applied sciences

*Article*

# A Blockchain-Based Supervision Data Security Sharing Framework

**Jiu Yong [1,2,*], Xiaomei Lei [2], Zixin Huang [1], Jianwu Dang [1] and Yangping Wang [1]**

[1] The School of Electronic and Information Engineering, Lanzhou Jiaotong Univeristy, Lanzhou 730070, China; 13519311970@163.com (Y.W.)
[2] College of Intelligence and Computing, Tianjin University, Tianjin 300072, China; leixiaomei@tju.edu.cn
* Correspondence: yongjiu@mail.lzjtu.cn; Tel.: +86-139-9310-0018

**Abstract:** Ensuring trust, security, and privacy among all participating parties in the process of sharing supervision data is crucial for engineering quality and safety. However, the current centralized architecture platforms that are commonly used for engineering supervision data have problems such as low data sharing and high centralization. A blockchain-based framework for the secure sharing of engineering supervision data is proposed by utilizing the tamper-proof, decentralized, and traceable characteristics of blockchain. The secure storage of supervision data is achieved by combining it with the IPFS (InterPlanetary File System), reducing the storage pressure of on-chain data. Additionally, a fast data retrieval framework is designed based on the storage characteristics of supervision data. Then, CP-ABE (Ciphertext Policy Attribute Based Encryption) is combined with a data storage framework to ensure the privacy, security, and reliability of supervisory data during the sharing process. Finally, smart contracts are designed under the designed framework to ensure the automatic and trustworthy execution of access control processes. The analysis and evaluation results of the security, encryption and decryption, and cost performance of the proposed blockchain framework show that the encryption and decryption time is completed within 0.1 s, the Gas cost is within the normal consumption range, and the time cost of smart contract invocation does not exceed 5 s, demonstrating good availability and reusability of the method proposed in this article.

**Keywords:** blockchain; engineering supervision; smart contract; secure sharing; CP-ABE

check for updates

**Citation:** Yong, J.; Lei, X.; Huang, Z.; Dang, J.; Wang, Y. A Blockchain-Based Supervision Data Security Sharing Framework. *Appl. Sci.* **2024**, *14*, 7034. https://doi.org/10.3390/app14167034

Academic Editor: Juan-Carlos Cano

Received: 8 July 2024
Revised: 2 August 2024
Accepted: 9 August 2024
Published: 10 August 2024

## 1. Introduction

With the rapid development of engineering construction toward informatization and intelligence, there is a large quantity, large volume, and complex composition of various participating systems, which requires high requirements for the reliability and accessibility of engineering supervision data, making it more difficult to control the quality of engineering construction during the engineering supervision process. More specialized and standardized digital applications in the supervision industry are needed to provide guarantees for engineering quality and safety. Therefore, building a trustworthy and secure engineering supervision data-sharing platform can improve the efficiency of engineering supervision work, which is of great significance for the openness, visualization, and transparency of supervision work.

Engineering supervision data are a true reflection of the implementation of the supervision process and an important basis for ensuring project quality and assigning safety responsibilities. Therefore, improving the sharing efficiency of engineering supervision data while ensuring data security has become an urgent problem to be solved [1]. However, in the traditional process of engineering supervision data management, on the one hand, the widely used centralized databases are vulnerable to attacks and tampering. For example, hackers invaded the Aadhaar database in India and leaked information of over 1.1 billion Indian citizens in 2018, and hackers sold 700 million users' data of LinkedIn on the dark web in 2021, indicating that integrated supervision platforms have data security risks. On the other hand, due to the scattered organizational structure of various participating parties in the project, supervision record data are not synchronized with the actual situation on site, resulting in the inability of all participating parties in the project construction to reach a consensus on the storage data, making it difficult to achieve complete trust and sharing of data. Therefore, with the expansion of project scale and the complexity of the construction environment, it is necessary to build a safer and more reliable engineering supervision data-sharing platform to provide support for more efficient and standardized information management of engineering supervision data [2].

With the increasing difficulty of information management of engineering supervision data, higher requirements have been placed on the reliability and accessibility of engineering supervision data. Existing centralized data governance and sharing application platforms are prone to becoming targets of hacker attacks due to the centralized storage characteristics, leading to serious consequences such as data leakage, system tampering, or paralysis, making it difficult to meet the actual needs of engineering construction supervision [3]. The immutability, decentralization, and traceability of blockchain are highly compatible with the requirements of trustworthy and secure management of engineering supervision data [4]. However, the composition of participating departments is complex, and the amount of engineering supervision data information is large. It is difficult to directly optimize the engineering supervision data-sharing platform using blockchain. On the one hand, the scalability and performance defects of consensus algorithms, as well as the storage pressure of blockchain data, will become the main bottleneck of technology integration [5]. On the other hand, due to the open and transparent characteristics of blockchain, data privacy needs to be guaranteed in data sharing [6]. Therefore, there is an urgent need to design a new blockchain-based framework for secure sharing of engineering supervision data, in order to achieve secure sharing applications and trustworthy confidentiality governance of engineering supervision data.

This article proposes a blockchain-based framework for the secure sharing of engineering supervision data, which improves the efficiency of blockchain data storage and system scalability by combining with the IPFS (InterPlanetary File System), achieving secure storage of large-scale engineering supervision data. Then, the smart contract and ciphertext policy attributes are combined with encryption, so that only authorized users can access engineering supervision data without the need for third-party participation, achieving fine-grained access control of engineering supervision data, thereby ensuring the security of engineering supervision data and avoiding the leakage of private data. Finally, the security, encryption and decryption, and cost performance of the proposed framework are analyzed and evaluated in this article. The main contributions of this article are as follows:

(1) The existing engineering construction projects have the characteristics of large quantity, large scale, and complex composition of various participating systems, which increases the difficulty of controlling the quality of engineering construction. This article designs a blockchain-based engineering supervision data security sharing framework to solve the problems of low data sharing, high centralization, and data privacy and security in the engineering supervision information system.

(2) The existing blockchain data-sharing system suffers from a large amount of data information and transaction processing. This article combines the IPFS to achieve secure storage of large-scale supervision data, and designs a data fast retrieval framework based on the storage characteristics of supervision data to ensure fast sharing of the entire process of data storage, reading, and execution on the massive engineering supervision chain.

(3) There are trust and data security issues among the participating parties in the current application of information management for engineering supervision data. Based on blockchain technology, this article designs a smart contract for the sharing application of supervision data, utilizing the mandatory execution and tamper-proof characteris-

tics of smart contracts to ensure the transparency, traceability, and tamper-resistance of supervision data, achieving trustworthy governance of engineering supervision data.

The remaining work of this article is as follows: Section 2 analyzes the research progress on secure sharing and application of blockchain data. Section 3 proposes a framework structure for the secure sharing of blockchain engineering supervision data. Section 4 evaluates and analyzes the performance of the proposed framework in this article. Section 5 delves into the usability of the framework proposed in this article and explores the application architecture based on the framework. Section 6 summarizes the entire text.

## 2. Related Work

As the underlying technology of Bitcoin, blockchain is essentially a technical solution that involves centralized maintenance of a reliable database in a distributed environment. It adopts a blockchain data structure for data storage, and smart contracts to program and operate data. With the development of blockchain technology, it has been applied in complex scenarios. Public blockchain, consortium blockchain, and private blockchain have emerged depending on the number of nodes, degree of decentralization, and security requirements of blockchain [7]. As shown in Table 1, the degree of decentralization in public chains is the highest, but public chains suffer from low transaction efficiency and poor privacy. A private blockchain is created by an organization or unit, so that the access to reading and writing data and the addition of new nodes are controlled by the organization or unit. However, the eligibility to participate in nodes is strictly limited and there are fewer participating nodes on private blockchain. There are only a small number of nodes on the consortium blockchain with a high degree of trust. Transactions do not require confirmation from all network nodes, and have the characteristics of multiple organizations participating in management, good privacy protection, low transaction costs, and fast transaction speed. Therefore, the transaction speed in consortium blockchain is faster than any other blockchain. Moreover, because the permission is determined by the organization, their own privacy protection is better, making the adoption of consortium blockchain solutions more optimal in data security sharing and trusted governance in the engineering supervision industry [8]. In the blockchain application framework, Zhou et al. [9] proposed a full lifecycle data transaction integration blockchain framework with trust and dispute resolution; Kumar et al. [10] utilized blockchain and machine learning to provide a trustworthy privacy protection security framework for sustainable smart cities; Makhdoom et al. [11] proposed a blockchain-based privacy protection and secure data-sharing framework in smart cities; Quan et al. [12] proposed a trusted medical data-sharing framework that uses blockchain and outsourcing computing to achieve edge computing; RK et al. [13] proposed a blockchain-driven framework with attribute aware encryption for enhancing cloud communication security; Wei et al. [14] proposed a blockchain data access control framework for secure data sharing in the Internet of Things. Therefore, with the development of blockchain technology, blockchain is no longer bound to cryptocurrencies. And the scalability of blockchain has become a major research issue, which is mainly divided into two aspects: sharing and storage. The research on the scalability of blockchain from the perspective of sharing mainly solves the problems of long transaction confirmation delay and slow transaction speed in blockchain. The research on the scalability of blockchain from the storage level is aimed at reducing the storage cost of blockchain through a data storage-based scalability solution.

**Table 1.** Types and characteristics of blockchain.

| Projects | Public Chain | Private Chain | Consortium Chain |
|---|---|---|---|
| Participants | Anyone | Internal company | Chain members |
| Consensus Algorithm | PoW/PoS/DPoS | Distributed consistency | Distributed consistency |
| Accounting | All participants | User-defined | Members consultation |
| Excitation Mechanism | Needed | Optional | Optional |
| Degree of Centralization | Decentralization | Centralization (Multi-centralization) | Multi-centralization |
| Characteristic | Credit self establishment | Data transparency and traceability | Efficiency and low cost |
| Throughput | 3–20 times/second | 1000–200,000 times/second | 1000–20,000 times/second |
| Typical scenarios | Cryptocurrency | Audit and issuance | Payment, settlement, and public welfare |

The key to the immutability of blockchain data is distributed ledger technology and consensus algorithms. According to the different selection methods of accounting nodes, consensus algorithms can be divided into PoW (Proof of Work), PoS (Proof of Stake), DPoS (Delegated Proof of Stake), and PBFT (Practical Byzantine Fault Tolerance) algorithms. The characteristics and performance comparison of four consensus algorithms are shown in Table 2. With the consensus on the security of blockchain itself and the launch of third-generation blockchain technology represented by EOS, the transaction processing speed of blockchain can support most applications, and the application research of blockchain technology in various fields has received attention from all parties. For the study of data-sharing solutions using blockchain, Banik et al. [15] proposed a blockchain public key encryption and keyword search method for traditional Chinese medicine data sharing in cloud environments; Eltayieb et al. [16] proposed an attribute signature scheme based on blockchain for cloud data sharing; Guo et al. [17] studied the fine-grained privacy protection of policy fuzzy matching in blockchain based mobile crowd perception; Jia et al. [18] proposed a blockchain data security sharing protocol based on the threshold Paillier algorithm; Li et al. [19] proposed a blockchain based privacy protection and security sharing scheme for flight operation data; López Sorribes et al. [20] proposed a blockchain based record and interoperability network; Ma et al. [21] studied trusted data sharing based on blockchain flexible access control; Qin et al. [7] proposed a blockchain based multi-attribute permission access control scheme; Singh et al. [22] studied chaos and Paillier secure image data sharing based on blockchain and cloud security; Wan et al. [23] conducted research on privacy protection in federated learning; Wang et al. [24] proposed a blockchain data-sharing scheme to improve the security of the Internet of Vehicles; Wang et al. [25] researched health data security sharing based on hybrid blockchain. In addition, Agyekum et al. [26] proposed a blockchain-based proxy re-encryption method for secure data sharing in the Internet of Things; Cao et al. [27] studied a blockchain-based cross-domain data security sharing for the Internet of Things; Deng et al. [28] proposed a blockchain-assisted threshold cryptography method for key security management in power Internet of Things data sharing; Li et al. [29] studied privacy protection and rewards for the Internet of Things based on blockchain technology; Lu et al. [30] studied the privacy protection data sharing of blockchain and federated learning in the industrial Internet of Things; Zhou et al. [31] implemented secure and trustworthy federated data sharing in IIoT based on blockchain. Therefore, it can be seen that a large number of scholars have made varying degrees of contributions to data resource sharing and access control, and have conducted research on the scalability framework of blockchain, exploring how to apply blockchain technology in fields such as data sharing, industry regulation, trust transfer, and quality tracing [32]. However, these solutions only provide certain privacy protection during the data-sharing process, and shared resource storage also has shortcomings in terms of security and cost issues, which are not suitable for complex application environments of engineering supervision data sharing.

**Table 2.** Performance comparison of mainstream consensus algorithms in blockchain.

| Consensus Algorithm | PoW | PoS | DPoS | PBFT |
|---|---|---|---|---|
| Transaction Processing Per Second (TPS) | 7 | 300 | 500 | 1000 |
| Block time | 10 min | 1 min | 8 s | 1 s |
| Degree of decentralization | completely | completely | partially | partially |
| Resource consumption | high | higher | general | low |
| Maximum number of faulty nodes | $2\lambda + 1 \leq \mathrm{N}$ | $2\lambda + 1 \leq \mathrm{N}$ | $3\lambda + 1 \leq \mathrm{N}$ | $3\lambda + 1 \leq \mathrm{N}$ |
| Needed cryptocurrency incentives or no | yes | yes | yes | no |
| Selection of accounting nodes | network-wide competition | network-wide competition | voting commission | random rotation |
| Complexity | $O(n)$ | $O(n)$ | $O(n^2)$ | $O(n^2)$ |
| Applicable scenarios | public chain | public chain | public chain | private chain and consortium chain |

In terms of research on the application of blockchain engineering supervision, Perera et al. [4] conducted a use case and literature analysis on the potential possibilities of applying blockchain to the construction industry. They believe that the various advantages of blockchain technology will help improve the operation mode of the traditional construction industry, and the application of blockchain in the construction industry has great potential; Qian et al. [33] studied how the introduction of blockchain in engineering project supply chain management can improve trust from different sources and dimensions in the management process, and explained the impact of blockchain technology on trust; Sheng et al. [34] proposed a blockchain-based quality information management model, which ensures consistency and transparency in quality information management and avoids disputes among stakeholders. Liu et al. [35] studied blockchain-based file prediction cloud storage shared data integrity auditing. Therefore, it can be seen that existing researchers have studied the application of blockchain in the field of data sharing from different scenarios by adopting different technologies, and some theoretical studies have applied blockchain to engineering project management. However, current engineering supervision data are generally shared through centralized storage platforms, which often leads to changes in engineering supervision data due to subjective and non-subjective factors. This makes it urgent to have a more efficient and reliable blockchain system to provide guarantees for engineering supervision data management.

In summary, existing research has explored the application of blockchain technology in the field of data security sharing from different application scenarios. However, for the sharing application and security governance needs of engineering supervision data, it is necessary to construct a blockchain application framework from the aspects of data security, data interaction efficiency, and functional implementation [2], in order to achieve data exchange among participating nodes of the blockchain and effectively solve the problems of access barriers and information exchange difficulties between nodes. In addition, the immutable features of blockchain can ensure the security of engineering supervision data storage. As long as the data are verified and recorded in the blockchain, intelligent contracts are introduced to achieve unified management and control of the data-sharing process and improve the efficiency of engineering supervision data sharing. Ultimately, blockchain-based engineering supervision can be achieved to secure sharing and trustworthy governance of data.

## 3. Blockchain-Based Framework for Secure Sharing of Engineering Supervision Data

A blockchain-based engineering supervision data security sharing framework is designed to address the high pressure of on-chain data storage, data privacy, and security issues in the process of supervision data sharing. As shown in Figure 1, the framework mainly consists of several parts: user end, supervision data-sharing consortium chain, and IPFS cluster. This framework utilizes the immutability, decentralization, and traceability characteristics of blockchain, which utilizes smart contract design and data on-chain storage to achieve decentralized sharing of supervision data. And a fast file retrieval scheme is designed based on the storage characteristics of supervision data. Finally, the proposed sharing framework and smart contract mechanism are combined to solve the problem of difficult information exchange between the supervision party and all participating parties due to high trust costs, effectively improving the security and sharing efficiency of engineering supervision data.

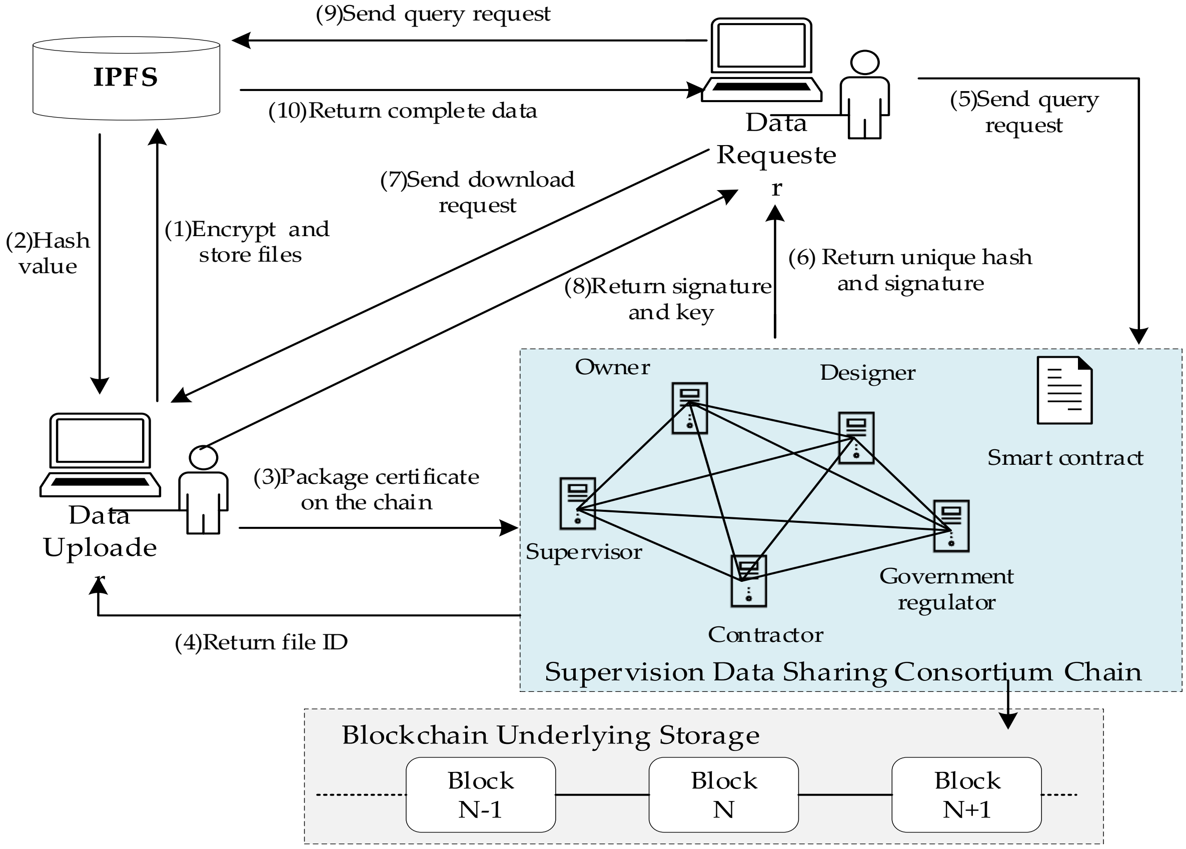

**Figure 1.** Blockchain-based framework for secure sharing of engineering supervision data.

(1) Client

After identity authentication, registered users initiate transaction requests from the client to the supervision data-sharing consortium blockchain. Before uploading to the IPFS, the original file is encrypted. To ensure the traceability of the file source, users need to upload a private key signature. Finally, a fully deployed smart contract is used to store and query supervision data.

(2) Supervision data-sharing consortium blockchain

The supervision data-sharing consortium blockchain stores supervision data metadata, responding to user upload and query requests, mainly including node networks, consensus mechanisms, and smart contracts. The node network consists of the owner, supervisor, designer, constructor, and government regulator participating in the engineering construction. The supervisor in the engineering supervision department plays the role of a system administrator, mainly including creating and deploying blockchain networks, managing nodes and ledgers on the chain, registering user identities, and the issuing and managing of certificates and keys. This framework uses the PBFT algorithm to confirm transactions [36]. Nodes participating in the consensus can query data and execute the consensus algorithm, while nodes not participating in the consensus are responsible for supervising the consensus and synchronizing the consensus results to their respective distributed ledgers. Smart contracts are deployed on the blockchain and can be automatically called and perform preset response actions, ensuring consistency and transparency in the data flow process through smart contracts.

(3) IPFS cluster

As a peer-to-peer distributed file system, the IPFS has high security and storage capabilities. In this framework, it is used for the off-chain storage of encrypted file data. Users can search for corresponding files in the IPFS by uniquely identifying the hash address of the file.

### *3.1. Rapid Retrieval Mode for Engineering Supervision Data*

Due to the business communication between the supervision party and various construction departments, a large amount of data will be generated, and all data corresponding to the same file number will be stored in a dispersed database. If traditional data retrieval frameworks are used, database management software will screen full-text data based on file *ID* as the retrieval criteria to obtain target data. The data retrieval time is longer due to the high time complexity and excessive interaction with the hard disk. And as the amount of data increases, the efficiency of data retrieval will further decrease. The nodes on the proposed framework chain will maintain a hash table to record the position of the supervision data summary in the consortium blockchain, and achieve fast retrieval by locating the block where the data are located.

(1) For file data scattered in the database, all data related to the file can be quickly found if the block positions of all data corresponding to the file can be obtained. An index data structure is established for file *ID* and block number based on the above data retrieval characteristics. There are two variables in this data structure: one is used to record the file *ID*, and the other is a collection class used to store the block numbers of all database records related to the file.

(2) In the traditional framework for fast data retrieval, node *ID,* and block number information are maintained by creating a new B+tree index structure. If querying file data file *ID* with 01, it is inefficient to retrieve information from the database according to the retrieved block number.

(3) By establishing a hash index with the file *ID* as the primary key, the encapsulation class of the file *ID* and block number is stored in the hash table. Firstly, all block numbers related to the file *ID* with 01 are searched in the hash table in memory, and then file data in the database are searched based on the set of block numbers. Compared to the solution framework of creating a new B+tree index, this framework can reduce disk I/O times by half. According to the storage characteristics of hash tables, compared to I/O interaction, the search time of hash tables can be ignored and mainly determined by the number of disk and memory interactions, which can effectively improve retrieval efficiency.

### *3.2. Engineering Supervision Data-Sharing Framework Based on Consortium Blockchain*

Due to the large amount of engineering supervision data, the existing blockchain system cannot meet the data storage needs of the engineering supervision department. This framework combines the IPFS to achieve distributed storage of large-scale supervision data, solving the problem of insufficient storage space on the blockchain. Efficient, transparent, and secure control of data access processes between nodes is achieved by combining CP-

ABE (Ciphertext Policy Attribute Based Encryption) with a data-sharing framework, while ensuring data privacy and security.

### 3.2.1. Definition Form of Access Policy

In order to achieve fine-grained access control, it can be divided into a key policy attribute-based encryption and CP-ABE according to different decryption positions. This article adopts the CP-ABE encryption algorithm based on ciphertext policy attributes. Data can be encrypted and assigned an access control structure, and the holders of which attributes can access the data are defined based on this structure. Each user has a set of attributes that can be used to determine who has access to the data, and only users whose attributes match the ciphertext access structure can decrypt the ciphertext. Under normal circumstances, CP-ABE mainly uses the following four algorithms:

(1) Initialization algorithm

$$\text{Setup } (\lambda) \rightarrow (PK, MK)$$

Initialization parameters $\lambda$ is input to generate public parameter $PK$ and master key $MK$, completing the initialization process, where $\lambda$ is the security parameter. The Setup $(\lambda)$ takes the number of attributes $U$ in the system as input. It then selects a group of prime order $p$, $a$ generator $g$, and $U$ random group elements $h_1, \ldots, h_U$ related to system properties. And choose a random exponent $\alpha$ and $a$. The generated public key is:

$$PK = g, e(g,g)^{\alpha}, g^{a}, h_1, \ldots, h_U \tag{1}$$

where $MK = g^{\alpha}$ is set as the master key.

(2) Key generation algorithm

$$\text{KeyGen } (MK, S) \rightarrow SK$$

The master key $MK$ and attribute set $S$ are entered to generate the attribute private key $SK$ for the user. A random number $t$ is selected to create a private key $SK = g^{\alpha} g^{at}$ for user decryption.

(3) Encryption algorithm

$$\text{Encrypt } (PK,\ M,\ T(A,\rho)) \rightarrow CT$$

Public parameter $PK$ and access structure $T(A,\rho)$ are used to encrypt plaintext $M$ into ciphertext $CT$. The access structure $T(A,\rho)$ uses LSSS (Linear Secret Sharing Scheme) to associate the rows and attributes of plaintext message $M$, where $A$ is the $l \times n$ matrix, and $\rho$ represents the mapping function that maps each row of $A$ to the corresponding attribute. The secret value $s$ is divided into $l$ parts for encryption, and the final ciphertext obtained is:

$$CT = (C = M \cdot e(g,g)^{\alpha s}, C' = g^{s}, (C_1 = g^{a\lambda_1} h_{\rho(1)}^{-r_1}, D_1 = g^{r_1})), \ldots, (C_l = g^{a\lambda_l} h_{\rho(l)}^{-r_n}, D_l = g^{r_l}) \tag{2}$$

where $\lambda_i = A_i \cdot \vec{v}$, $\vec{v}$ is a random vector containing the secret value $s$, $r_1, \ldots, r_l$ is a random number belonging to the group, and $h_{\rho(1)}$ is corresponding to the random value of the first attribute involved in the access policy.

(4) Decryption algorithm

$$\text{Decrypt } (CT, SK, PK) \rightarrow M$$

Public parameters $PK$ and private key $SK$ are used to decrypt the ciphertext $CT$ into plaintext $M$. Only when the attribute set $S$ associated with $SK$ satisfies the access policy $(A,\rho)$ related to ciphertext $CT$ will decryption be successful and output plaintext $M$. The definition form of access policy based on CP-ABE is shown in Table 3. The data access control strategy is defined by the initiator of the contract, combined with the application background of supervision data sharing. In this article, attributes are defined as position permission, user department, and registration time period. The supervision department defines the access control strategy and achieves more refined access control by combining these three attributes. The defined access strategy is as follows: users who meet the specified department attributes, while also meeting both permission level attributes and registration time period attributes, can access plaintext. Secure sharing and privacy protection of supervision data can be achieved by designing access control policies and using encryption algorithms based on CP-ABE to generate keys and access policies.

**Table 3.** Access policy definition.

| User Attribute | Operator | Attribute Value | Remark |
|---|---|---|---|
| User position permission | ≥ | Supervisor, Engineer, Department head, Documenter | Only allow users with permissions not lower than specified level to access |
| User department | in | Supervision party, Construction party, Government department | Only allow users located in the department to access |
| Registration time period | between | The start and end dates | Only allow users registered within time period to access |

### 3.2.2. Supervision Data-Sharing Protocol

(1) Key generation stage

Users register with the CA (Certificate Authority) organization, interact with the CA organization and the consortium network through the client, and use the initialization algorithm setup ($\lambda$) to generate the master key *MK* and public parameter *PK* in the CP-ABE framework based on the given security parameters. The user sends a registration application to the supervision department, which verifies the user's identity. After verification, the public and private key pairs *Upk* and *Usk* are sent to the user. Then, the corresponding digital certificate *Ucert* attribute set *Su* is generated based on the registration request sent by the user, and the key distribution algorithm KeyGen (*MK*, *SU*) is run to generate the user attribute private key *U'sk*. Finally, the supervision department will send the public key *Upk*, private key *Usk*, digital certificate *Ucert*, and attribute private key *U'sk* to the user for safekeeping through a secure channel.

(2) Data encryption storage stage

In the process of sharing engineering supervision data, larger-scale supervision data original files will be stored in the IPFS file system. After storage is completed, the IPFS file system returns the hash address *CID* (Content Identifier) that uniquely identifies the file to the client. Users can use *CID* to query the stored files in the IPFS file system. For the supervision data stored in the IPFS file system, it is necessary to encrypt the data and use the CP-ABE algorithm to encrypt the symmetric key before uploading to the system. Firstly, the data uploader specifies the encryption strategy *Pa* and the supervision data to be stored, then selects the key *K* and uses a symmetric encryption algorithm to encrypt the supervision data, obtains the ciphertext, and uploads the ciphertext to the IPFS file system. Then, encrypt the symmetric key *K* and use the encrypt algorithm to encrypt the key *K* based on the policy, obtaining $Enck(K) = Encrypt(PK, K, Pa)$, where *PK* is a common parameter in the CP-ABE framework and *Pa* is the encryption policy. Finally, the uploader constructs the supervision data metadata *Tx* and uploads the file address *CID* returned by the IPFS file system to the blockchain along with the user's department, file name, timestamp, and digital signature.

(3) Data-sharing stage

Firstly, the requester sends a file download request to the file uploader who sends the private key signature and encrypts key $K'$ to the requester after verifying the identity of the requester. The visitor requests metadata information from the consortium blockchain network through the file *ID*, and obtains encrypted files from the IPFS file system using the file address *CID* in the metadata information. The requester takes the attribute key ciphertext $Enck(K)$ and attribute private key $U'sk$ as inputs. When the private key attribute satisfies the access control policy $Pa$ contained in the ciphertext, decryption obtains plaintext $K$, that is, decryption obtains symmetric key $K = Decrypt\ (Enck(K),\ U'sk)$. Finally, the ciphertext is decrypted using key $K$ to obtain the original file data, achieving blockchain data access control based on the CP-ABE framework.

### *3.3. Design of Smart Contract for Engineering Supervision Data Sharing*

In response to the problems of a long sharing process, low efficiency, inconsistent technical standards, and easy data tampering in the sharing of supervision data, this article utilizes the automatic mandatory execution feature of smart contracts to perform heuristic operation management on supervision data. Deploying smart contracts on blockchain networks to achieve access control during data sharing through preset conditions in the contracts, avoiding third-party intervention in the data-sharing process, and improving the efficiency of supervision data sharing. The smart contract used is written in Solidity language, including supervision data upload contract, metadata upload contract, metadata query contract, and IPFS file download contract.

#### 3.3.1. Design of Smart Contract Structure

(1) Design of access policy structure

The access policy structure is used to define access control policies, including attributes such as allowed user roles, project types, levels, regions, time periods, etc. The design of the access policy structure is shown in Table 4.

**Table 4.** Access policy structure.

| Attribute | Data Type | Description |
|---|---|---|
| Role | uint8 | User role |
| Location | uint8 | User department |
| Start Time | uint256 | Start time |
| End Time | uint256 | End time |

(2) Key structure design

The key structure is used to store the generated encryption key, including access policies and encryption parameters. The key structure design is shown in Table 5.

**Table 5.** Key structure.

| Attribute | Data Type | Description |
|---|---|---|
| Attrs | bytes32 | Access policy attribute list |
| Key | bytes | Encryption key |
| Iv | bytes | Key initialization vector |

(3) Design of ciphertext structure

The ciphertext structure is used to store encrypted data and related information, including encrypted ciphertext and key *ID*. The design of the ciphertext structure is shown in Table 6.

**Table 6.** Ciphertext structure.

| Attribute | Data Type | Description |
|---|---|---|
| Key *ID* | bytes32 | Key *ID* |
| Ciphertext | bytes | Encrypted ciphertext |

(4) Design of contract resource collection structure

The set of contract resources is the set of supervision data, and the design of the structure of contract resources set is the design of the supervision data structure. The supervision data in this system mainly include engineering management official documents, engineering technical materials, and document receipts. The design of the supervision data structure mainly includes file *ID*, file name, file *CID* value, file creation time, uploader unit name, and signature information. The design of the supervision data structure is shown in Table 7.

**Table 7.** Supervision data structure.

| Attribute | Data Type | Description |
|---|---|---|
| File *ID* | unit32 | File *ID* |
| File Name | string | File name |
| File Hash | string | File *CID* value |
| Create Time | timestamp | File creation time |
| Owner | string | Uploader's unit name |
| Audit Option | bool | Document review comments |
| Signature | string | File signature information |

(5) Structural design of contract participants

The contract participants are the participating users in the execution of smart contracts, including users who provide data and users who use data. The structural design of the participating users in contract execution includes the user's address in the blockchain network, file *ID*, supervision data hash value stored on the blockchain, public key for encrypting files, and private key for signature operations. The design of the contract structure is shown in Table 8.

**Table 8.** Contract structure.

| Attribute | Data Type | Description |
|---|---|---|
| Address | address | User's address in blockchain |
| File *ID* | unit32 | File *ID* |
| FileHash | string | File *CID* value |
| PublicKey | string | Public Key |
| SecretKey | string | Private key |

### 3.3.2. Design of Smart Contract Algorithm

(1) Design of contract algorithm for uploading supervision data

The user uploads the supervision data contract and uploads the original supervision data file to the IPFS. Firstly, construct supervision data, which includes the user public key, construction time of supervision data, encrypted file data, and digital signature of the uploader. The data structure for constructing supervision data $D$ is:

$$D = \{PK, Ts, E(Dlog + Ts\|K), Sign(H(E(Dlog + Ts\|K))\|SK\} \quad (3)$$

where $PK$ is the user's public key; $Ts$ is the time when users construct supervision data, and the data construction time is represented using Unix timestamps; $E(Dlog + Ts\|K)$ represents encrypting the original file $Dlog$ using the key $K$; $Sign(H(E(Dlog + Ts\|K))\|SK$ means that the uploader uses their own private key $SK$ to digitally sign the encrypted file, in order to confirm the source of the data. After receiving the uploaded supervision data, the IPFS node uses the uploader's public key for signature verification. If the verification is successful, it indicates that the uploaded file data source is consistent with the user's public key source in the supervision data, and the system allows uploading. According to the CP-ABE algorithm, the attribute key $Enck(K)$ is generated, and the file data are stored in the IPFS. The unique hash address $CID$ that identifies the file is returned to the client. If the verification fails, the upload will be abandoned and the upload failure message is returned to the client. The algorithm for uploading supervision data is shown in Algorithm 1.

**Algorithm 1**: Uploading Supervision Data

```
Input: Encrypt file data $E(Dlog + Ts\|K)$, digital signature Sign, user public key $Upk$, file generation timestamp $Ts$
Output: Success or Failed
$Ek\ (Dlog) = Ecnrypt\ (Dlog, Ts, K)$  //Encrypt the original file
address = pkToAddress ($Upk$)//Convert public key to address type
flag = SignVerify ($PK$, $Sign\ (H(E(Dlog + Ts\|K))\|SK)$  //IPFS node verifies the received file information
If flag! = nil {   //Evaluate the validation results
    return Failed   //Signature verification failed with error message returned
}
file_hash = uploadToIPFS ($Ek(Dlog + Ts\|K)$)  //Upload the ciphertext information of the original data to IPFS
$Enck(K) = Ecnrypt\ (K, Pa, PK)$  //Generate access policy key $Enck(K)$ based on CP-ABE algorithm
ipfs_hash_$Enck(K)$ = uploadToIPFS ($Enck(K)$)  //Upload the ciphertext information of the attribute key to IPFS
return Success  //Complete the upload operation
```

(2) Algorithm design for metadata upload contract

Users upload metadata to the blockchain by uploading contracts through metadata. Firstly, construct the supervision data metadata $Tx$, which mainly includes the following: uploading the user's department, file name, completion timestamp of supervision data construction, hash address that uniquely identifies the file, and digital signature. Specifically:

$$Tx = \{N, PK, TS, CID, Sign(CID + Ts\|SK)\} \tag{4}$$

where $N$ refers to the file name; $PK$ is the upload of the user public key; $TS$ is the timestamp of the completion of supervision data construction; $CID$ refers to the hash address that uniquely identifies the file returned by the IPFS file system; $Sign(CID + Ts\|SK)$ refers to the user using a private key to digitally sign and verify the authenticity of uploaded data on a file hash address. After sending the metadata to the blockchain node, the signature information in the metadata is first verified. If the verification is successful, send a query request to the IPFS node, compare whether the public key of the signer in data $D$ and $Tx$ is the same, and ensure that the identity of the supervisor data uploader and metadata uploader is consistent. If consistent, send the metadata to the blockchain nodes for broadcasting and packaging to the blockchain system. If there is inconsistency, return the uplink failure message to the client. The metadata upload contract algorithm can be found in Algorithm 2.

**Algorithm 2**: Metadata Upload

```
Input: File name $N$, user public key $Upk$, file $CID$, timestamp $Ts$, digital signature $Sign$
Output: Success or Failed
1. requestMsg = GenerateUploadRequest ($N$, $Upk$, $Ts$, $CID$, $Sign$) //Construct the request body for uploading data hash to the blockchain
2. address = pkToAddress ($Upk$) //Convert public key to address type
3. flag = SignVerify (requestMsg. $Upk$, requestMsg.($CID + Ts \| SK$)//Verify the signature information in the request body
4. If flag! = nil{
    5. return failed//Signature verification failed, message tampered with
}
6. IpfsAddress = requestIpfsInfo ($CID$)//Retrieve the identity information of the uploader based on the summary information
7. flag = VerifyUploaderInfo (address, ipfs_Address)//Verify the identity information to ensure the consistency of the uploader's identity
If flag == false{
    Return failed
}
9. tx_hash = contractInstance.request_upload ($N$, $Upk$, $Ts$, $CID$, $Sign$). transaction ({from: address)})//Call the request_upload method in the contract to upload the file hash to the blockchain
10. return Success//Complete the upload operation
```

(3) Design of metadata query contract algorithm

The user sends a query request $R$ to the blockchain, indexing it through the file $ID$, which is the unique feature value mapped to the file. After completing the query, the result is returned to the user. The query request message is defined as equation:

$$R = \{TYPE = \text{“}SHARE\text{”}, ID\} \tag{5}$$

where $SHARE$ indicates that the request is a production query request. To avoid performance degradation caused by increased query volume, this article designs a fast file retrieval framework: the supervising party node maintains a hash table $TABLE$ to achieve queries based on file $ID$s, which stores the mapping of file $ID$s to the blockchain location where the files are located. The $TABLE$ structure is shown in equation:

$$TABLE = \{KEY{:}ID,\ VLAUE{:}(TimeStamp, n)\} \tag{6}$$

where $ID$ refers to the project data number that will not be duplicated globally; $Timestamp$ is the file storage timestamp, and $n$ is the blockchain location where the query information is stored. Verify the return result of the contract by calling the size method to determine whether file data have been queried. If the query is successful, the first storage record is returned. If it fails the default string is returned. The metadata query contract algorithm can be found in Algorithm 3.

**Algorithm 3**: Metadata Query

```
Input: Query request $R$
Output: File Hash
1. if request. type == "SHARE" {//Determine the current request type, where "Share" indicates a data query request
    2. If hashTable [$ID$]! = nil {//Check if the request $ID$ exists in the hash table. If it does not exist, it indicates that the current request is illegal
        3. file_hashs = contractInstance. get_filehash (address). call()//Call the get_filehash method to obtain the file hash
        4. file_hash, flag = Size (file_hashs)//Verify the return results of the contract using the size method
        5. If flag! = nil{
            6. return nil//Query failed
        }
        7. return file_hash//Return the final hash
    }
}
8. Function Size (array) string{
    9. If len (array) == 0{
        Return nil. //The current array is empty
    }
    10. return array [len (array) $-$ 1]//Returns the latest value
}
```

(4) Design of IPFS file download contract algorithm

The user first sends the original file download request to the data uploader, and the data uploader sends the private key signature and access attribute key *Enck* (*K*) to the requester after verification. Then construct the download request information as shown in equation:

$$DQ = \{PK, CID, Sign(CID \| SK)\} \tag{7}$$

where *PK* is the uploader's public key, *CID* is the hash address that uniquely identifies the file returned by the IPFS file system, $Sign(CID\|SK)$ refers to the private key signature sent by the data uploader to the downloading user. After receiving the download request in the IPFS, the user's public key *PK* is first used to decrypt the signature in the message. If the verification is successful, it indicates that the downloading user is authorized by the file uploader, and then the encrypted file is obtained based on the *CID* query. The user first decrypts the key, and then decrypts the encrypted file in *D* using the key *K* to obtain the original file. The IPFS file download contract algorithm can be found in Algorithm 4.

**Algorithm 4**: Data Download

```
Input: File $CID$, uploader public key $Upk$, private key signature $Sign(CID\|Usk)$
Output: success or reject
1. requestBody = generateRequest ($CID$, $PK$, $Sign(CID\|SK)$)
2. flag = requestToOwner (requestBody)//The user initiates a download request, first verifying whether the user meets the download requirements
3. If flag == false{
    return reject//Reject the user's download request
}
4. ciphertext_file = GetFileByHash ($CID$)//Request encrypted file from ipfs through cid
5. sendKeyToRequester ($Enck(K)$)//Send the encrypted key to the requester
6. flag, _$K$ = decryptFile ($Enck(K)$, $PK$, $U'sk$)//Run the CP-ABE algorithm to decrypt the key
7. If flag == flame{
    8. raw_file = decryptFile ($K$, ciphertext_file)//Decrypt the file
    9. return success
}
10. return failed
```

### *3.4. Secure Sharing Process of Engineering Supervision Data Based on Blockchain*

This article constructs an engineering supervision data-sharing framework based on consortium chain through rapid retrieval of engineering supervision data, and designs corresponding smart contracts. The specific data flow of this framework model is as follows:

Step 1. After categorizing and organizing the files, the participating unit users encrypt the original files using the project participant's public key and store them in the IPFS cluster.

Step 2. After the upload is completed, IPFS returns the data access address to the client of the participating unit.

Step 3. The participating unit users package the attribute information such as private key signatures, file *ID*s, and file hash addresses of all parties related to the file, and construct the authentication information. Upload information to the blockchain through deployed smart contracts and stamp it with a timestamp.

Step 4. After completing the on-chain process, send back the on-chain file *ID* to the client of the participating unit.

Step 5. Ordinary users on the chain send query requests and perform query operations through smart contracts based on the on-chain file *ID*.

Step 6. After completing the query, return the operation status and query results.

Step 7. Project participants send file data retrieval requests to IPFS and send the hash address of the file to the IPFS cluster for querying.

Step 8. IPFS retrieves the encrypted file based on the index query and sends it to the client. The searcher decrypts the complete original file using the private key off chain.

Step 9. During the data circulation process of this framework, a hash table will be maintained for the on-chain node data to record the position of the engineering supervision data summary in the consortium chain, and fast retrieval will be achieved by locating the block where the data are located.

## 4. Performance Evaluation and Analysis of Proposed Framework

The proposed framework adopts a consortium blockchain with a limited number of nodes and is jointly maintained by multiple organizations or institutions. We will mainly analyze and evaluate the performance of the framework proposed in this paper from the aspects of security, encryption and decryption, cost, and framework comparison, in order to evaluate the security and reliability of engineering supervision data sharing.

### *4.1. Building the Blockchain Experimental Environment*

Before system development, it is necessary to deploy the development environment. In order to meet the requirements of the supervision data-sharing function, a total of four nodes are deployed to represent various departments in the supervision data-sharing process, to complete operations such as data verification query, consensus, and on-chain certificate storage. The specific information of the nodes is shown in Table 9.

**Table 9.** Node information.

| Node | Node Type | Node Function | Node Organization |
|---|---|---|---|
| Peer0 | Consensus node | Endorsement, Verification, Submission | Supervision company |
| Peer1 | Consensus node | Endorsement, Verification, Submission | Supervision company |
| Peer2 | Consensus node | Endorsement, Verification, Submission | Government management |
| Peer3 | Consensus node | Endorsement, Verification, Submission | Construction unit |

This system is developed based on the B/S architecture using Ubuntu 18.04 as the underlying operating system. The front-end uses Vue as the UI framework, and the back-end is implemented based on Java language. The development environment of the system is shown in Table 10.

**Table 10.** NSystem development environment.

| Type | Content |
|---|---|
| Overall environment | Virtual machine VMware 14 Pro |
| Operating system | Ubuntu 18.04 |
| Lower-level consortium blockchain | FISCO BCOS V2.8.0 |
| Programming language | Solidity, Java, Go |
| Database | Mysql 5.7.31 IPFS 0.9.0 |
| Developing software | LiteIDE x37.3 |

The development is carried out in the virtual machine VMware 14 Pro under Windows 10. Firstly, install the Ubuntu 18.04 system on the virtual machine and download the corresponding script. By using the build_chain blockchain script, build a 4-node FISCO BCOS blockchain, input instructions to start all nodes, and check the node process and log output status. After confirming the completion of node startup, enter the command to obtain the console file, and copy the console configuration file and configuration console certificate to complete the Java SDK-based console setup. After the console is successfully launched, deploy the IPFS network. The specific process is as follows:

Step 1. Install go-ipfs under all nodes in the IPFS network and initialize the IPFS network.
Step 2. Input instructions to create a key file and send the file to other IPFS nodes, which will copy the file to the control folder under the node.
Step 3. Enter the command to enable the node, which can connect and access each other.
Step 4. The smart contract development is completed through the console based on the supervision data-sharing process, and the smart contract is deployed to the corresponding directory.

### *4.2. Analysis of Security Assessment*

The security assessment of the proposed framework is analyzed from the aspects of data immutability, data integrity, data privacy, and protocol attacks.

#### 4.2.1. Data Immutability

When a user queries data from the IPFS file system through *CID*, the visited node will combine the distributed hash table and routing mechanism based on the anti-tampering mechanism stored in the IPFS, as shown in Figure 2 to obtain the NodeID of the node where the searched file is located. Then, the routing mechanism will route to the node, query the file, and return it to the client. If a user modifies the original file data, the IPFS generates a hash address that uniquely corresponds to the file. Changing the original file will result in a change in its corresponding hash address. Due to the characteristics of the hash algorithm, the hash address uploaded to the blockchain cannot be tampered with, and forged data will not pass verification. Comparing and testing ensures the immutability of supervision data.

#### 4.2.2. Data Integrity

By adopting a data storage framework that combines on-chain metadata storage and the off-chain IPFS storage of original files, the system can operate normally even if one or several nodes fail, avoiding data loss caused by single-node downtime and protecting data integrity. In the application scenario of supervision data sharing, data are uploaded by the supervision party and exchanged with all participating parties in the construction. The entire process is executed by smart contracts, protecting the security of data during transmission.

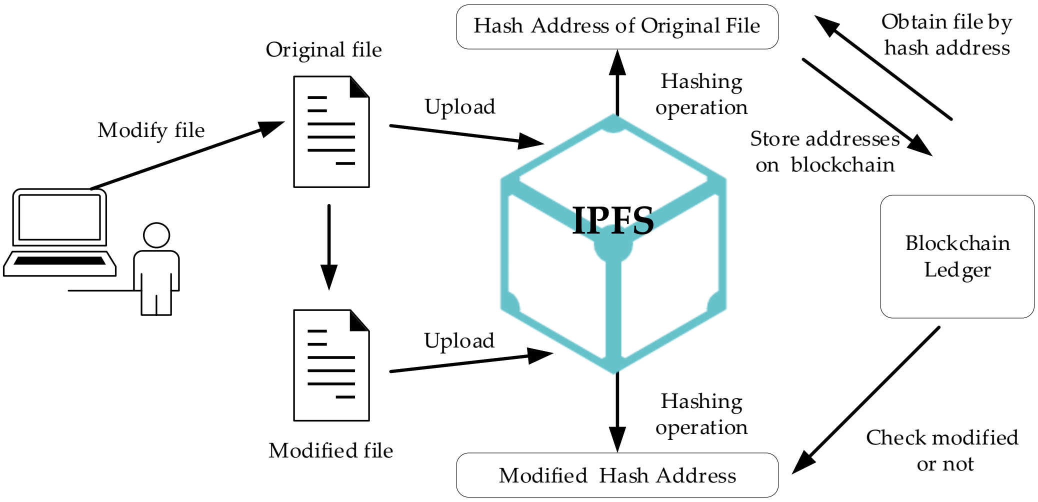

**Figure 2.** Anti-tampering mechanism diagram based on IPFS storage.

### 4.2.3. Data Privacy

The proposed framework involves data uploaders encrypting and storing supervision files in the IPFS file system, uploading the file addresses returned by the IPFS file system to the blockchain system, and encrypting the keys using the CP-ABE framework. On-chain users are only visible to file addresses. The file uploader restricts the download permission of the data by specifying the policy Pa. Only data users who meet the policy *Pa* attribute can decrypt the key, thereby decrypting the supervision file and completing data privacy protection.

### 4.2.4. Protocol Attacks

This framework can effectively resist identity disguise attacks and replay attacks. The requester sends a file download request to the supervisor data uploader, and after passing identity verification, the uploader authorizes and obtains the ciphertext. Only users who meet the policy *Pa* attribute can decrypt the ciphertext. During this process, third-party attackers are unable to obtain plaintext supervision data, and protocol attacks are divided into the following two situations.

(1) The attacker queries the file address *CID* on the blockchain and sends a download request to the IPFS file system. Due to the data uploader not authorizing the user, the attacker was unable to authenticate and the IPFS node refused the data download request.

(2) Attackers perform replay attacks by intercepting user-sent request messages. Attackers can disguise themselves as data requesters and send download requests to data uploaders to obtain their signature information. Through identity authentication of IPFS nodes, combined with the file address *CID* on the chain, attackers can obtain data ciphertext. During this process, attackers can obtain encrypted files through replay attacks, but the key to the ciphertext is only owned by users who meet policy attributes, so the attacker cannot decrypt and obtain the original file.

## *4.3. Performance Evaluation Analysis on Encryption and Decryption*

The encryption and decryption performance of the proposed framework is analyzed in the experimental environment, as shown in Table 10. In terms of performance indicator selection, the CP-ABE algorithm in the proposed framework is simulated and analyzed based on the evaluation method in reference [25]. Combined with the application background of engineering supervision data sharing, three attributes are set for the attribute private key in the CP-ABE algorithm, including user registration time, position name, and department name. The simulation results are shown in Figure 3. The encryption and decryption time increases linearly with the increase in data to be encrypted and decrypted, but all are completed within 0.1 s. The supervision data access control mechanism based on the CP-ABE algorithm implemented on the consortium blockchain network has good feasibility.

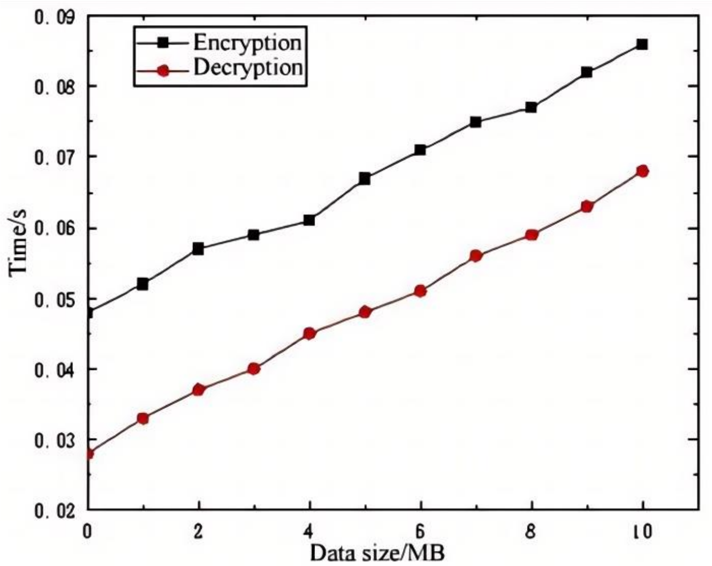

**Figure 3.** Analysis of encryption and decryption performance.

### *4.4. Cost Evaluation*

Through specific deployment and invocation of data upload, metadata upload, metadata query, and IPFS file download of the framework proposed in this article, the Gas cost and time cost are evaluated and tested. The Gas cost refers to the efficiency cost, which can be used as a count to evaluate the cost of blockchain systems. Its normal range is several thousand Gas to over 20,000 Gas.

Due to the fact that blockchain consumes a certain amount of Gas for each operation, testing items such as supervision data upload, metadata upload and query, and IPFS file download are set based on the consumed Gas. The test results are shown in Table 11, indicating that the cost of Gas in this blockchain framework is relatively low and within the normal consumption range. In addition, the time cost of smart contracts is tested on a locally built blockchain platform, and the average time cost of 100 calls to smart contracts is tested. The results show that the smart contract call time cost of the framework proposed in this paper is relatively small, not exceeding 5 s.

**Table 11.** Gas Cost estimation of blockchain.

| Test Items | Specific Settings Section | Cost/Gas |
|---|---|---|
| Supervision data | Supervision data upload | 25,236 |
| Metadata | Metadata upload | 20,640 |
| | Metadata query | 16,906 |
| IPFS file | IPFS file download | 24,677 |

### *4.5. Comparative Evaluation and Analysis of Frameworks*

The supervision data-sharing framework based on consortium blockchain proposed in this article is compared with existing engineering data-sharing frameworks using a comparative analysis method from four aspects including blockchain type, user-level access control granularity, data privacy, and computing power requirements. Among them, user-level access control granularity refers to whether the framework supports user-level access control functions; data privacy refers to whether the framework protects the privacy of user information on the chain.

The analysis results are shown in Table 12; when using the consortium blockchain as the underlying framework of the data-sharing application, the system runtime is determined by the consensus algorithm. The PBFT consensus algorithm is used in the proposed framework, which has better performance and lower computational power requirements compared to the POW consensus algorithm used in reference [37]. Compared with reference [18], this proposed framework combines CP-ABE encryption with smart contracts to achieve privacy protection for on-chain data and user-level access control functions. Therefore, this proposed framework is more suitable for engineering data security sharing applications based on consortium blockchains compared to other frameworks.

**Table 12.** Comparison between the proposed framework and existing frameworks.

| Frame | Blockchain Type | User Level Access Control Granularity | Data Privacy | Computing Power Requirement |
|---|---|---|---|---|
| Reference [37] | Consortium blockchain | No | No | High |
| Reference [18] | Consortium blockchain | No | No | Low |
| Proposed framework | Consortium blockchain | Yes | Yes | Low |

## 5. Discussion

Engineering construction supervision has the characteristics of multiple participants, a large amount of supervision data, and high difficulty in information control. If blockchain technology is directly used to build a supervision data-sharing platform, issues such as blockchain data storage pressure, data-sharing privacy, consensus algorithm scalability, and performance will become the main technical bottlenecks [38]. This article proposes a blockchain-based framework for the secure sharing of engineering supervision data, by combining it with the IPFS to achieve the large-scale storage of engineering supervision data. A data fast retrieval framework is designed based on the storage characteristics of engineering supervision data. And CP-ABE is combined with smart contract technology to achieve access control in fine-grained privacy protection scenarios, and then smart contracts required for engineering supervision data-sharing applications are designed. Finally, the security, encryption and decryption, and cost performance of the framework are analyzed and evaluated. The experimental results show that the proposed framework in this paper solves the trust problem between various parties in engineering supervision data sharing, effectively meets the security requirements of engineering supervision data sharing, ensures the traceability of the engineering supervision data-sharing process, and has good usability and reusability.

The engineering supervision data security sharing system is mainly applied in the consortium chain network environment, including the IPFS network for storing metadata off chain and the consortium chain network for storing summary information on chain. By sharing engineering supervision data, each block node of the supervision party supervises each other and jointly maintains the security and consistency of the on-chain engineering supervision data, achieving efficient and standardized management of engineering supervision data and secure sharing applications. Therefore, by combining the application scenario of blockchain engineering supervision data sharing, this article proposes a secure sharing application architecture for engineering supervision data based on a consortium chain, as shown in Figure 4. The architecture mainly includes five layers: application layer, contract layer, interface layer, consortium chain service layer, and data storage layer. Relevant application research and development can be carried out based on this architecture.

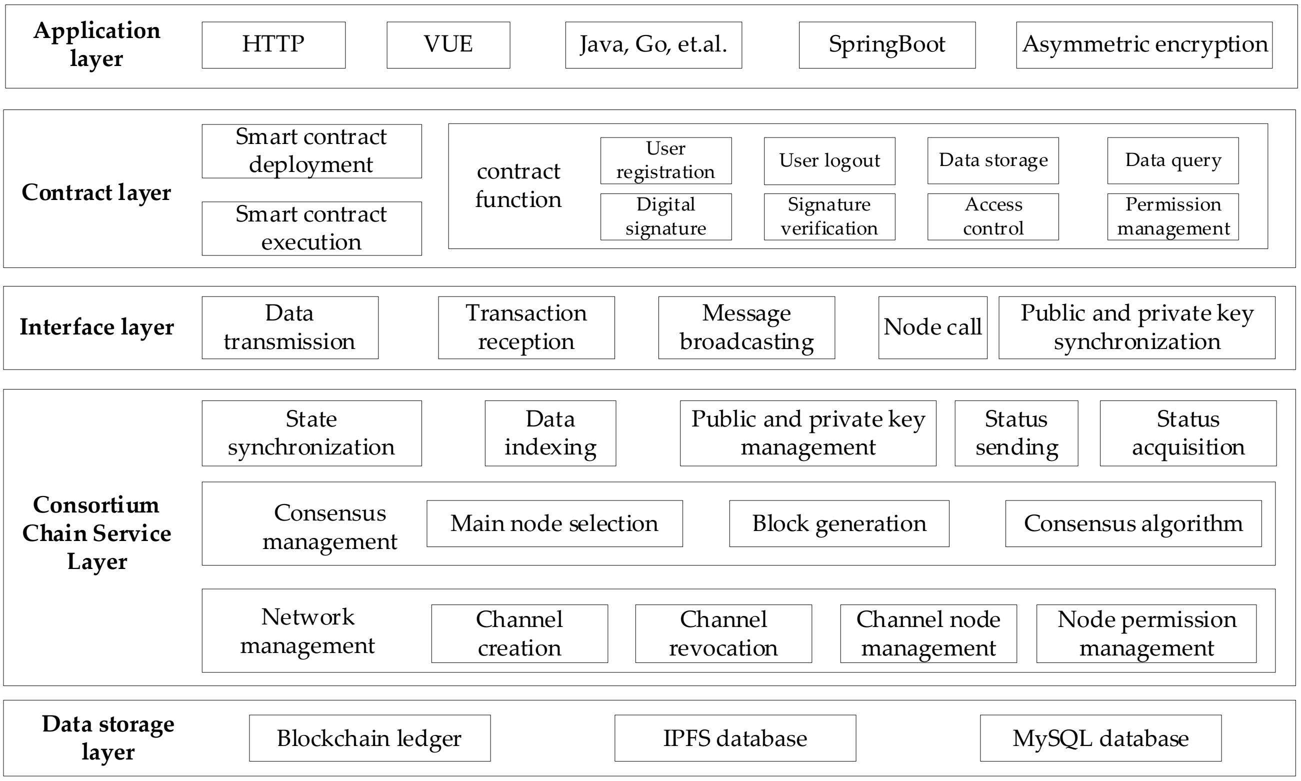

**Figure 4.** Overall architecture of blockchain-based supervision data-sharing system.

(1) Application layer. This layer refers to the application implementation of establishing a data-sharing system based on blockchain, which implements transaction functions in the form of client operations. Depending on the permission settings, users can complete operations such as uploading, querying, and downloading supervision data through the client.

(2) Contract layer. This layer is the technical layer that enables data sharing and transactions on the blockchain network, including data uploading, querying, and user access control, while being able to manage the deployment and execution of smart contracts. This contract combines the sharing requirements of supervision data, implements access control for the sharing process, and completes the spontaneous operation management of supervision data sharing.

(3) Interface layer. The function of this layer is to connect the blockchain network and clients. It provides a way for clients to interact with the blockchain system, enabling transactions to be transmitted and received through the client, and accessing the functions and data of the blockchain.

(4) Consortium Chain Service Layer. This layer is based on the consortium chain and mainly implements management of ledger status, consensus management, and blockchain network management. The framework of this article is based on a decentralized consortium chain implementation, where participating parties can interact with supervision data without relying on third-party trusted institutions, effectively improving data-sharing efficiency.

(5) Data storage layer. This layer is mainly used to store data information generated during system operation, including three parts: blockchain ledger, IPFS distributed file system, and MySQL relational database. The data layer in the framework of this article mainly stores user information of various participating parties, as well as various file data required in the supervision business process.

Therefore, this article proposes a secure sharing framework for engineering supervision data based on a consortium chain, which solves the storage and transmission problems of large-scale data in engineering supervision under the premise of decentralization of blockchain, and addresses the security risks [39] in engineering supervision data sharing. The proposed framework realizes the secure sharing application and trusted confidentiality governance of engineering supervision data, effectively meeting the needs of complex engineering supervision data security sharing applications and trusted management. However, there are many factors involved in the calculation of the trust degree and credit rating of blockchain nodes, which makes the calculation process slightly cumbersome. In the subsequent work, it is necessary to further streamline the node calculation method and improve the stability and robustness of the operation of the engineering supervision data security sharing application system. In addition, in order to apply this framework to more practical scenarios in the future, it is necessary to further analyze the actual business scenarios of data security sharing and trustworthy governance, and then optimize and integrate the system with existing businesses, reflecting the practical significance and value of blockchain technology in actual industry applications.

## 6. Conclusions

In response to the security and trust issues in the application process of engineering supervision data sharing, this paper proposes a blockchain-based framework for the secure sharing of engineering supervision data. By combining with the IPFS, the proposed framework achieves secure storage and fast retrieval of large-scale supervision data. Then, the CP-ABE technology is combined with a data storage framework to achieve access control in fine-grained privacy protection scenarios. Subsequently, a smart contract is designed for the shared operation of engineering supervision data, which utilizes the mandatory execution and anti-tampering characteristics of smart contracts to ensure the automatic and trustworthy execution of access control processes for various blockchain nodes. Finally, a security and performance cost evaluation analysis is conducted on the framework proposed in this article, and a secure sharing application architecture based on engineering supervision data is proposed. The experimental results show that the framework proposed in this paper can effectively achieve secure data sharing and meet practical application needs. It solves the storage problem of large-scale data while ensuring decentralization, making it suitable for more scenarios of engineering supervision data-sharing applications. In the subsequent work, digital signature technology will be combined to further ensure data privacy and security, and more detailed access control mechanisms will be developed to ensure data security during the data flow process.

**Author Contributions:** Conceptualization, J.Y.; methodology, J.Y. and Z.H.; software, J.Y., X.L. and Z.H.; validation, X.L. and Z.H.; formal analysis, J.D. and Y.W.; data curation, X.L.; writing—original draft preparation, J.Y.; writing—review and editing, J.Y., X.L., J.D. and Y.W.; visualization, J.Y. and X.L.; supervision, J.D. and Y.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded in part by the National Natural Science Foundation of China, grant numbers 62367005 and 62067006; in part by the Natural Science Foundation of Gansu Province, grant numbers 23JRRA845; in part by the Youth Science and Technology Talent Innovation Project of Lanzhou, grant numbers 2023-QN-117; and in part by the Joint Innovation Fund Project of Lanzhou Jiaotong University and Corresponding Supporting University, grant numbers LH2024012.

**Data Availability Statement:** The data are contained within this article.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Saxena, S.; Bhushan, B.; Ahad, M.A. Blockchain based solutions to secure IoT: Background, integration trends and a way forward. *J. Netw. Comput. Appl.* **2021**, *18*, 103050. [CrossRef]
2. Song, R.; Xiao, B.; Song, Y.; Guo, S.; Yang, Y. A Survey of Blockchain-Based Schemes for Data Sharing and Exchange. *IEEE Trans. Big Data* **2023**, *9*, 1477–1495. [CrossRef]

3. Bhushan, B.; Khamparia, A.; Sagayam, K.M.; Sharma, S.K.; Ahad, M.A.; Debnath, N.C. Blockchain for smart cities: A review of architectures, integration trends and future research directions. *Sustain. Cities Soc.* **2020**, *61*, 102360. [CrossRef]
4. Perera, S.; Nanayakkara, S.; Rodrigo, M.N.N.; Senaratne, S.; Weinand, R. Blockchain technology: Is it hype or real in the construction industry? *J. Ind. Inf. Integr.* **2020**, *17*, 100125. [CrossRef]
5. Fugkeaw, S.; Wirz, L.; Hak, L. Secure and Lightweight Blockchain-Enabled Access Control for Fog-Assisted IoT Cloud Based Electronic Medical Records Sharing. *IEEE Access* **2023**, *11*, 62998–63012. [CrossRef]
6. Rahman, M.S.; Al Omar, A.; Bhuiyan, M.Z.A.; Basu, A.; Kiyomoto, S.; Wang, G. Accountable Cross-Border Data Sharing Using Blockchain Under Relaxed Trust Assumption. *IEEE Trans. Eng. Manag.* **2020**, *67*, 1476–1486. [CrossRef]
7. Qin, X.; Huang, Y.; Yang, Z.; Li, X. A Blockchain-based access control scheme with multiple attribute authorities for secure cloud data sharing. *J. Syst. Archit.* **2021**, *112*, 101854. [CrossRef]
8. Li, W.; Feng, C.; Zhang, L.; Xu, H.; Cao, B.; Imran, M.A. A Scalable Multi-Layer PBFT Consensus for Blockchain. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 1146–1160. [CrossRef]
9. Xu, J.; Hua, C.; Zhang, Y. A Blockchain-Based Framework for Supervision of Livelihood Issues: Proof of Concept With Optimized Consensus. *IEEE Access* **2023**, *11*, 73414–73434. [CrossRef]
10. Kumar, P.; Gupta, G.P.; Tripathi, R. TP2SF: A Trustworthy Privacy-Preserving Secured Framework for sustainable smart cities by leveraging blockchain and machine learning. *J. Syst. Archit.* **2021**, *115*, 101954. [CrossRef]
11. Makhdoom, I.; Zhou, I.; Abolhasan, M.; Lipman, J.; Ni, W. PrivySharing: A blockchain-based framework for privacy-preserving and secure data sharing in smart cities. *Comput. Secur.* **2020**, *88*, 101653. [CrossRef]
12. Quan, G.; Yao, Z.; Chen, L.; Fang, Y.; Zhu, W.; Si, X.; Li, M. A trusted medical data sharing framework for edge computing leveraging blockchain and outsourced computation. *Heliyon* **2023**, *9*, e22542. [CrossRef] [PubMed]
13. R., R.K.; Kallapu, B.; Dodmane, R.; S., K.R.N.; Thota, S.; Sahu, A.K. Enhancing Cloud Communication Security: A Blockchain-Powered Framework with Attribute-Aware Encryption. *Electronics* **2023**, *12*, 18. [CrossRef]
14. Wei, X.; Yan, Y.; Guo, S.; Qiu, X.; Qi, F. Secure Data Sharing: Blockchain-Enabled Data Access Control Framework for IoT. *IEEE Internet Things J.* **2022**, *9*, 8143–8153. [CrossRef]
15. Banik, M.; Kumar, S. Blockchain-based public key encryption with keyword search for medical data sharing in cloud environment. *J. Inf. Secur. Appl.* **2023**, *78*, 103626. [CrossRef]
16. Eltayieb, N.; Elhabob, R.; Hassan, A.; Li, F. A blockchain-based attribute-based signcryption scheme to secure data sharing in the cloud. *J. Syst. Archit.* **2020**, *102*, 101653. [CrossRef]
17. Guo, H.; Liang, H.; Zhao, M.; Xiao, Y.; Wu, T.; Xue, J.; Zhu, L. Privacy-Preserving Fine-Grained Redaction with Policy Fuzzy Matching in Blockchain-Based Mobile Crowdsensing. *Electronics* **2023**, *12*, 16. [CrossRef]
18. Jia, L.; Chen, X.; Liu, L.; Wang, X.; Xiao, K.; Xu, G. Blockchain data secure sharing protocol based on threshold Paillier algorithm. *High-Confid. Comput.* **2023**, *3*, 4. [CrossRef]
19. Li, X.; Zhao, H.; Deng, W. BFOD: Blockchain-Based Privacy Protection and Security Sharing Scheme of Flight Operation Data. *IEEE Internet Things J.* **2024**, *11*, 3392–3401. [CrossRef]
20. López-Sorribes, S.; Rius-Torrentó, J.; Solsona-Tehàs, F. BRAIN: Blockchain-Based Record and Interoperability Network. *Electronics* **2023**, *12*, 22. [CrossRef]
21. Ma, X.; Wang, C.; Chen, X. Trusted data sharing with flexible access control based on blockchain. *Comput. Stand. Interfaces* **2021**, *78*, 103543. [CrossRef]
22. Singh, C.E.J.; Sunitha, C.A. Chaotic and Paillier secure image data sharing based on blockchain and cloud security. *Expert Syst. Appl.* **2022**, *198*, 116874. [CrossRef]
23. Wan, C.; Wang, Y.; Xu, J.; Wu, J.; Zhang, T.; Wang, Y. Research on Privacy Protection in Federated Learning Combining Distillation Defense and Blockchain. *Electronics* **2024**, *13*, 4. [CrossRef]
24. Wang, L.; Guan, C. Improving Security in the Internet of Vehicles: A Blockchain-Based Data Sharing Scheme. *Electronics* **2024**, *13*, 714. [CrossRef]
25. Wang, T.; Wu, Q.; Chen, J.; Chen, F.; Xie, D.; Shen, H. Health data security sharing method based on hybrid blockchain. *Future Gener. Comput. Syst.* **2024**, *153*, 251–261. [CrossRef]
26. Agyekum, K.O.-B.O.; Xia, Q.; Sifah, E.B.; Cobblah, C.N.A.; Xia, H.; Gao, J. A Proxy Re-Encryption Approach to Secure Data Sharing in the Internet of Things Based on Blockchain. *IEEE Syst. J.* **2022**, *16*, 1685–1696. [CrossRef]
27. Cao, B.; Wang, X.; Zhang, W.; Song, H.; Lv, Z. A Many-Objective Optimization Model of Industrial Internet of Things Based on Private Blockchain. *IEEE Netw.* **2020**, *34*, 78–83. [CrossRef]
28. Deng, S.; Hu, Q.; Wu, D.; He, Y. BCTC-KSM: A blockchain-assisted threshold cryptography for key security management in power IoT data sharing. *Comput. Electr. Eng.* **2023**, *108*, 108666. [CrossRef]
29. Li, T.; Wang, H.; He, D.; Yu, J. Blockchain-Based Privacy-Preserving and Rewarding Private Data Sharing for IoT. *IEEE Internet Things J.* **2022**, *9*, 15138–15149. [CrossRef]
30. Lu, Y.; Huang, X.; Dai, Y.; Maharjan, S.; Zhang, Y. Blockchain and Federated Learning for Privacy-Preserved Data Sharing in Industrial IoT. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4177–4186. [CrossRef]
31. Zhou, Z.; Tian, Y.; Xiong, J.; Ma, J.; Peng, C. Blockchain-Enabled Secure and Trusted Federated Data Sharing in IIoT. *IEEE Trans. Ind. Inform.* **2023**, *19*, 6669–6681. [CrossRef]

32. Hassan, M.M.; Lin, K.; Yue, X.; Wan, J. A multimedia healthcare data sharing approach through cloud-based body area network. *Future Gener. Comput. Syst.* **2017**, *66*, 48–58. [CrossRef]
33. Wang, Y.; Cai, S.; Lin, C.; Chen, Z.; Wang, T.; Gao, Z.; Zhou, C. Study of Blockchains's Consensus Mechanism Based on Credit. *IEEE Access* **2019**, *7*, 10224–10231. [CrossRef]
34. Sheng, D.; Ding, L.; Zhong, B.; Love, P.E.D.; Luo, H.; Chen, J. Construction quality information management with blockchains. *Autom. Constr.* **2020**, *120*, 103373. [CrossRef]
35. Liu, Z.; Wang, S.; Liu, Y. Blockchain-based integrity auditing for shared data in cloud storage with file prediction. *Comput. Netw.* **2023**, *236*, 110040. [CrossRef]
36. Liu, Y.; Xu, G. Fixed degree of decentralization DPoS consensus mechanism in blockchain based on adjacency vote and the average fuzziness of vague value. *Comput. Netw.* **2021**, *199*, 108432. [CrossRef]
37. Lee, D.; Lee, S.H.; Masoud, N.; Krishnan, M.S.; Li, V.C. Integrated digital twin and blockchain framework to support accountable information sharing in construction projects. *Autom. Constr.* **2021**, *127*, 103688. [CrossRef]
38. Fugkeaw, S.; Hak, L.; Secure, T.A. Verifiable, and Efficient Boolean Keyword Searchable Encryption for Cloud Data Warehouse. *IEEE Access* **2024**, *12*, 78743–78758. [CrossRef]
39. Popoola, O.; Rodrigues, M.; Marchang, J.; Shenfield, A.; Ikpehia, A.; Popoola, J. A critical literature review of security and privacy in smart home healthcare schemes adopting IoT & blockchain: Problems, Challenges and Solutions. *Blockchain Res. Appl.* **2023**, *5*, 100178.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.