the robustness of system.

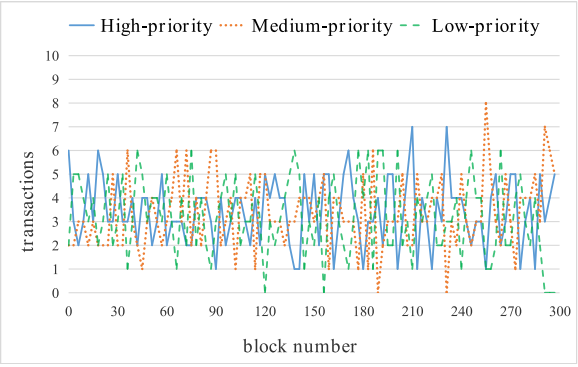
6. Conclusion

The emergence of blockchain provides a new technical mean for achieving the trusted data storage in BOM. However, due to the inherent performance bottleneck of single-chain architecture, existing blockchain-based schemes have encountered the problems of high storage overhead and low concurrency. Furthermore, these schemes ignore the impact of the priority and real-time of O&M event on blockchain transaction, and lack the mechanism or strategy for efficiently handling the transactions.

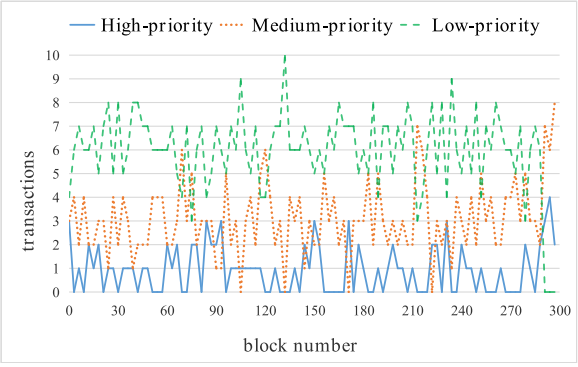
In response to these problems, this paper proposes a BOM framework based on sharding blockchain (named SBC-BOMF), which realizes data offloading and parallel transaction processing by sharding technology, thereby solving the problems of storage limitation and scalability in the existing schemes. Secondly, a handling strategy for blockchain transactions is designed, which can achieve reasonable and rapid response to transactions based on priority and real-time of O&M events. Finally, the effectiveness of proposed



(a) Test 1 — v_h: v_m: v_l = 5:3:2, n_h: n_m: n_l = 1:1:1



(b) Test 2 — v_h: v_m: v_l = 4:4:2, n_h: n_m: n_l = 1:1:1



(c) Test 3 — v_h: v_m: v_l = 5:3:2, n_h: n_m: n_l = 1:3:6

Fig. 15. Number of 3 priority-type transactions in each block with different dispersions (without strategy).

Table 3
Comparison of the timeout rate with different priorities.

Test No.	Test 1			Test 2			Test 3		
Priority-type	h	m	*l*	h	m	*l*	h	m	*l*
This paper	**4.2%**	**7.8%**	**11.6%**	**4.7%**	**7.1%**	**11.8%**	**2.0%**	**7.3%**	**10.7%**
Oktian [43]	7.3%	11.7%	15.4%	8.2%	11.3%	14.9%	4.3%	11.0%	13.5%
Without priority	13.1%	12.8%	13.6%	12.5%	13.7%	13.4%	12.3%	12.7%	14.2%

scheme is verified through prototype construction and performance tests.

This study is not free from limitations: 1) Sharding will downgrade the security: nodes are honest and credible in BOM, but for external security attacks, an effective detection and prevention mechanism should be designed. 2) Efficient storage mechanism needs to be designed to reasonably allocate the storage locations based on storage capacity of each node.

Therefore, future works will focus on:

Table 4
Storage overhead.

Channel	Storage overhead (ignore redundancy)
ba	260.2 MB
ca	82.7 MB
fa	163.3 MB
oa	104.0 MB
sa	234.6 MB
master	26.1 MB
Total (global ledger)	**870.9 MB**

1) Designing a blockchain security management mechanism based on intrusion detection systems (IDS) to prevent external security attacks against blockchain nodes.
2) Combining erasure code with sharding blockchain, and designing an efficient storage allocation strategy to further reduce the storage requirements of each node in the shard.

Author contribution statement

Jinlong Wang, Ph. D.: Conceived and designed the experiments; Wrote the paper.
Xu Wang, M. S. candidate: Performed the experiments; Analyzed and interpreted the data; Wrote the paper.
Yumin Shen: Conceived and designed the experiments.
Xiaoyun Xiong: Performed the experiments.
Wenhu Zheng: Analyzed and interpreted the data.
Peng Li; Xiaoxue Fang: Contributed reagents, materials, analysis tools or data.

Funding statement

Support for this work was provided by National Natural Science Foundation of China [62001262], National Key R&D Program of China [2020YFB1711903], Key Technology Research and Development Program of Shandong [2019GGX101017].

Data availability statement

Data included in article/supp. material/referenced in article.

Declaration of interest's statement

The authors declare no competing interests.

References

[1] Z. Ma, Y. Ren, X. Xiang, Z. Turk, Data-driven decision-making for equipment maintenance, Autom. ConStruct. 112 (2020), 103103, https://doi.org/10.1016/j.autcon.2020.103103.
[2] M. Marocco, I. Garofolo, Integrating disruptive technologies with facilities management: a literature review and future research directions, Autom. ConStruct. 131 (2021), 103917, https://doi.org/10.1016/j.autcon.2021.103917.
[3] Z.Z. Hu, P.L. Tian, S.W. Li, J.P. Zhang, BIM-based integrated delivery technologies for intelligent MEP management in the operation and maintenance phase, Adv. Eng. Software 115 (2018) 1–16, https://doi.org/10.1016/j.advengsoft.2017.08.007.
[4] W. Tushar, N. Wijerathne, W.T. Li, C. Yuen, H. Vincent Poor, T.K. Saha, K.L. Wood, Internet of things for green building management: disruptive innovations through low-cost sensor technology and artificial intelligence, IEEE Signal Process. Mag. 35 (5) (2018) 100–110, https://doi.org/10.1109/MSP.2018.2842096.
[5] L. Bottaccioli, A. Aliberti, F. Ugliotti, E. Patti, A. Osello, E. Macii, A. Acquaviva, Building energy modelling and monitoring by integration of IoT devices and building information models, 2017, in: IEEE 41st Int. Comput. Softw. Appl. Conf., IEEE Computer Society, 2017, pp. 914–922, https://doi.org/10.1109/COMPSAC.2017.75.
[6] H.U. Gökçe, K.U. Gökçe, Integrated system platform for energy efficient building operations, J. Comput. Civ. Eng. 28 (6) (2014), 05014005, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000288.
[7] J. Li, D. Greenwood, M. Kassem, Blockchain in the built environment and construction industry: a systematic review, conceptual models and practical use cases, Autom. ConStruct. 102 (2019) 288–307, https://doi.org/10.1016/j.autcon.2019.02.005.
[8] M.S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli, M.H. Rehmani, Applications of blockchains in the internet of things: a comprehensive survey, IEEE Commun. Surv. Tutor. 21 (2) (2019) 1676–1717, https://doi.org/10.1109/COMST.2018.2886932.
[9] J. Li, M. Kassem, Applications of distributed ledger technology (DLT) and blockchain-enabled smart contracts in construction, Autom. ConStruct. 132 (2021), 103955, https://doi.org/10.1016/j.autcon.2021.103955.
[10] Z. Zheng, S. Xie, H. Dai, X. Chen, H. Wang, An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends, IEEE 6th Int. Congr. Big Data, 2017, 2017, pp. 557–564, https://doi.org/10.1109/BigDataCongress.2017.85.
[11] S. Huh, S. Cho, S. Kim, Managing iot devices using blockchain platform, 2017, in: 19th Int. Conf. Adv. Commun. Technol., IEEE Computer Society, 2017, pp. 464–467, https://doi.org/10.23919/ICACT.2017.7890132.
[12] Q. Xu, Z. He, Z. Li, M. Xiao, R.S.M. Goh, Y. Li, An effective blockchain-based, decentralized application for smart building system management, in: Real-Time Data Anal. Larg. Scale Sens. Data, Elsevier, 2020, pp. 157–181, https://doi.org/10.1016/B978-0-12-818014-3.00008-5.
[13] O. Van Cutsem, D. Ho Dac, P. Boudou, M. Kayal, Cooperative energy management of a community of smart-buildings: a blockchain approach, Int. J. Electr. Power Energy Syst. 117 (2020), 105643, https://doi.org/10.1016/j.ijepes.2019.105643.

[14] V. Agarwal, S. Pal, Blockchain meets IoT: a scalable architecture for security and maintenance, in: 2020 IEEE 17th Int. Conf. Mob. Ad Hoc Smart Syst. MASS 2020, Institute of Electrical and Electronics Engineers Inc., 2020, pp. 53–61, https://doi.org/10.1109/MASS50613.2020.00017.
[15] Z. Ye, M. Yin, L. Tang, H. Jiang, Cup-of-Water theory: a review on the interaction of BIM, IoT and blockchain during the whole building lifecycle, in: ISARC. Proc. Int. Symp. Autom. Robot. Constr., International Association for Automation and Robotics in Construction, 2018, pp. 1–9, https://doi.org/10.22260/isarc2018/0066.
[16] D. Jia, J. Xin, Z. Wang, W. Guo, G. Wang, Efficient query model for storage capacity scalable blockchain system, J. Softw. 30 (9) (2019) 2655–2670, https://doi.org/10.13328/j.cnki.jos.005774.
[17] J. Shi, R. Li, Survey of blockchain access control in internet of things, J. Softw. 30 (6) (2019) 1632–1648, https://doi.org/10.13328/j.cnki.jos.005740.
[18] W. Tong, X. Dong, Y. Shen, X. Jiang, A Hierarchical Sharding Protocol for Multi-Domain Iot Blockchains, IEEE Int. Conf. Commun., 2019, 2019, pp. 1–6, https://doi.org/10.1109/ICC.2019.8761147.
[19] J. Wang, C. Han, X. Yu, Y. Ren, R. Sherratt, S. Sherratt, Distributed secure storage scheme based on sharding blockchain, Comput. Mater. Continua (CMC) 70 (2022) 4485–4502, https://doi.org/10.32604/cmc.2022.020648.
[20] K.H. El-Ammar, Visualization, Data Sharing and Interoperability Issues in Model-Based Facilities Management Systems, Concordia University, Diss, 2006.
[21] X. Zhang, H. Gao, Optimal performance-based building facility management, Comput. Aided Civ. Infrastruct. Eng. 25 (4) (2010) 269–284, https://doi.org/10.1111/j.1467-8667.2009.00633.x.
[22] D. Minoli, K. Sohraby, B. Occhiogrosso, IoT considerations, requirements, and architectures for smart buildings-energy optimization and next-generation building management systems, IEEE Internet Things J. 4 (1) (2017) 269–283, https://doi.org/10.1109/JIOT.2017.2647881.
[23] E.Z. Tragos, M. Foti, M. Surligas, L. George, P. Stelios, P. Stefanos, A. Vangelis, An IoT based intelligent building management system for ambient assisted living, in: 2015 IEEE Int. Conf. Commun., 2015, pp. 246–252, https://doi.org/10.1109/ICCW.2015.7247186.
[24] A.H. Oti, E. Kurul, F. Cheung, J.H.M. Tah, A framework for the utilization of building management system data in building information models for building design and operation, Autom. ConStruct. 72 (2016) 195–210, https://doi.org/10.1016/j.autcon.2016.08.043.
[25] Y.J. Chen, Y.S. Lai, Y.H. Lin, BIM-based augmented reality inspection and maintenance of fire safety equipment, Autom. ConStruct. 110 (2020), 103041, https://doi.org/10.1016/j.autcon.2019.103041.
[26] J. Wang, G. Wei, X. Dong, A dynamic fire escape path planning method with BIM, J. Ambient Intell. Hum. Comput. 12 (2021) 10253–10265, https://doi.org/10.1007/s12652-020-02794-2.
[27] S. Nakamoto, Bitcoin: a peer-to-peer electronic cash system, https://bitcoin.org/bitcoin.pdf, 2018. (Accessed 22 July 2022). Accessed.
[28] C. Xu, K. Wang, P. Li, S. Guo, J. Luo, B. Ye, M. Guo, Making big data open in edges: a resource-efficient blockchain-based approach, IEEE Trans. Parallel Distr. Syst. 30 (4) (2019) 870–882, https://doi.org/10.1109/TPDS.2018.2871449.
[29] P. Treleaven, R.G. Brown, D. Yang, Blockchain technology in finance, Comput. Times 50 (9) (2017) 14–17, https://doi.org/10.1109/MC.2017.3571047.
[30] C. Lin, D. He, X. Huang, K.K.R. Choo, A.V. Vasilakos, BSeIn: a blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0, J. Netw. Comput. Appl. 116 (2018) 42–52, https://doi.org/10.1016/j.jnca.2018.05.005.
[31] M. Madine, K. Salah, R. Jayaraman, Y. Al-Hammadi, J. Arshad, I. Yaqoob, AppxChain: application-level interoperability for blockchain networks, IEEE Access 9 (2021) 87777–87791, https://doi.org/10.1109/ACCESS.2021.3089603.
[32] K. Siountri, E. Skondras, D.D. Vergados, Developing smart buildings using blockchain, internet of things, and building information modeling, Int. J. Interdiscip. Telecommun. Netw. (IJITN) 12 (3) (2020) 1–15, https://doi.org/10.4018/IJITN.2020070101.
[33] N. Koseleva, G. Ropaite, Big data in building energy efficiency: understanding of big data and main challenges, in: Procedia Eng., Elsevier Ltd, 2017, pp. 544–549, https://doi.org/10.1016/j.proeng.2017.02.064.
[34] H. Huang, W. Kong, X. Peng, Z. Zheng, Survey on blockchain sharding technology, Comput. Eng. 48 (6) (2022) 1–10, https://doi.org/10.19678/j.issn.1000-3428.0063887.
[35] C. Zhang, Y. Zhang, X. Li, T. Nie, G. Yu, Survey of new blockchain techniques: DAG based blockchain and sharding based blockchain, Comput. Sci. 47 (10) (2020) 282–289, https://doi.org/10.11896/jsjkx.191000057.
[36] Q. Zhou, H. Huang, Z. Zheng, J. Bian, Solutions to scalability of blockchain: a survey, IEEE Access 8 (2020) 16440–16455, https://doi.org/10.1109/ACCESS.2020.2967218.
[37] Y. Jiang, C. Wang, Y. Wang, L. Gao, A cross-chain solution to integrating multiple blockchains for IoT data management, Sensors 19 (9) (2019), https://doi.org/10.3390/s19092042.
[38] I.S., Building Automation and Control Systems (BACS)—part 3: Functions, 16484-3, 2005, https://www.iso.org/standard/37205.html. (Accessed 24 May 2022). Accessed.
[39] Ethereum 2. Ethereum, 0 Phase 0-the Beacon Chain, 2020. https://github.com/ethereum/consensus-specs/blob/dev/specs/phase0/beacon-chain.md. (Accessed 5 May 2022). Accessed.
[40] M. Li, Y. Qin, Scaling the blockchain-based access control framework for IoT via sharding, in: IEEE Int. Conf. Commun., IEEE, 2021, pp. 1–6, https://doi.org/10.1109/ICC42927.2021.9500403.
[41] F. Li, Z.R. Li, H. Zhao, Research on the progress in cross-chain technology of blockchains, J. Softw. 30 (6) (2019) 1649–1660, https://doi.org/10.13328/j.cnki.jos.005741.
[42] D. Ongaro, J. Ousterhout, Search of an Understandable Consensus Algorithm, USENIX Annu. Tech. Conf., 2014, 2014, pp. 305–319.
[43] Y.E. Oktian, S.G. Lee, H.J. Lee, Hierarchical multi-blockchain architecture for scalable internet of things environment, Electron. Switz. 9 (6) (2020) 1–27, https://doi.org/10.3390/electronics9061050.
[44] GB 50352-2019, Uniform Standard for Design of Civil Buildings, 2019. https://www.mohurd.gov.cn/gongkai/fdzdgknr/tzgg/201905/20190530_240715.html. (Accessed 24 May 2022). Accessed.
[45] N. Kullig, P. Lämmel, N. Tcholtchev, Prototype implementation and evaluation of a blockchain component on IoT devices, in: Procedia Comput. Sci., Elsevier, 2020, pp. 379–386, https://doi.org/10.1016/j.procs.2020.07.054.
[46] M. Collina, mcollina/autocannon, https://rb.gy/snantd, 2016. (Accessed 23 May 2022). Accessed.
[47] GB 50339-2013, Code for Acceptance of Quality of Intelligent Building Systems, 2013. https://www.mohurd.gov.cn/gongkai/fdzdgknr/tzgg/201306/20130628_224775.html. (Accessed 24 December 2021). Accessed.
[48] S. Samonas, D. Coss, The CIA strikes back: redefining confidentiality, integrity and availability in security, J. Inf. Syst. Secur. 10 (3) (2014) 21–45.