*electronics*

MDPI

*Article*

# A Blockchain and Zero Knowledge Proof Based Data Security Transaction Method in Distributed Computing

**Bowei Zhang [1,*], Heng Pan [1,*], Kunyang Li [1], Ying Xing [1], Jiaxiang Wang [2], Dongdong Fan [1] and Wenjie Zhang [1]**

[1] Cyber Security Academy, Zhongyuan University of Technology, Zhengzhou 450007, China; likunyanggg@163.com (K.L.); xingying_vip@163.com (Y.X.); fanddong00@zut.edu.cn (D.F.); z972843100@gmail.com (W.Z.)

[2] School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China; jiaxiang.wang21@student.xjtlu.edu.cn

* Correspondence: bll05444@163.com (B.Z.); panheng@zut.edu.cn (H.P.)

**Abstract:** In distributed computing, data trading mechanisms are essential for ensuring the sharing of data across multiple computing nodes. Nevertheless, they currently encounter considerable obstacles, including low accuracy in matching trading parties, ensuring fairness in transactions, and safeguarding data privacy throughout the trading process. In order to address these issues, we put forward a data trading security scheme based on zero-knowledge proofs and smart contracts. In the phase of preparing the security parameters, the objective is to reduce the complexity of generating non-interactive zero-knowledge proofs and to enhance the efficiency of data trading. In the pre-trading phase, we devise attribute atomic matching smart contracts based on precise data property alignment, with the objective of achieving fine-grained matching of data attributes between trading parties. In the trading execution phase, lightweight cryptographic algorithms based on elliptic curve cryptography (ECC) and non-interactive zero-knowledge proofs are employed for the dual encryption of trading data and the generation of attribute proof contracts, thus ensuring the security and privacy of the data. The results of experiments conducted on the Ethereum platform in an industrial IoT scenario demonstrate that our scheme maintains stable and low-cost consumption while ensuring accuracy in matching and privacy protection.

**Keywords:** blockchain; data trading; zero knowledge proof; privacy-preserving; smart contract

check for updates

**Citation:** Zhang, B.; Pan, H.; Li, K.; Xing, Y.; Wang, J.; Fan, D.; Zhang, W. A Blockchain and Zero Knowledge Proof Based Data Security Transaction Method in Distributed Computing. *Electronics* **2024**, *13*, 4260. https://doi.org/10.3390/electronics13214260

Academic Editor: Mehdi Sookhak

Received: 9 September 2024
Revised: 15 October 2024
Accepted: 29 October 2024
Published: 30 October 2024

**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

With the rapid evolution of information technology, various connected devices are constantly generating massive amounts of data. For example, various distributed devices, social networks and other technologies have contributed enormously to increased data generation. According to media predictions, the number of IoT-related devices in everyday life is expected to increase to 41.6 billion within a year. This exponential growth makes it extremely challenging to perform complex retrieval and analysis from such vast and diverse data sources. However, with the rapid advancement of AI in recent years and the corresponding progress in related technologies, the potential of distributed computing has been unleashed. Distributed computing provides efficient computing power and resource management for handling large-scale data. Consequently, leveraging these cutting-edge technologies for model training on the vast amounts of data generated daily has become an inevitable trend [1–3].

Consequently, as data-driven approaches have become central to many distributed computing systems, data trading has emerged as a model for the circulation of data. By monetizing data resources, data trading facilitates the connection between data owners and data consumers, thereby enhancing the utilization of multi-source data and addressing the current issue of data silos. By leveraging large-scale AI models, the data demand side can analyze data acquired from various distributed scenarios through data transactions,

thereby deriving greater benefits, enhancing efficiency, and reducing production costs. In this context, a typical distributed computing application, federated learning (FL) in edge computing, becomes a key technology for processing and exploiting massive data [4,5]. For instance, industrial sectors can subscribe to diverse types of industrial data (such as temperature and humidity data from assorted sensors) through data exchange platforms to train their models, resulting in enhanced efficiency and superior outcomes.

In traditional data trading systems, data owners and consumers are required to rely on trading service providers for the management of access control, authorization, and data retrieval. This centralized approach relies on third-party entities, lacks transparency, and increases the impact of issues arising from a single centralized node. This is inconsistent with the core concept of distributed systems. Blockchain is an emerging distributed technology that has been widely applied in data trading systems due to its decentralized, publicly verifiable, and tamper-resistant ledger characteristics [6]. Nevertheless, given the highly sensitive nature of the data being traded, relying on a single distributed technology alone does not fully address all issues. As an illustrative example, the low accuracy of trading group matching results leads to inefficient matching between data owners and consumers. Insufficient privacy protection for sellers may result in data breaches and misuse. A lack of security during data transmission renders the data vulnerable to tampering or theft. Furthermore, unfair data value exchange can harm the interests of both parties involved in the transaction.

In order to protect against potential privacy breaches that may occur during the course of a transaction, the prevailing approach to data protection in the context of data trading is to rely on cryptographic security mechanisms. A number of researchers have put forward a variety of cryptographic solutions with the aim of ensuring the security of data trading. For example, there are schemes based on digital signatures [7,8] and those based on zero-knowledge proofs [9,10]. Schemes based on digital signatures permit data providers and consumers to publish and purchase data without disclosing their true identities, and to authenticate data sharing under conditions of anonymity. Nevertheless, the act of signing a data entity primarily serves to confirm the ownership of the data, rather than to verify the accuracy of the data entity itself. Conversely, zero-knowledge proof technology can validate the integrity and legality of the transaction data while maintaining the confidentiality of the transactional details. The development of zero-knowledge proofs has seen them applied extensively in data trading scenarios, whether in the form of basic interactive zero-knowledge proofs or noninteractive zero-knowledge proofs [11]. This is due to the unique characteristics of the technology, which have yielded positive results.

Nevertheless, in the context of data trading, the application of zero-knowledge proofs still presents certain challenges, such as the construction of proofs for confidential data. Furthermore, two significant challenges must be addressed and resolved in the context of data trading. Firstly, there is a risk that the data seller may send the data without receiving payment, thereby enabling the data buyer to obtain the data without paying. Secondly, there is a possibility that the data buyer may be unable to access the data before payment, while the data seller, after receiving payment, may fail to deliver the data as per the contract.

To address the limitations of existing data trading systems in terms of privacy protection, transaction fairness, and storage scalability, we propose a blockchain-based data trading solution. This solution utilizes a systematic technology selection process to choose appropriate technologies for different requirements. In terms of privacy protection, we employ non-interactive zero-knowledge proofs (zk-SNARKs) to ensure that both parties can complete the verification process without exposing private data. For transaction fairness, we adopt elliptic curve cryptography (ECC) to provide higher computational efficiency and lower storage costs. To achieve precise matching of data attributes, we design an atomic matching-based smart contract mechanism to enhance transaction accuracy. Lastly, we utilize off-chain storage technologies (such as IPFS) to alleviate on-chain storage pressure and address user scalability issues. By storing encrypted data off-chain and using

the blockchain for indexing, we enhance the system's scalability, making it suitable for large-scale and complex data trading scenarios.

The principal contributions of this paper are outlined as follows:

- This paper presents the design of a blockchain-based smart contract based on the concept of atomic matching, which enables the fine-grained matching of data sellers and buyers according to detailed data attributes. Furthermore, the model incorporates off-chain storage and the separation of parameter preparation phases, thereby enhancing the system's scalability in handling large volumes of data and increasing numbers of users.
- A fair end-to-end protection scheme is proposed, which employs lightweight cryptographic algorithms for dual encryption of transaction data prior to the completion of the transaction. Furthermore, zero-knowledge proofs are utilised to verify the nature of the transaction data, thereby safeguarding the privacy of sensitive information for transaction users.

Regarding the organization of this paper, Section 2 introduces the related work. Section 3 presents the overall model of the proposed system and the threat analysis. The Section 4 describes the system architecture. Section 5 conducts simulation experiments and experimental analysis of the system. Section 6 provides a summary and analysis of this work.

## 2. Related Work

The unique distributed nature of blockchain technology gives it great potential and application value in data transactions. At the present time, the applications of blockchain technology in conjunction with other scenarios can be broadly categorized into the following types.

The advent of blockchain technology has rendered the concept of peer-to-peer (P2P) energy trading a tangible reality. This has paved the way for consumers and small-scale energy producers, such as those utilizing home solar systems, to engage in direct trading, thereby reducing reliance on traditional intermediaries. This model serves to enhance the efficiency of energy trading and to reduce costs. In their paper, Shunrong Jiang et al. [12] put forth a proposal for a blockchain-based decentralized energy trading system. The system employs smart contracts and privacy-preserving technologies, including zero-knowledge proofs, to facilitate direct, secure, and private energy transactions between users and suppliers. Furthermore, it incorporates demand response capabilities, thereby enhancing the efficiency of energy management and system reliability. Aitzhan et al. [13] concentrated on the protection of decentralized energy trading in smart grid environments, seeking to eliminate the necessity for centralized third-party institutions. They employed distributed ledger technology and proof of concept, combining multi-signature and encrypted anonymous messaging methods to facilitate the anonymous negotiation of energy prices and the secure execution of transactions. In a further development of the concept, Ye-Byoul Son et al. [14] proposed a blockchain-based P2P energy trading system in which all bids are encrypted. The implementation of encrypted bidding and smart contracts based on functional encryption (FE) was also explored. In addition to cryptographic solutions, some energy trading scenarios employ non-cryptographic methods. To illustrate, Daghmehchi et al. [15] put forth a proposal for an IoT platform wherein the platform provider does not act as an intermediary. The platform allows users to securely and efficiently share credits with other users in need, thereby ensuring privacy protection through k-anonymity on network bridges.

Furthermore, the quantity of data in medical contexts is also expanding at a considerable rate, and the issues pertaining to the privacy and security of this data are becoming increasingly prominent. Blockchain technology provides a secure data-sharing platform for patients, hospitals, and research institutions, thereby ensuring data confidentiality and immutability while allowing authorized parties to access and utilize the data. Liang Xue et al. [10] put forth a solution tailored to the data generated in medical scenarios,

which employs blockchain technology to facilitate fair and granular trading. This solution guarantees the secure transaction and protection of medical data through the use of verifiable attribute credentials and cryptographic techniques. Moreover, the utilization of Merkle hash trees and cryptographic knowledge safeguards the anonymity of data sellers, guaranteeing that the transaction process remains undetectable and providing a secure and efficient solution for medical data trading. Kumar et al. [16] devised a novel and secure data-sharing framework for industrial medical systems. This framework implements registration, verification (using zero-knowledge proofs), and consensus mechanisms based on smart contracts through a blockchain solution.

In the context of industrial Internet of Things (IIoT) data trading, Zhang [17] put forth a controlled IIoT data trading solution that employs automated smart contracts and hardware trusted execution environments. This approach enables data producers, or data owners, to retain complete control over their sovereignty throughout the trading process, while simultaneously reducing their own costs. In a further development of this concept, Akanksha Dixit et al. [18] introduced an innovative platform named FASTDATA, which utilizes blockchain technology to create a decentralized IIoT data trading marketplace. VerneMQ is employed as the data network layer to guarantee dependable hosting and fault-tolerant transmission of data flows. The integration of fully symmetric encryption schemes and digital signatures ensures the end-to-end encryption of data during transmission and storage, thereby safeguarding data privacy. Furthermore, a TrustScore mechanism was introduced to assess and enhance the credibility of network participants. Zhou et al. [19] devised the Federated Extreme Gradient Boosting (FedXGBoost) algorithm and a secure sharing mechanism for the aforementioned trading process. The STFS framework, which employs a range of technologies, including homomorphic encryption, secret sharing (SS), proxy re-encryption, and threshold aggregation signatures, effectively addresses issues of data privacy protection, model credibility and secure and controllable data sharing in the context of the Industrial Internet of Things (IIoT). Huang Cheng et al. [20] additionally proposed a traceable and efficient solution, employing an adaptive decentralised protocol for transmission and utilising relevant cryptographic technologies for other aspects.

The aforementioned scenarios represent some of the most prevalent avenues for the integration of blockchain technology with data trading. Of these, energy trading has been the subject of particularly extensive study, with research activity concentrated within the context of data trading within the Internet of Things (IoT). In general, regardless of the data trading scenario, the majority of solutions rely on cryptographic technologies, which are commonly referred to as soft environment solutions. These technologies include homomorphic encryption, asymmetric encryption, zero-knowledge proofs, and digital signatures.

Specifically, the integration of blockchain and zero-knowledge proof (ZKP) technologies has shown significant potential in privacy protection, security enhancement, and performance optimization. Refs. [21–23] delve into the combination of zk-SNARKs technology with blockchain, applied to account privacy protection and transaction verification for high privacy-demanding scenarios. These studies demonstrate how to achieve efficient and secure transaction verification mechanisms without disclosing transaction details, which is particularly applicable in finance and healthcare, where privacy requirements are extremely high. However, the performance overhead associated with the computation and verification processes of zk-SNARKs poses a significant challenge, especially in blockchain systems that need to handle a large volume of transactions. Refs. [24,25] further explore the integration of blockchain and ZKP in the Internet of Things (IoT) environment, proposing improvements to zk-SNARKs models to optimize performance, specifically by reducing proof sizes and shortening verification times to enhance the overall efficiency of blockchain systems. In addition, Ref. [26] introduces a method that utilizes ZKP to ensure privacy and fairness in a distributed IoT environment while maintaining system efficiency. Despite these studies revealing the considerable advantages of ZKP technology in protecting privacy within blockchain systems, it is evident that the limitations regarding performance overhead and trusted setup remain critical issues that require attention, and some work

has already been completed in this regard. Furthermore, current research predominantly focuses on specific application scenarios, lacking in-depth exploration of the adaptability, flexibility, and scalability of the integration of blockchain and ZKP technologies in various complex scenarios.

Moreover, despite the plethora of proposed solutions, the majority of existing research primarily concentrates on the protection of bidding information or the pricing of data within the context of data trading. There is a paucity of research focusing on the specific content of the trading data itself. Furthermore, insufficient attention has been paid to the security of the data itself during the transaction process, as well as the growing issues related to device resource constraints and scalability.

The data transaction security scheme proposed in this paper builds upon existing research in the field, specifically the above research on data transaction work in different scenarios. The model proposed in this paper integrates data transaction privacy protection and ownership guarantee in these scenarios. Furthermore, this paper focuses on the proof of data protection in data transactions, differentiating itself from the various types of data transactions that are unique to specific scenarios. It aims to satisfy the majority of scenarios, including the use of this scheme for the transaction of medical electronic health records (EHRs), for the transaction of transactional data between financial institutions, and for the transaction of data among manufacturers, suppliers, and logistic companies in the manufacturing industry, promoting distributed computing for other applications.

The purpose of this paper is to address the most challenging problem in data transactions, i.e., how to ensure that the whole process of transactions is completed fairly and securely. We will solve this problem by combining advanced technologies used in existing working scenarios, such as elliptic curve cryptography (ECC), non-interactive zero-knowledge proofs, smart contracts, and IPFS, accompanied by a lightweight model design and combining technologies such as off-chain storage to keep the computational overhead and cost at a relatively low level in response to possible blockchain scalability and performance issues.

## 3. Scheme Analysis

### *3.1. System Model*

In our system model, there are four types of entities: authority, IPFS, data seller, and data buyer. There also exists a class of execution tools smart contracts, as shown in Figure 1.

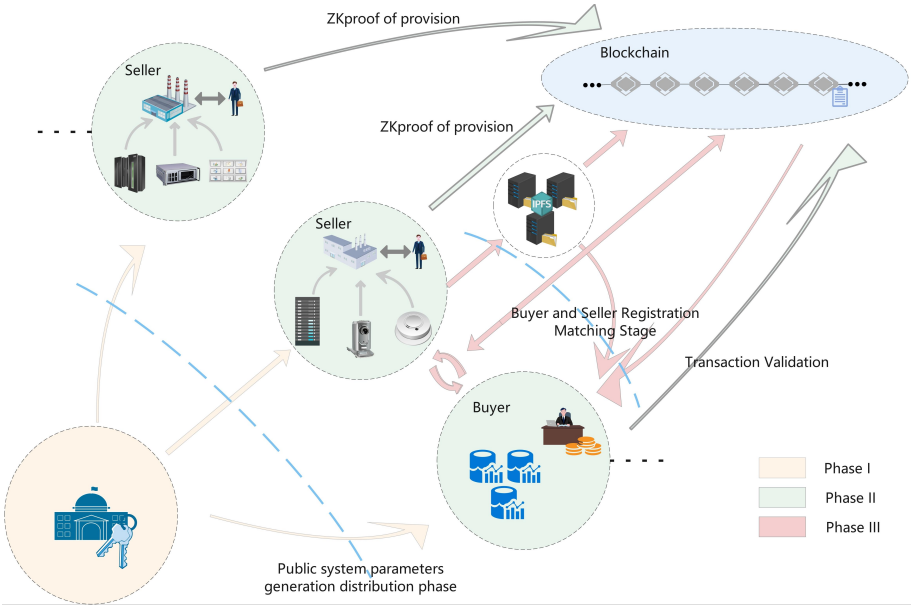



**Figure 1.** System model.

- Authority: The generation of zero-knowledge proof public parameters for data sellers, particularly those operating in resource-constrained environments such as the Internet of Things (IoT), can help to alleviate pressure on limited resources. Similarly, for data buyers, the generation of public system parameters and the publication of the buyer's public key can facilitate the transfer of data securely.
- IPFS: Relieves the storage pressure on the chain, saves the original data after double encryption of the seller, and provides a search function and encrypted data loading function for the buyer.
- Data Seller: The data seller is a user or agent who has data and wants to gain benefits from these data, but does not want to expose their original data information before the transaction is completed.
- Data buyer: A data buyer is a user or an agent who has a demand for these data; for example, a user under federated learning can obtain the demanded data through a data transaction to train his/her model.
- Smart contract: In the event that the pre-defined conditions are validated, the smart contract deployed on the blockchain will initiate its execution. Upon fulfilment of the stipulated conditions, the smart contract is capable of autonomously transferring funds, thereby ensuring the atomicity of the transaction.

The transaction is divided into three stages. In the initial stage, the authority generates two distinct types of public system parameters. Subsequently, the seller employs a zero-knowledge proof of these public parameters to generate their own proof, while the buyer returns their private key and publishes their public key. In the second stage, the data buyer and data seller invoke a matching smart contract on the blockchain. They input their attribute information such as data type, data size, and data price. The smart contract matches the best buyer and seller after receiving registration information. The third stage is the transaction phase. The data seller encrypts the original data twice: first with symmetric encryption, and then encrypting it with the buyer's publicly disclosed key, saving it on the IPFS, and returning the hash digest to the blockchain. The seller also uploads their zero-knowledge proof for verification. The buyer verifies the proof to ensure data integrity and correctness. Finally, the transaction is executed through the smart contract to ensure atomicity and fairness.

### *3.2. Threat Model*

In the full-process fair data transaction scheme we designed, we assumed that the authority and the storage center IPFS are credible, and in addition, that the smart contracts in the blockchain are generally executed strictly according to the predetermined logic, so they can also be recognized as credible. However, data sellers and data buyers, who are the subjects in the model, are considered to be likely to act maliciously in order to gain more benefits because they are highly influenced by human factors, i.e., it is assumed that there is malicious behavior on the part of data sellers and data sellers.

This is modeled and analyzed to obtain a threat model as follows:

Sellers with data: There may be a situation where a seller does not have some actual characteristic data needed by the buyer, but in order to gain benefits, the seller will try to generate a zero-knowledge proof that can be verified by data characteristics by certain means, so as to deceive the data buyer to pay the cost, i.e., an act of misinformation and fraud. For example, a data seller does not have real industrial IoT data, but in order to obtain payment, he generates a zero-knowledge proof that seems to be valid to deceive the data buyer. The data buyer pays and then realizes that the data is invalid, resulting in wasted resources and financial losses.

The buyer of the demanded data: The buyer of the data may refuse to pay the corresponding monetary equivalent of the data after acquiring the data through certain means and may infer the real identity of the seller through some malicious means, thus compromising the the seller's privacy to gain more benefits, i.e., refusal to pay and fraud.

Blockchain network: Due to the distributed nature of blockchain networks, they consist of various types of nodes, where malicious users and legitimate users are mixed together. Malicious users may attempt to analyze the range of data transactions recorded on the blockchain ledger, such as the transaction information of buyers or sellers, and use this information to infer private details of the users.

Other potential attack vectors include smart contract vulnerabilities. While smart contracts are generally considered trustworthy, their code may still contain unknown vulnerabilities, allowing malicious users to exploit these weaknesses. For instance, logical errors or improper handling of boundary conditions within the contract can be exploited by attackers, leading to financial losses or inconsistencies in the contract's transaction state. Another type of potential attack vector comprises zero-knowledge proof vulnerabilities. Although zero-knowledge proofs (ZKP) are highly effective in protecting privacy during transactions, there may be potential vulnerabilities in the design and implementation of ZKP protocols. ZKP protocols rely on complex mathematical algorithms, and any small error in the proof generation, verification, or parameter setting processes could result in exploitable vulnerabilities. Furthermore, some ZKP protocols require a trusted setup, meaning that the system parameter generation process must be fully secure. If any part of this trusted setup phase is compromised, attackers could generate keys or parameters capable of forging proofs, thereby compromising the security of the entire system.

### *3.3. Security Objectives*

Based on the traditional data transaction model, when designing a full-process fair data transaction model like the one in this paper, methods for how to prevent the above threats should be considered. The scheme should have these security attributes: fair exchange of data between the two parties participating in the data transaction, privacy protection for the data seller, and security of data transmission. Then, the security goals of the scheme design are as follows:

Fair exchange of data value: For both data buyers and sellers, in the case of malicious behavior, fair exchange needs to meet the requirement that the data seller can normally receive the equivalent value of the incentive on the basis of providing the correct data attributes requested by the other party. The data buyer needs to obtain correct and complete data that meets his/her demanded attributes while they pay the equivalent amount of the data value in monetary terms.

Data seller privacy protection: For data sellers, it is necessary that their private information will not be exposed, such as the real content of the original data, when the transaction has not been concluded or when proving the nature of the requested data. Steps must be taken to prevent other malicious nodes from obtaining the data seller's private information through various means across all aspects of the transaction.

Secure transmission of data: Ensuring the integrity, confidentiality, and reliability of data as it travels from seller to buyer is critical. Data is encrypted using symmetric encryption and/or asymmetric encryption to ensure that only authorized recipients can decrypt and access the data content.

Potential attack protection: To address vulnerabilities in smart contracts, code auditing and testing processes should be implemented to ensure that smart contracts are not affected by logical errors or unhandled boundary conditions during execution. For potential vulnerabilities in zero-knowledge proofs, we choose to use the widely validated zk-SNARKs protocol, which leverages its characteristics of uniqueness proof, integrity verification, and non-falsifiability to defend against potential attack vectors such as replay attacks, data tampering, and exploitation of protocol vulnerabilities .

## 4. Scheme Design

### *4.1. Overview*

For the data security trading scheme described in this paper, the complete process is primarily divided into three stages: the public system parameter generation and dis-

tribution stage, the registration and matching stage, and the transaction execution stage. Throughout this process, ensuring data privacy and secure exchange is the core objective, guaranteeing that data remains uncompromised at every stage. The following provides a detailed description of each stage and the corresponding privacy protection measures, and the specific process can be seen in Figure 2. To more clearly distinguish the three phases, a green arrow is used in the first phase to represent the key generation and distribution phase for the data trading group. A yellow arrow is used to indicate the zero-knowledge proof parameter generation phase for the data sellers. The second and third phases are represented by blue and red arrows, respectively.

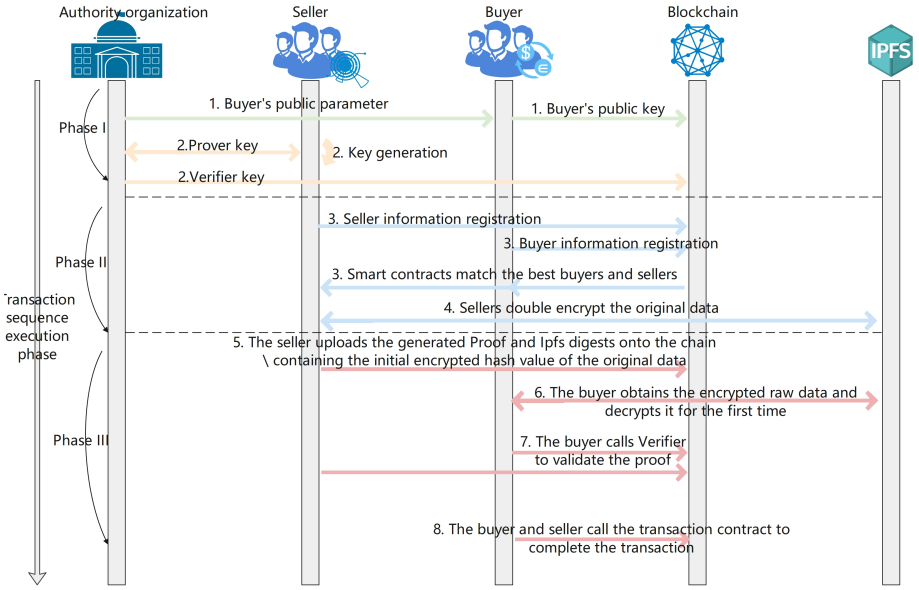



**Figure 2.** System flow chart.

a. In the public system parameter generation and distribution stage, an authority generates an asymmetric key pair for each buyer and returns the generated public and private keys to the buyer. The buyer will then upload their public key to the blockchain for the seller's use. Concurrently, the seller requests the system parameters necessary for zero-knowledge proofs from the authority based on their own parameters and receives a verification key. By ensuring that all key generation processes occur in a trusted environment and strictly controlling the distribution of keys, the buyer's privacy is effectively safeguarded, preventing potential key leaks.

b. In the registration and matching stage, both the seller group and the buyer group will register their information on the blockchain. This part involves the invocation of smart contracts, where both parties must fill in their data attribute information according to the provisions of the smart contract, without disclosing any raw data information. The smart contract will process the registration information and perform internal matching to identify suitable buyers and sellers. To further protect the seller's privacy, the seller uses their symmetric key to encrypt the raw data and generate a hash digest. Subsequently, the seller performs a secondary encryption using the publicly published buyer's public key and uploads the encrypted data to an off-chain storage IPFS center, returning a searchable hash digest for retrieval. This dual encryption mechanism and atomic attribute matching ensure that the seller's raw data remains undisclosed during the matching process.

c. In the transaction execution phase, the seller first generates the proof required for the buyer's verification using the zero-knowledge proof key generated in the first stage, and both the generated proof and the IPFS hash digest are recorded on-chain. The buyer retrieves the seller's uploaded data based on the on-chain IPFS index and uses their private key to perform the initial decryption of the data, obtaining the raw

data and generating a hash value for subsequent integrity verification. The execution rules of the transaction contract are pre-negotiated, and the buyer confirms the validity of the transaction contract based on the verification results of the zero-knowledge proof contract. This contract ensures the correctness and completeness of the data. Upon confirmation of the transaction, the buyer returns the agreed-upon currency to the seller, while the seller returns the key to the buyer. Through this series of processes, data privacy is effectively safeguarded at every stage of the transaction.

### *4.2. Detailed Programs*

#### 4.2.1. Phase I

In the data transaction scenario, in order to ensure that the key generation process is efficient and suitable for all kinds of resource-constrained devices under the premise of security, we choose an elliptic curve cryptography (ECC)-based encryption system to achieve higher security and smaller key sizes than traditional asymmetric encryption algorithms.

As shown in Figure 3, which illustrates the public system parameter generation and distribution phase, firstly, the authority selects a suitable elliptic curve $E$ and a large prime number $p$, and the elliptic curve is defined over a finite domain $GF(p)$. A generating element $G$ is chosen as the base point for generating the key pair, and the order of the generating element is $n$. $nG = O$ must be satisfied, where $O$ is the infinity point of the elliptic curve. After that, a private key $d_B$ (random large integer) is randomly generated for the buyer, making sure that it is complex enough to resist brute force decryption. We used a random number generator (CSPRNG) to create this and compute the public key $Q_B = d_B G$ for it in the set$(1, 2, \ldots, n-1)$. At the end of parameter generation, the public–private key pair is returned to the buyer.

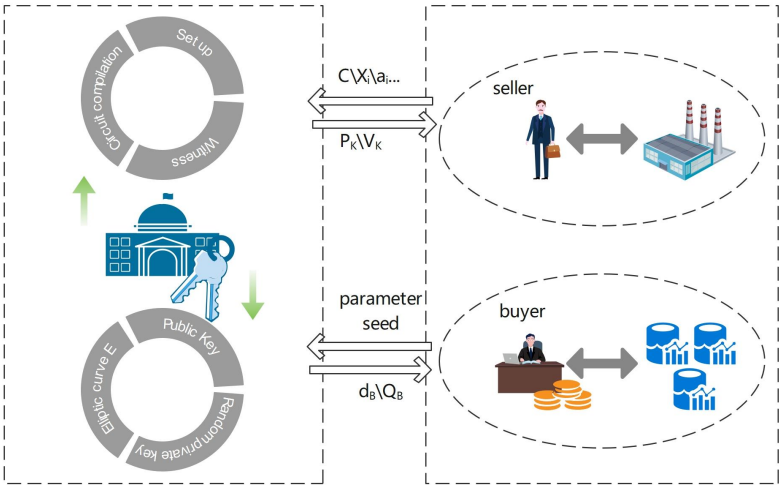



**Figure 3.** Security parameter distribution phase.

Secondly, based on the analysis of the zero-knowledge proof in the threat model, this paper selects zk-SNARKs (zero-knowledge succinct non-interactive argument of knowledge) as the zero-knowledge proof algorithm for implementation, primarily due to its significant advantages in terms of performance and security. Firstly, zk-SNARKs provide highly efficient verification times and compact proof sizes, making it well-suited for environments with stringent computational and bandwidth requirements. Additionally, zk-SNARKs offer strong security guarantees, including resistance to quantum attacks, which ensures a higher level of security for privacy-preserving systems. This algorithm has been widely adopted and extensively analyzed in cryptography, making it a reliable choice for securing transactions and ensuring privacy in the proposed design.

In the proposed scheme, the seller interacts with the authority, which uses its own deployed zk-SNARKs protocol to carry out the trusted setup phase, generating the public parameters of the system.

First, the system uses a random seed $s$ and a selected elliptic curve parameter set to generate the system's public parameters $\{G, H\}$, among others. Here, $G \in$ the elliptic curve parameter set, and $H = s \cdot G$ is another point generated based on the random number $s$. This parameter will be used by all participants in the zero-knowledge proof system.

Secondly, the parameters that sellers need to upload to the authority include:

- Computational logic circuit definitions, $C$ : For example, topological diagrams containing input wire definitions and logic operations.
- Input variables, $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$: These are the data attributes provided by the seller representing the values that need to be proven, such as data attributes collected by IoT sensor devices and case data from medical systems.
- Auxiliary information, $\mathbf{a} = \{a_1, a_2, \ldots, a_m\}$: These are the internal variables of the circuit, used to ensure the completeness of the computation.

Specifically, the circuit $C$ represents the computational logic that the seller wants to prove, such as proving that certain data processing (e.g., hash computation) was performed correctly. The circuit $C$ can be represented as a combination of gates through any Boolean or arithmetic circuit, and must satisfy the following condition: $C(x_1, x_2, \ldots, x_n) =$ true or false.

Finally, the authority generates the proving key <proving.key> based on the information uploaded by the data seller, which will be used by the seller to generate the subsequent proof. The formula is: <proving.key> $= H(C, x_i, a_i)$, where $H$ is a hash function or a mapping on the elliptic curve, ensuring the immutability of the input circuit and data.

The authority simultaneously generates the verification key <verification.key>, which is used for the on-chain verification process. The formula is: <verification.key> $= H(s, C, x_i)$, where the verification key is related to the system's generated random seed $s$ and ensures data integrity through hashing.

After the authority generates the keys and the zero-knowledge proof parameters, it returns the <proving.key> to the seller locally, which will be used later to generate the zero-knowledge proof. This proof will allow the data buyer to verify the claimed attributes through the blockchain platform. The <verification.key> is stored locally and will be used later for zero-knowledge proof contract generation when verification is required. At this point, the construction of the public parameters by the authority is complete.

#### 4.2.2. Phase II

After the authority generates the public parameters for the buyers and sellers, the buyers and sellers need to invoke the on-chain smart contracts to establish the best buyer and seller matching in the second stage. In our design, the smart contract acts as a marketplace function. As shown in Figure 4, we designed a smart contract that can support fine-grained matching based on the linear search matching function, which defines the basic requirements of the transaction and the basic attribute set of the data. The registration information $D_i$ can be as specific as the address of the buyer and seller, the amount of data, the type of data, the data threshold, and the price of the data, etc., and it can be changed according to actual buyer needs to achieve more specific and fine-grained matching.

After the buyer and seller groups have all registered their information, the smart contract will traverse all buyers, and for each buyer, traverse all sellers, looking for the seller that meets the data conditions preset by the buyer ahead of time. The seller's data thresholds should satisfy the buyer's threshold requirements, the seller's price requirements should meet the buyer's price thresholds, etc. If qualified sellers are found and the number of sellers is greater than or equal to two, the seller with the lowest price is selected as the best match. After successful matching, the MatchFound event is triggered and the matched buyer is removed from the buyer array to improve matching efficiency and avoid repeated matching, which satisfies the resource consumption problem in the vast majority of data transaction scenarios.

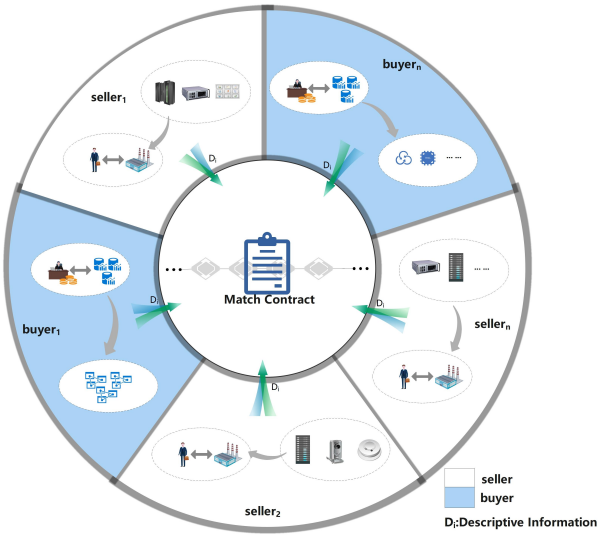



**Figure 4.** Matching of trading groups.

After the contract function successfully returns the best buyer and seller from the matching process, the seller first needs to encrypt the data for the first time. Here, the seller uses their own generated symmetric key to do so; the seller selects the key $h$ and divides the plaintext into 128-bit data blocks and fills them. The plaintext data blocks are encrypted and the final output is the ciphertext $M$. Then, the seller will obtain the buyer's public key $Q_B$ uploaded to the chain in the first stage, and assign it to the seller's initial encrypted data $M$ for $0 \leq M \leq n-1$. Subsequently, the seller selects a random number $k$, and $k \in \{1, 2, \ldots, n-1\}$ calculates the point:

$$X_1(x_1, y_1) = kQ, \tag{1}$$

The seller re-selects $k$the random number if $x_1 = 0$ and calculates the point:

$$X_2(x_2, y_2) = kG, \tag{2}$$

and then calculates the ciphertext:

$$C \equiv Mx_1 modn, \tag{3}$$

$(X_2, C)$ is to be used as the final ciphertext $M$.

After the seller performs double encryption, they will encapsulate the final ciphertext $(X_2, C)$ into a file and upload it to the storage center IPFS to get its returned hash value summary CID, which can be used for subsequent data lookup and access. The storage center will divide the file into fixed-size data blocks, typically 256 KB, to ensure more efficient file distribution and storage. Each data block is assigned a unique content identifier (CID), generated by hashing the block using a cryptographic hash function (SHA-256). This guarantees that each block has a unique identifier and ensures the integrity of the data block. The storage center also creates a Merkle Directed Acyclic Graph (Merkle DAG) structure from these blocks, using hash values to represent the data dependencies. The hash value of each file block is used to generate higher-level hash values, eventually forming the root hash (Root CID) of the entire file, which serves as the unique identifier for the file. Any request for this file will be made through this root hash, which facilitates subsequent file lookup and retrieval.

Because the storage center can distribute each block of the file across different nodes rather than storing it centrally on a single server, this decentralized storage enhances the system's redundancy and fault tolerance. This process guarantees the confidentiality and

integrity of the data during transmission and storage, and meets the strict requirements for data security and traceability in the data trading environment.

#### 4.2.3. Phase III

This phase is the execution phase of the transaction between the buyer and the seller after the successful matching. As shown in Figure 5, Following the completion of the first two phases of preparation, the data trading scenario will enter the final execution phase of the transaction. The seller will use the zero-knowledge proof <proving.key> proof key generated in the first stage combined with their own proof parameters to generate verifiable proofs $\Pi$, including the specific nature of the data, the initial encryption of the data hash value, and other items. The construction of the proof is usually based on the definition of circuit $C$, along with the seller's private input $x_i$. It is a non-interactive proof $\Pi$ that contains information about the correct execution of the circuit and does not reveal the specific value of $x_i$. The seller then uplinks both the generated proof file and the IPFS CID for the buyer to access. After the buyer obtains the CID, it will go to the IPFS to query, and after obtaining the correct return, it will capture the file locally, and then use its own key to decrypt it for the first time.

$$\Pi \leftarrow prove(< proving.key >, \{x_i\}, C, \{a_i\} \cdots >), \tag{4}$$

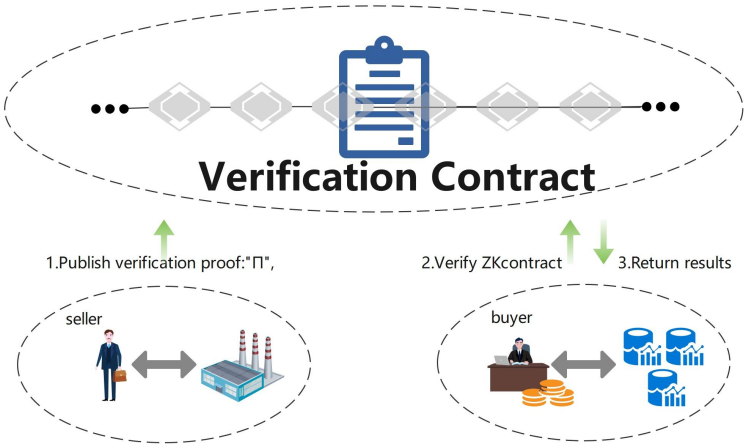



**Figure 5.** Transaction execution.

The buyer first calculates:

$$X_1(x_1, y_1) = dX_2 = d(kG) = k(dG) = kQ, \tag{5}$$

and subsequently can calculate to obtain, once encrypted, the ciphertext $M \equiv Cx_1^{-1} \mathrm{mod} n$. Subsequently, the buyer calls the pre-prepared ZK verification smart contract for transaction verification, and they use the seller-uploaded $\Pi$ as an input to the zero-knowledge proof contract to verify the nature of what the buyer wants to let the seller to prove. If the smart contract returns the correct response, it serves as proof that the seller's behavior is fair. In order to meet the relatively perfect requirements of full-process fairness and security, after the verification, the buyer can be assured based on the initial encrypted hash summary and the buyer's own, already obtained for comparison. If correct, this shows that the seller's behavior is safe and that the data transmission process is free of tampering.

$$true/false \leftarrow Verify(< verification.key >, \Pi), \tag{6}$$

After this, the buyer and seller jointly invoke the transaction contract, confirm the transaction according to the pre-determined rules, return the key that can unlock the data for the buyer, return the currency for the seller, and the transaction is successfully concluded.

Through the above operations, the transaction is successfully completed, ensuring the security and integrity of the data as well as the fairness of the transaction. In the data trading environment, this solution not only guarantees the security of information transmission and storage, but also ensures the rights and interests of all parties involved in the transaction, and improves the overall trust and transaction efficiency of the system. Ultimately, efficient, secure, and trustworthy data exchange and value transfer is realized.

## 5. Experimentation and Evaluation

### *5.1. Experimental Environment*

We implemented the proposed fair trade system and conducted a series of experiments to evaluate its performance. The system consists of two main components: a fine-grained matching module based on smart contracts, and a verification module based on zero-knowledge proofs and blockchain networks. For the verification module based on zero-knowledge proofs, the ZKP scheme is employed to perform several functions including witness generation, key generation, zero-knowledge proof generation, contract generation, and verification of contracts. These functions are programmed by using Zokrates 0.8.7, a toolkit for zero-knowledge proof implementations. The blockchain network is developed on the Ethernet public chain, using the Solidity language to implement smart contracts. We define multiple seller–buyers as participants in the blockchain network. The system has an Intel(R) Core(TM) i7-8700 CPU @ 3.20 GHz 3.19 GHz processor and 16.0 GB of RAM.

### *5.2. Experimental Results*

To validate the feasibility of the proposed scheme, we simulated a real-world industrial IoT data trading scenario. The experiment was conducted on a small-scale blockchain network consisting of 10 participant nodes, including eight data buyers and two data sellers. Each data seller managed a set of 10 to 50 different types of industrial IoT devices, such as temperature and humidity sensors or pressure monitoring devices, which generated continuous data streams. In this simulation, each data seller held between 30 to 50 sets of data based on the actual data generated by their devices, with each set containing 100 sensor readings or other industrial equipment data. Specifically, Figure 6 illustrates the amount of data held by buyers and sellers and their respective data demands, while Figure 7 shows the price demands of buyers and sellers.

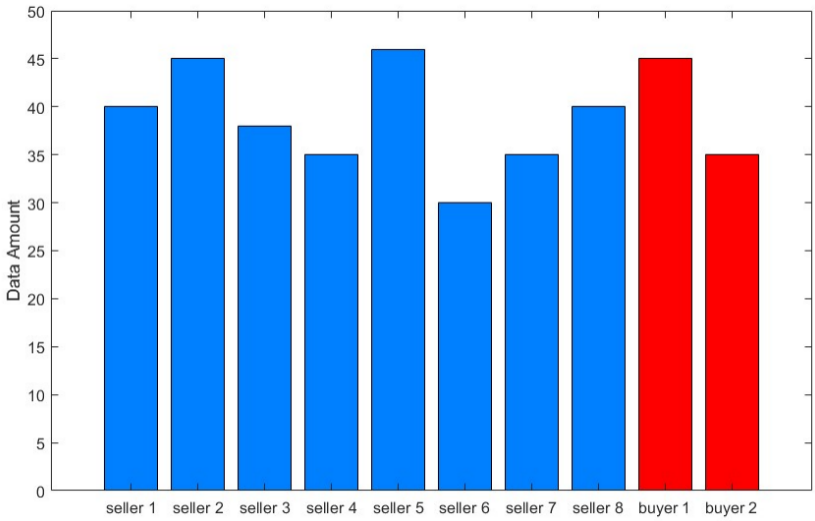



**Figure 6.** Data ownership requirements.

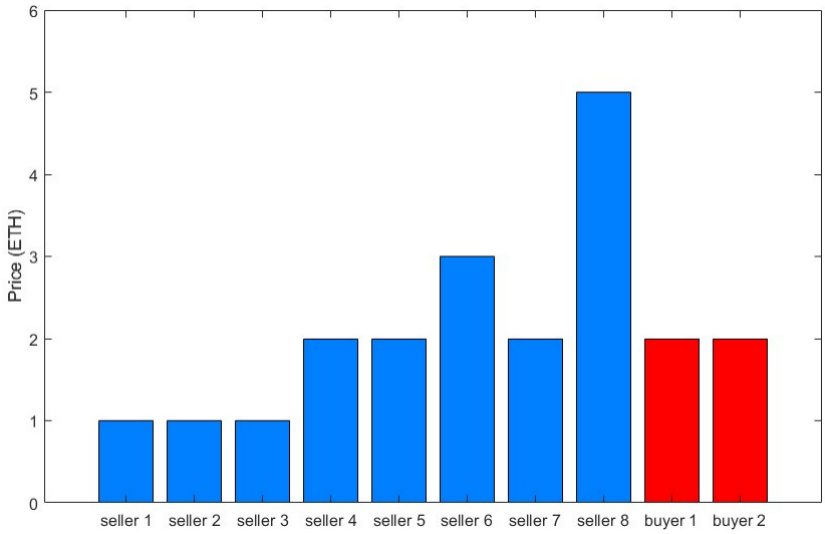



**Figure 7.** Data prices.

#### 5.2.1. Matching Simulation

In our simulation, we assumed that all users had different data preferences or data type requests. By using the linear search matching mechanism, the smart contract firstly compared and excluded specific buyer and seller groups based on data preferences and types, and then selected the buyer and seller groups with matching preferences. Secondly, the time threshold requirements and price demands of buyers and sellers were matched to obtain the best buyer and seller pairs.

In the pre-transaction phase, that is, the matching phase, the seller and buyer groups simultaneously called the registration matching contract deployed on the blockchain to register the data information they had. We assumed that the eight sellers possessed types of data containing information about temperature, humidity, wind speed in different amounts, and that the price of the demand for each type of data was also high and low. The two buyers were defined as having different types of data requirements, namely, temperature and humidity. Figure 8 shows the gas fee required by a group of ten buyers and sellers when calling the registered matching contract. In order to facilitate the calculation, we uniformly set the Gas Price to 0.001 Gwei. From Figure 8, it can be seen that the gas required by our setup contract was maintained in a stable and low range, and the cost of the transaction members was also kept in a relatively low range. Figure 9 shows the execution cost of the matching function after the transaction group has registered its information, and the execution cost of the function in matching suitable sellers and buyers, which is the same as the cost of registering the information, and because we set the matching function to remove a buyer after it has been matched with a suitable seller, we avoided wasted resources due to repeated matching. Figure 9 shows that the execution cost of the matching contract in the first two phases of the linear search for suitable sellers was relatively high. Figure 9 shows that the execution cost incurred by the matching contract in the first two phases of the linear search of the transaction population for suitable sellers is relatively high, and after matching two buyers with suitable sellers, the transaction cost plummeted to a very low level due to the completion of successful matches for the subject buyers. This will be maintained until new buyers appear.

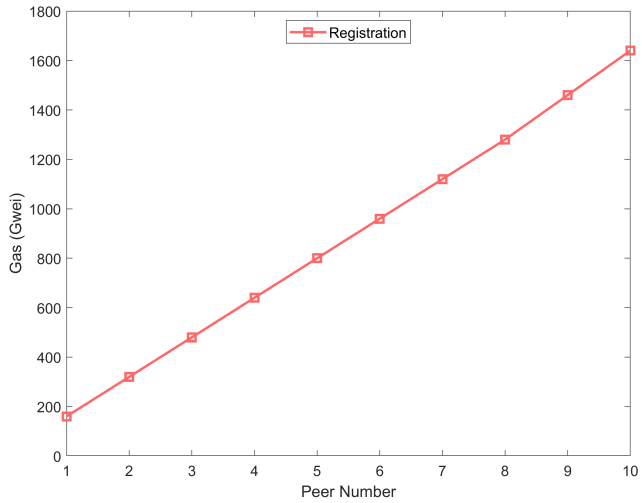



**Figure 8.** Registration cost.

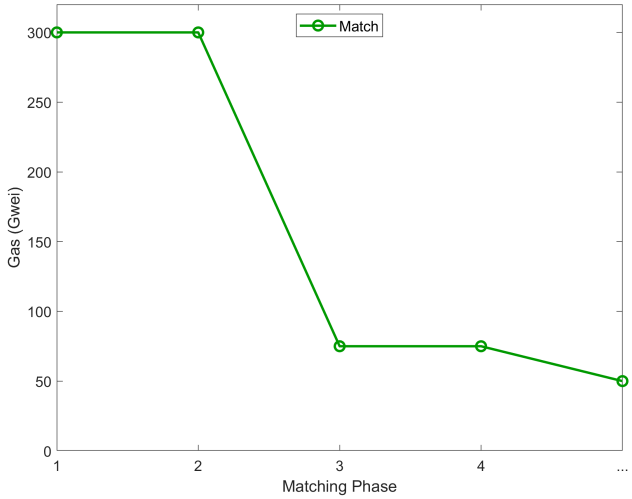



**Figure 9.** Matching phase costs.

#### 5.2.2. Zero-Knowledge Proof Simulation

After going through the matching phase, the seller needs to verify the zero-knowledge proof of their data. We verify the assumptions about the nature of the data required by the buyer, such as the buyer needing the seller to prove that all the data in the dataset is contained in a specific threshold, the seller needing to prove that the amount of data in the dataset and the sum of the data satisfies a certain requirement, and the seller ensuring that the data, after initial encryption, satisfies the hash image matching requirement, etc.

For these hypothetical requirements, we carried out the actual simulation of matching buyers and sellers on the chain. First of all, in the previous simulation, we obtained the blockchain addresses of buyers and sellers, and then we used their addresses to generate high-level DSL code in accordance with the nature of the various hypothetical requirements.

As demonstrated in Algorithms 1–3, we present three logical verification methods that data buyers can utilize: verifying that the values within the dataset fall within a specified range, verifying that the dataset size meets the expected range, and verifying that the dataset's hash is correct. These three verification methods ensure that the data seller's requirements regarding the dataset's attributes are met.

**Algorithm 1** Value Range Verification Process

**Input:** Array $a[N]$, $threshold_min$, $threshold_max$
**Output:** True or False

**for** each element $a[i]$ in array $a[N]$ **do**
  **if** $a[i] < threshold_min$ **or** $a[i] > threshold_max$ **then**
    **return** False
  **end if**
**end for**
**return** True

**Algorithm 2** Dataset Size Verification Process

**Input:** Array $a[N]$, $expected_size$
**Output:** True or False

**if** $N \neq expected_size$ **then**
  **return** False
**else**
  **return** True
**end if**

**Algorithm 3** Hash Verification Process

**Input:** Array $a[N]$, $hash$, $hash_function$
**Output:** True or False

$computed_hash \leftarrow hash_function(a)$
**if** $computed_hash \neq hash$ **then**
  **return** False
**else**
  **return** True
**end if**

This represents the process of constructing the validation properties of the premise assumptions, and in the process of constructing the DSL code for several properties, it can be found that the size of the DSL code is maintained at a very small level. The DSL code is compiled, witnessed, and proofed, resulting in a provable Proof.Json file that can be exported to the contract and then verified on the chain, as shown in Figure 10, where other participants can verify the smart contract's commitment to the industrial data in each match.

Figures 11 and 12 show the Solidity contract and verification result output corresponding to one of the verification logics. Upon input validation, the contract generates '0x0000...00001', which signifies successful verification.

We have uplinked the validated contracts generated for several properties, and the Table 1 shows the size of the corresponding validation file and the corresponding DSL code generated for each property, which shows a trend in size that renders these more adaptable for the end-side nodes in industrial IoT scenarios. As shown in Figure 13, we have verified a total of four properties of contracts and divided the data into two parts: deployment cost and verification cost. Figure 13 demonstrates that the deployment cost increases with the number of contracts in a linear trend, and the execution cost required for the verification of each contract stabilizes at a smaller value.

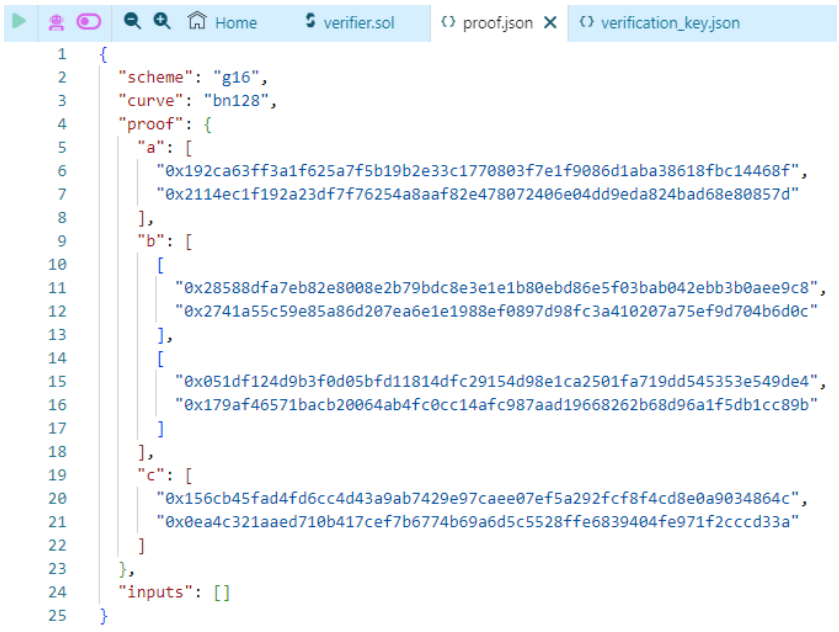



**Figure 10.** Proof file.

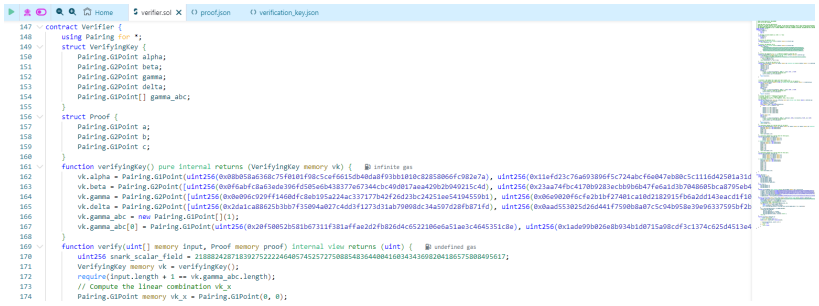


**Figure 11.** Verification of Contract Overview.




**Figure 12.** Verifier result.

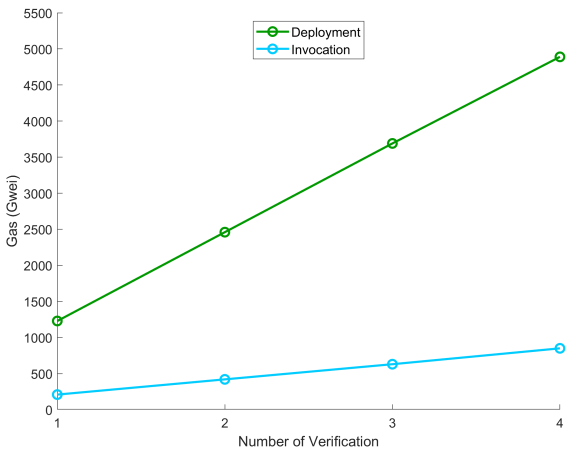



**Figure 13.** Verification cost.

**Table 1.** Verification sizes for different characteristics.

| | **Characteristic 1** | **Characteristic 2** | **Characteristic 3** | **Characteristic 4** | ... |
|---|---|---|---|---|---|
| **DSL Code** | 330 bytes | 324 bytes | 274 bytes | 285 bytes | ... |
| **Verification Size** | 10.3 kb | 10 kb | 9.63 kb | 9.82 kb | ... |

Secondly, we compared the gas consumption required for different input data with the BZDIMS system [27], as shown in Table 2. BZDT represents the execution costs of our proposed scheme under three different input data sizes: 100 bytes, 200 bytes, and 300 bytes. The results indicate that the costs of the two schemes are roughly similar for different data inputs, demonstrating the strong stability of our approach.

Additionally, we conducted a comparative analysis of the gas used in our scheme with another relatively mature solution: the Aztec protocol, which is based on ZK Rollup technology. The fixed gas cost for a batch of ZK Rollup transactions is approximately 500,000 gas. In comparison, the execution cost of the verification contract for data sellers in our scheme is about 200,000 gas. However, due to the particular nature of the Aztec protocol, its cost applies to a batch of transactions, meaning the cost of ZK Rollup technology is significantly lower than that of our scheme. Nevertheless, in highly privacy-sensitive data trading scenarios, sellers may only need to prove the attributes of their data, and the trading groups often exhibit complex and dynamic changes. Our scheme, with its fixed cost structure, offers a more flexible proof generation method while ensuring both privacy and accuracy.

**Table 2.** Gas used Comparison for different data sizes.

| **Data (Bytes)** | **Gas Used** | |
|---|---|---|
| | **zk-SNARKs** | |
| | **BZDT** | **BZDIMS** |
| **100** | 149,675 | 153,859 |
| **200** | 187,924 | 195,061 |
| **300** | 208,967 | 215,663 |

We conducted a transaction simulation for industrial IoT data, and based on the simulation results of this experiment, it can be inferred that the scheme has high generality and can be applied to other scenarios. For example, in medical scenarios, the model can build a transaction bridge between hospitals and patients. First, both parties register their information through a common transaction platform and find the best buyer and seller through a fine-grained matching algorithm. Next, double encryption is utilized to protect the patient's private information, while the authenticity of patient-provided data, such as consultation information and results, is verified through non-interactive zero-knowledge proof. In the financial industry, the model enables credit data transactions between banks and credit bureaus, and the scheme's efficient matching and automated transaction mechanisms can significantly reduce transaction costs. In the manufacturing industry, the model can be used for transaction sharing of supply chain data. Manufacturers and suppliers register and match through the platform, and utilize encryption and zero-knowledge proof technology to protect trade secrets, making data transactions in each link of the supply chain safer and more efficient, while keeping transaction costs within a low range.

In conclusion, the scheme is highly versatile and practicable in different scenarios, which not only enhances the security and privacy protection of data transactions, but also reduces the transaction costs through efficient matching algorithms and automated transac-

tion mechanisms, and provides a solution to the subsequent scalability problems that may exist in terms of the continuous growth of users through off-chain storage.

Of course, we have also noted that current blockchain systems often face limitations in transaction processing speed and throughput when handling large-scale data transactions. The computational complexity of zk-SNARKs is another challenge that needs to be addressed. While IPFS performs well in terms of data storage and access, it still encounters challenges such as network latency and node availability when dealing with large-scale data storage. These challenges require further optimization and improvement. In the future, we plan to introduce sharding technology, Layer 2 solutions, zk-SNARKs optimization models, and IPFS node expansion to further enhance the overall performance and scalability of the solution, ensuring its effectiveness in handling larger-scale data transactions.

## 6. Conclusions

In this paper, we propose a data transaction security mechanism that combines smart contracts and staged non-interactive zero-knowledge proof techniques to solve the fairness and security problems that exist in data transactions in distributed systems. We provide the amount of underlying data as well as the execution cost of the matching and proof phases in our proposed modeling scheme, demonstrating the low cost and feasibility of the model. Meanwhile, the scalability of the system in terms of handling large amount of data and a growing number of users is enhanced using an off-chain storage technique. The blockchain plays the role of a verifier in the authentication scheme, reducing unnecessary communication overhead. The participating nodes of the experimental setup are able to verify the correctness of the prover's statement through a contract in the blockchain. In our future research, we will focus on continuing to optimize the computational performance of zero-knowledge proofs by reducing their computational complexity to better accommodate resource-constrained devices. In addition, we will explore the applicability of the system in more application scenarios and further integrate advanced privacy protection techniques to enhance the level of privacy protection during data transactions.Concurrently, an investigation will be conducted into the potential for blockchain to be integrated with other distributed architectures, with the objective of enhancing the flexibility and compatibility of the system.

**Author Contributions:** Conceptualization, B.Z. and H.P.; methodology, B.Z.; formal analysis, B.Z., H.P. and K.L.; writing—original draft preparation, B.Z.; writing—review and editing, B.Z., H.P. and Y.X.; writing—review, J.W., D.F. and W.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded in part by the National Key Research and Development Project of China under Grant 2023YFB2703603, Key Research Project for Higher Education of Henan Province 24A520059, Strength Enhancement Plan for Advantageous Disciplines of Zhongyuan University of Technology GG202425, and 2024 Scientific and Technological Research Project in Henan Province 242102210136.

**Data Availability Statement:** Data are contained within the article.

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

## References

1. Liu, J.; Zhang, J.; Song, S.H.; Letaief, K.B. Client-Edge-Cloud Hierarchical Federated Learning. In Proceedings of the ICC 2020—2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
2. Farahani, B.; Tabibian, S.; Ebrahimi, H. Toward a Personalized Clustered Federated Learning: A Speech Recognition Case Study. *IEEE Internet Things J.* **2023**, *10*, 18553–18562. [CrossRef]
3. Zhao, Y.; Zhao, J.; Jiang, L.; Tan, R.; Niyato, D.; Li, Z.; Lyu, L.; Liu, Y. Correction to "Privacy-Preserving Blockchain-Based Federated Learning for IoT Devices". *IEEE Internet Things J.* **2023**, *10*, 973. [CrossRef]
4. Fan, S.; Zhang, H.; Zeng, Y.; Cai, W. Hybrid Blockchain-Based Resource Trading System for Federated Learning in Edge Computing. *IEEE Internet Things J.* **2021**, *8*, 2252–2264. [CrossRef]

5. Wang, X.; Hu, J.; Lin, H.; Liu, W.; Moon, H.; Piran, J. Federated Learning-Empowered Disease Diagnosis Mechanism in the Internet of Medical Things: From the Privacy-Preservation Perspective. *IEEE Trans. Ind. Inform.* **2022**, *19*, 7905–7913. [CrossRef]
6. Chen, C.; Wu, J.; Lin, H.; Chen, W.; Zheng, Z. A Secure and Efficient Blockchain-Based Data Trading Approach for Internet of Vehicles. *IEEE Trans. Veh. Technol.* **2019**, *68*, 9110–9121. [CrossRef]
7. 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]
8. Zhang, M.; Liu, J.; Feng, K.; Beltran, F.; Zhang, Z. SmartAuction: A blockchain-based secure implementation of private data queries. *Future Gener. Comput. Syst.* **2022**, *138*, 198–211. [CrossRef]
9. Pop, C.D.; Antal, M.; Cioara, T.; Anghel, I.; Salomie, I. Blockchain and Demand Response: Zero-Knowledge Proofs for Energy Transactions Privacy. *Sensors* **2020**, *20*, 5678. [CrossRef]
10. Xue, L.; Ni, J.; Liu, D.; Lin, X.; Shen, X. Blockchain-Based Fair and Fine-Grained Data Trading With Privacy Preservation. *IEEE Trans. Comput.* **2023**, *72*, 2440–2453. [CrossRef]
11. Hopwood, D.; Bowe, S.; Hornby, T.; Wilcox, N. *Zcash Protocol Specification*; GitHub: San Francisco, CA, USA, 2016.
12. Jiang, S.; Li, J.; Zhang, X.; Yue, H.; Wu, H.; Zhou, Y. Secure and Privacy-Preserving Energy Trading With Demand Response Assistance Based on Blockchain. *IEEE Trans. Netw. Sci. Eng.* **2023**, *11*, 1238–1250. [CrossRef]
13. Aitzhan, N.Z.; Svetinovic, D. Security and Privacy in Decentralized Energy Trading Through Multi-Signatures, Blockchain and Anonymous Messaging Streams. *IEEE Trans. Dependable Secur. Comput.* **2018**, *15*, 840–852. [CrossRef]
14. Son, Y.-B.; Im, J.-H.; Kwon, H.-Y.; Jeon, S.-Y.; Lee, M.-K. Privacy-Preserving Peer-to-Peer Energy Trading in Blockchain-Enabled Smart Grids Using Functional Encryption. *Energies* **2020**, *13*, 1321. [CrossRef]
15. Daghmehchi Firoozjaei, M.; Ghorbani, A.; Kim, H.; Song, J. Hy-Bridge: A Hybrid Blockchain for Privacy-Preserving and Trustful Energy Transactions in Internet-of-Things Platforms. *Sensors* **2020**, *20*, 928. [CrossRef]
16. Kumar, R.; Kumar, P.; Tripathi, R.; Gupta, G.P.; Islam, A.K.M.N.; Shorfuzzaman, M. Permissioned Blockchain and Deep Learning for Secure and Efficient Data Sharing in Industrial Healthcare Systems. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8065–8073. [CrossRef]
17. Zhang, X.; Li, X.; Miao, Y.; Luo, X.; Wang, Y.; Ma, S.; Weng, J. A Data Trading Scheme With Efficient Data Usage Control for Industrial IoT. *IEEE Trans. Ind. Inform.* **2021**, *18*, 4456–4465. [CrossRef]
18. Dixit, A.; Singh, A.; Rahulamathavan, Y.; Rajarajan, M. FAST DATA: A Fair, Secure, and Trusted Decentralized IIoT Data Marketplace Enabled by Blockchain. *IEEE Internet Things J.* **2021**, *10*, 2934–2944. [CrossRef]
19. Zhou, Z.; Tian, Y.; Xiong, J.; Ma, J.; Peng, C. Blockchain-Enabled Secure and Trusted Federated Data Sharing in IIoT. *IEEE Trans. Ind. Inform.* **2022**, *19*, 6669–6681. [CrossRef]
20. Huang, C.; Liu, D.; Ni, J.; Lu, R.; Shen, X. Achieving Accountable and Efficient Data Sharing in Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2020**, *17*, 1416–1427. [CrossRef]
21. Guan, Z.; Wan, Z.; Yang, Y.; Zhou, Y.; Huang, B. BlockMaze: An Efficient Privacy-Preserving Account-Model Blockchain Based on zk-SNARKs. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 1446–1463. [CrossRef]
22. Singh, M.; Aujla, G.S.; Bali, R.S. A Deep Learning-Based Blockchain Mechanism for Secure Internet of Drones Environment. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 4404–4413. [CrossRef]
23. Huang, H.; Zhu, P.; Xiao, F.; Sun, X.; Huang, Q. A blockchain-based scheme for privacy-preserving and secure sharing of medical data. *Comput. Secur.* **2020**, *99*, 102010. [CrossRef]
24. Li, Y.; Tao, X.; Zhang, X.; Liu, J.; Xu, J. Privacy-Preserved Federated Learning for Autonomous Driving. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 8423–8434. [CrossRef]
25. Sun, X.; Yu, F.R.; Zhang, P.; Sun, Z.; Xie, W.; Peng, X. A Survey on Zero-Knowledge Proof in Blockchain. *IEEE Netw.* **2021**, *35*, 198–205. [CrossRef]
26. Li, Y.; Susilo, W.; Yang, G.; Yu, Y.; Liu, D.; Du, X.; Guizani, M. A Blockchain-Based Self-Tallying Voting Protocol in Decentralized IoT. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 119–130. [CrossRef]
27. Yang, X.; Li, W. A zero-knowledge-proof-based digital identity management scheme in blockchain. *Comput. Secur.* **2020**, *99*, 102050. [CrossRef]


**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.