Journal of Cloud Computing

RESEARCH Open Access

A trusted IoT data sharing method based on secure multi-party computation

Li Ma[1*], Binbin Duan[1], Bo Zhang[1], Yang Li[1], Yingxun Fu[1] and Dongchao Ma[1*]

Abstract
Edge computing nodes close to the perception layer of IoT systems are susceptible to data leaks and unauthorized access. To address these security concerns, this paper proposes a trusted IoT data sharing method based on secure multi-party computation (SMC). By running a reliable third-party blockchain service at edge computing nodes, the data computation relationships between IoT devices in the perception layer are registered in blockchain smart contracts. This constructs a publicly verifiable IoT data sharing method combining on-chain audit verification and off-chain SMC. Furthermore, a Bloom filter is maintained at the on-chain smart contract layer to track the trust status of IoT devices in the perception layer, filtering out non-trustworthy device requests and enabling secure data sharing among trusted devices. Comparative analysis and performance tests demonstrate the proposed method's high computational efficiency for IoT device nodes.

Keywords IOT, Blockchain, Edge computing, Secure multi-party computing, Bloom filter

Introduction

The storage and sharing process of massive real-time data of the Internet of Things based on edge computing is carried out at the device end or edge end of the sensor network to meet the requirements of low latency. However, due to the limitation of edge device resources, this scheme cannot support various complex algorithms to ensure data security. For example, sensitive data on edge devices will be processed by edge computing servers with incomplete security protection, which will increase the risk of IoT data disclosure [1], untrusted edge nodes may become the entrance for attackers to invade the network [2], and frequent communication and data transmission between intermediate devices are vulnerable to man-in-the-middle attacks [3]. In order to solve the above problems, the existing security solutions in the edge computing environment include encryption and IoT data protection, edge node reliability assessment, security authentication and authentication, enhanced network security measures, decentralized and distributed architectures, secure communication protocols, and security updates and vulnerability management. However, each of these schemes has its own shortcomings, such as the increased overhead of encryption and decryption processes, the risk of miscalculation in the determination of trusted nodes, the increased complexity of the use of strong authentication, the failure of network security measures to eliminate all attack risks, the introduction of complexity in decentralized architectures, the possibility of attacks on secure communication protocols, and the possibility of new problems introduced by security updates. Therefore, trade-offs need to be integrated in practical applications, and continuous improvement and innovation to improve the security of the edge computing environment.

*Correspondence:
Li Ma
mali@ncut.edu.cn
Dongchao Ma
mail@ncut.edu.cn
[1]School of Information, North China University of Technology, Beijing 100144, China

This paper utilizes technologies such as blockchain, secure multi-party computation, and Bloom filters to address the issue of secure data sharing between devices in the IoT perception layer. It proposes a trusted IoT data sharing method based on secure multi-party computation. This method transforms the data exchange problem between IoT devices into a secure multi-party computation feedback control problem. By leveraging technologies such as Pedersen [4] commitments, Shamir's secret sharing [5], and blockchain, it constructs a publicly verifiable IoT data sharing protocol that combines on-chain audit verification and off-chain secure multi-party computation. Based on the audit verification results of the protocol behavior executed by perception layer devices, a Bloom filter is constructed to trace the trust status of perception layer devices. This filter screens out requests from devices with low trust values in the perception layer, ensuring the normal operation of trusted IoT devices in the perception layer, thereby enhancing the security and reliability of data sharing between IoT devices.

Related work

Blockchain and edge computing, along with secure multi-party computing, are three compelling technologies that have emerged and gradually become key drivers in the IoT space.

Blockchain technology, with its decentralized, immutable and trusted characteristics, provides a secure and reliable data exchange and sharing platform for the Internet of Things. There are many researches on the application of blockchain in the field of Internet of Things. For example, Yuan Liu et al. [6]. proposed the application of blockchain in the medical IoT physical network system. The authors also proposed a decentralized blockchain-based reputation system, providing a new secure data-sharing solution for IoT data sharing [6]. Yuan Liu et al. [7]. further proposed a blockchain-based spatial crowdsourcing service (BlockSC) and verified the effectiveness of its task matching scheme. but most of their research focused on identity authentication and access control [8, 9]. Ouaddah A et al. [10] proposed a blockchain-based IoT data sharing framework for protecting IoT data in the Internet of Things. The research team ensures the security and IoT data sharing by using blockchain technology for data encryption, anonymity, and access control. Liu CH et al. [11] realized blockchain-based secure data sharing in the IoT environment and introduced a fine-grained access control mechanism. By utilizing the characteristics of blockchain to ensure data integrity, trustworthiness and immutability, and adopting fine-grained access control policies, the security and IoT data are protected. This research provides new solutions for secure data sharing in the IoT field.

Edge computing technology pushes computing and data processing capabilities to the edge of the network, enabling lower latency and more efficient data processing and decision making. Liang X et al. [12] proposed a secure data sharing architecture based on blockchain and edge computing. By using edge devices as data processing nodes and leveraging blockchain technology to ensure that data is secure and trusted, secure and efficient data sharing is achieved in the IoT environment.

Secure multi-party computing technology protects IoT data by allowing data to be encrypted and desensitized before being processed. By using secure multi-party computing technology on edge devices, data can be processed at the source of data collection, avoiding the transmission of raw data to a central server and reducing the risk of data leakage. Yang Y et al. [13] proposed a collaborative intrusion detection system based on secure multi-party computing for network security monitoring in the Internet of Things. By applying secure multi-party computing to intrusion detection algorithm, it realizes secure data sharing and collaboration among multiple parties and improves the network security of IoT.

Although the above research has improved the security of the Internet of Things, the content of the research is only aimed at specific security threats, and it is difficult to ensure security in the face of a threat model with multiple attacks. For example, the core feature of blockchain technology is open and transparent, and all data sharing can be viewed by every authorized node on the network. However, this leads to a situation that contradicts the protection of privacy [14–16]. In some application scenarios, users may need to protect their identity and transaction details, but the transparency of the blockchain makes this information easy to infer and analyze. Edge computing involves processing and storing data locally, rather than sending it to the cloud. This can lead to an increased risk of data being stolen, tampered with or leaked during transmission [17–19]. Secure multi-party computing requires data to be transferred to a compute node or service provider for processing, so the security of data is highly dependent on the trust of the compute node or service provider. If a compute node or service provider has a security breach or is attacked, it may lead to data leakage or tampering, which threatens the security and data of IoT systems.

Design of IoT data sharing protocol

This study aims to design a flexible and secure trusted data sharing method for the Internet of Things (IoT), leveraging the technological advantages of edge computing, secure multi-party computation (SMC), and blockchain to address the information silo problem among IoT devices. To achieve this goal, a trusted third-party service, namely an authorized blockchain network, is introduced into the edge servers.

As illustrated in Fig. 1, using the example of the autonomous vehicle “Luobo Kuaipao,” the vehicle, smart traffic signals, additional parking lots, and mobile phones securely and reliably share IoT data in real-time. The IoT devices initiating tasks first register the computation relationships needed among participating IoT devices in the form of arithmetic circuits to the blockchain smart contract. This registration provides an interface for IoT data sharing. Although the third-party blockchain network is considered trustworthy, it does not have access to the complete plaintext information of the IoT data.

Protocol execution process

As shown in Fig. 2, if the IoT device x of the task initiator wants to share IoT data with the participants of the arithmetic gate by invoking the interface, it must first determine that the device is a trusted device after passing the identity review on the blockchain chain and Blum filter verification according to the device Id and PKI certificate, and perform the secure multi-party calculation task of the arithmetic gate on the blockchain. Wake up each IoT device participant of the arithmetic gate circuit and ask the participant’s IoT device to provide its IoT data information (IoT data information includes: The local IoT device uses Shamir secret sharing to cut its own IoT data into n secret shards (n is the number of arithmetic gate participants) and generate Pedersen promises of n secret shards. The Paillier public key of the arithmetic gate participant is used to encrypt the n secret fragments and n blind factors that generate Pedersen promise, and n ciphertext fragments and n blind factor ciphertext are obtained. The IoT data information of all participants is distributed twice on the chain, and the IoT data information after the secondary fragmentation is distributed to the off-chain IoT device participants, and then the off-chain IoT device participants submit the Pedersen commitment of revenue secret sharding, and the coarse-grained audit on the chain verifies whether the total revenue expenditure is balanced. After the verification is passed, the on-chain smart contract layer sends calculation instructions to the participants of the off-chain IoT device according to the arithmetic gate circuit, and the IoT devices of each participant perform calculation, obtain the arithmetic gate circuit result fragment according to the calculation result of the secret fragment, and submit the result fragment to the on-chain to recover the execution result of the arithmetic gate circuit on the chain. Finally, the on-chain smart contract returns the result to the IoT device of the task initiator.

In the above process, Pedersen commitment, Paillier homomorphic encryption and Shamir secret sharing are the core algorithms to ensure on-chain and off-chain cooperation. In the IoT data sharing method based on secure multi-party computing, the principle and implementation process of these algorithms are as follows:

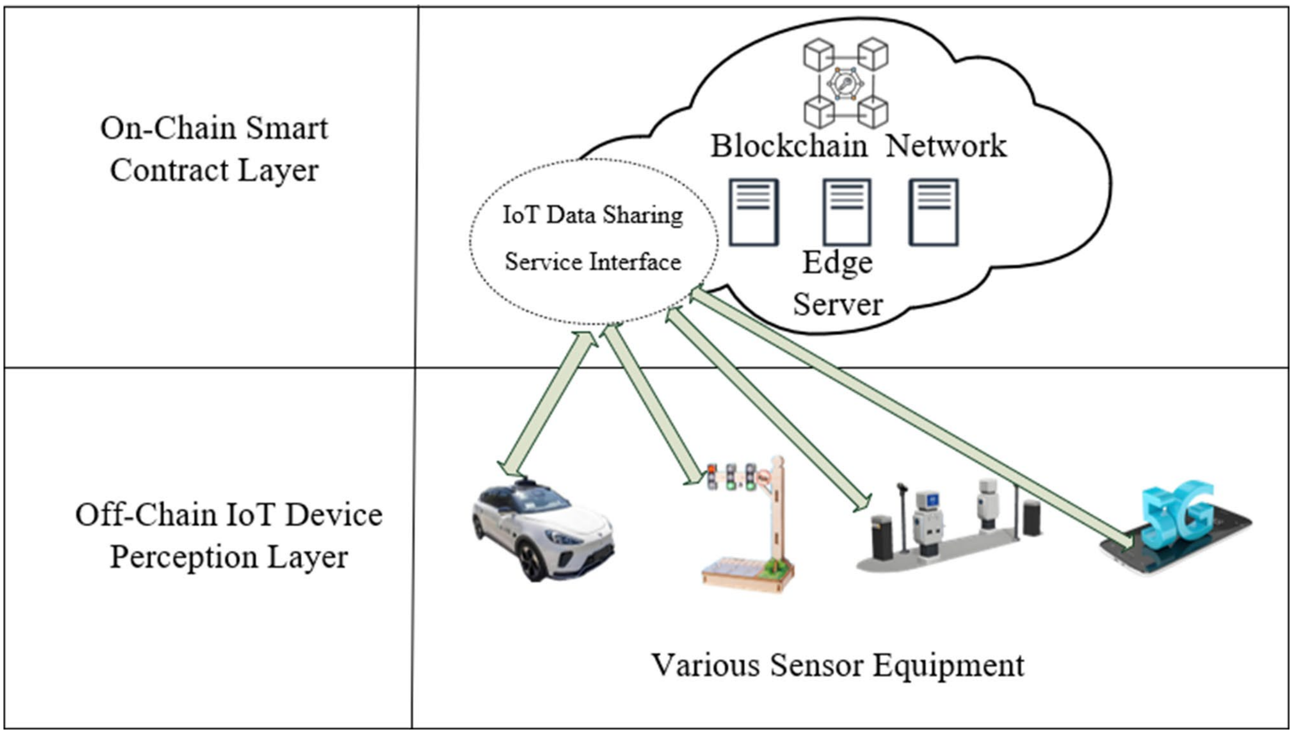
On-Chain Smart Contract Layer
Blockchain Network
IoT Data Sharing Service Interface
Edge Server
Off-Chain IoT Device Perception Layer
Various Sensor Equipment

Fig. 1 Overall architecture

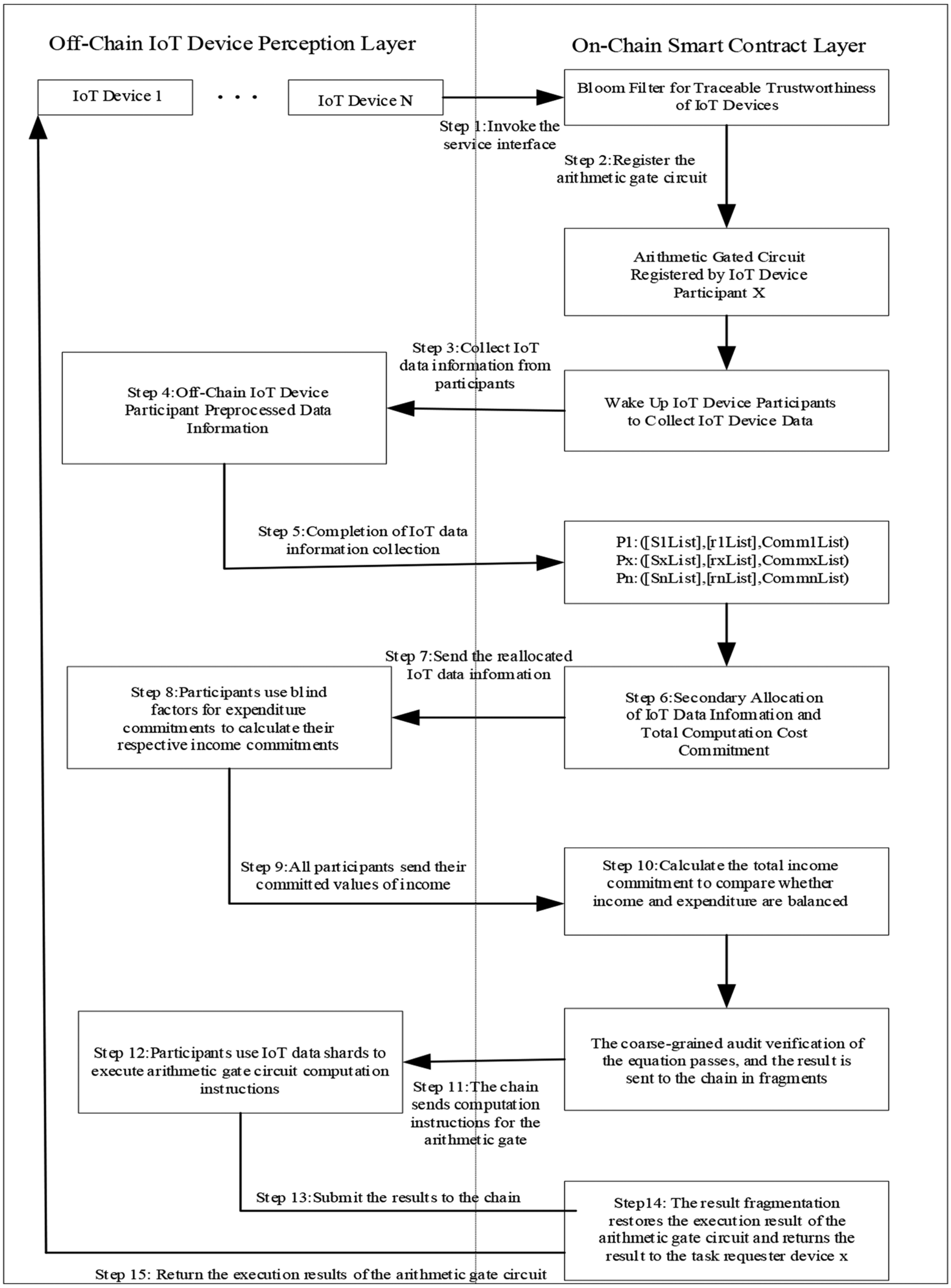
Off-Chain IoT Device Perception Layer
On-Chain Smart Contract Layer
IoT Device 1
IoT Device N
Bloom Filter for Traceable Trustworthiness of IoT Devices
Step 1:Invoke the service interface
Step 2:Register the arithmetic gate circuit
Arithmetic Gated Circuit Registered by IoT Device Participant X
Step 3:Collect IoT data information from participants
Step 4:Off-Chain IoT Device Participant Preprocessed Data Information
Wake Up IoT Device Participants to Collect IoT Device Data
Step 5:Completion of IoT data information collection
P1:([S1List],[r1List],Comm1List)
Px:([SxList],[rxList],CommxList)
Pn:([SnList],[rnList],CommnList)
Step 7:Send the reallocated IoT data information
Step 8:Participants use blind factors for expenditure commitments to calculate their respective income commitments
Step 6:Secondary Allocation of IoT Data Information and Total Computation Cost Commitment
Step 9:All participants send their committed values of income
Step 10:Calculate the total income commitment to compare whether income and expenditure are balanced
Step 12:Participants use IoT data shards to execute arithmetic gate circuit computation instructions
The coarse-grained audit verification of the equation passes, and the result is sent to the chain in fragments
Step 11:The chain sends computation instructions for the arithmetic gate
Step 13:Submit the results to the chain
Step14: The result fragmentation restores the execution result of the arithmetic gate circuit and returns the result to the task requester device x
Step 15: Return the execution results of the arithmetic gate circuit

Fig. 2 Execution flow of IoT data sharing service

Pedersen Commitments: Pedersen commitments are a cryptographic scheme used to conceal the value of data while ensuring its integrity and immutability. In the trusted secret sharing method studied in this paper, the distribution and reception of IoT data fragments among participating IoT devices during a round of data sharing are viewed as a data transaction. Each participant submits the ciphertext fragments of the IoT data to the consensus layer, which is regarded as the expenditure of a data transaction. The receipt of ciphertext fragments from the consortium blockchain consensus layer by each participant is considered as the income of a data transaction. Each participant generates their Pedersen commitments for expenditure and income and submits them to the on-chain smart contract layer. The on-chain smart contract audits and verifies the legality of the IoT data sharing by calculating whether the data transaction income and expenditure are balanced.

Assume there are n IoT device participants, and participant x selects n blinding factors $r_{xi} \in F_n \ (i = 1,2, \ldots, n)$. During the expenditure phase, participant x calculates the commitment values $Comm_{xList}$ of the n secret fragments and submits them as part of the IoT data information to the on-chain smart contract layer. The Pedersen commitment for generating the IoT data fragments $S_{xi} \ (i \in 1,2, \ldots, n)$ involves the following two steps:

Step 1: Choose a cyclic subgroup G of a large prime order q, and select two generators g and h from G. The public tuple (g, h, q) serves as the system's public parameters.

Step 2: Commitment generation phase: The committing party selects a random number r as the blinding factor and computes the commitment value $(Comm_{Sxi} = g^{Sxi} \cdot h^r mod\, q)$.After computation, $Comm_{xList}$ is sent to the on-chain smart contract layer as part of the IoT data information.

Shamir Secret Sharing: In the IoT data sharing service process based on secure multi-party computation, off-chain IoT devices use Shamir's secret sharing Sharding algorithm to split data, and the on-chain smart contract layer uses Shamir's secret sharing Re algorithm to reconstruct the actual arithmetic circuit results from the off-chain computation result fragments.

Sharding: Based on Shamir's secret sharing (n, t) threshold data splitting, IoT device participants exchange their data ciphertext fragments through the trusted on-chain smart contract layer and decrypt to obtain data fragments. Here, n represents the number of participants. If there are n participants in this data sharing, they split their data into n fragments using the Sharding algorithm. The parameter t denotes the minimum number of fragments needed to reconstruct the data, and the value of t is set to n by default in this data sharing design. Assume the private value of a participant is S_x. The Sharding algorithm executes as follows:

Secret Distribution Algorithm Sharding: Let F be a finite field with sufficiently large characteristic, and $F[x]$ be the polynomial ring over F The participant randomly selects a polynomial $f(x)$ of degree $n-1$ from $F[x]$:

$$f(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1} \quad (1)$$

with $f(0) = S_x.For\ x_i \in \{1,2, \ldots, n\}$, compute $f(x_i)$to obtain the list of secret fragments $S_x = \{S_{x1}, S_{x2}, \ldots, S_{xn}\}$.

Re : After receiving the result fragments from the$n$ IoT device participants, the on-chain smart contract layer executes the Re algorithm to reconstruct the actual result fragments, returning the final arithmetic circuit computation result to the IoT data sharing service caller.

Secret Reconstruction Algorithm Re: $Given\ n$ distinct points $(x_i, f(x_i))$ on $f(x)$, where $i \in \{1, 2, \ldots, n\}$, reconstruct $f(x)$using the Lagrange interpolation formula

$$f(x) = \sum\nolimits_{i=1}^{n} f(x_i) \prod\nolimits_{j=1, j \neq i}^{n} \frac{x - x_j}{x_i - x_j} \quad (2)$$

where $f(0)$= result represents the desired arithmetic circuit execution result.

SM2 Asymmetric Encryption: To ensure the physical security of IoT data, this study stipulates that plaintext data must not leave the IoT device holder, and the plaintext value of data fragments can only be exposed to assigned IoT device participants. Asymmetric encryption further ensures the data security of fragments on the chain. Each of the n IoT device participants generates a public-private key pair (SK_i, PK_i). Taking device x as an example, after Sharding, x splits its data S_x into secret fragments $S_{xi} \ (i = 1,2, \ldots, n)$and encrypts each fragment$S_{xi}$with the public key PK_i of device$i$.

Protocol audit verification

In Fig. 2, during the implementation of secure multi-party computation, the on-chain smart contract layer acts as a transfer station for ciphertext fragment sending and receiving between the participants of the IoT device who execute the arithmetic gate circuit. On the chain, the behavior of the participants of the IoT device sending and receiving ciphertext fragments will be audited and verified as follows:

Coarse-grained audit verification: After each participant of the IoT device divides its IoT data into n secret shards, n Pedersen commitment values of these n secret shards are generated as the expenditure of the participant of the IoT device, and the corresponding Paillier public key of the participant of the IoT device is used to encrypt these n blind factors to obtain n blind factor ciphertext. The n expenditure commitment values of each participant of the IoT device and the ciphertext of the n blind factors corresponding to the generated commitment value (the blind factor is a random number) are sent to the blockchain along with the n ciphertext fragments, and the total expenditure commitment value is calculated on the chain. After the secondary fragment ciphertext allocation on the blockchain, the allocated ciphertext fragment, the corresponding secret fragment's expenditure commitment value, and the ciphertext of the blind factor generating the commitment value are sent to the corresponding participant of the IoT device. The participant of the IoT device uses the Paillier private key to decrypt the blind factor ciphertext and the ciphertext fragment to obtain the blind factor plaintext and the secret fragment corresponding to the blind factor plaintext, generate the promise value of the secret fragment as the income promise value and send it to the chain. On the chain, the comparison between the total income promise value and the total expenditure promise value is calculated. If the total expenditure equals the total income, the participant's sending and receiving operations are correct.

Fine-grained audit verification: The fine-grained audit verification supplements the coarse-grained audit verification. If the coarse-grained audit verification fails, the fine-grained audit verification locates the IoT device participants that fail to implement the protocol. After receiving the message that fails the coarse-grained verification, the participant generates the Pedersen commitment value and compares the expenditure commitment value according to the blind factor and ciphertext fragments received and decrypted. If they are not equal, the spender sends and receives private data ciphertext fragments according to the protocol to reduce the spender's trust and update the Bloom filter.

Figure 3 shows the behavior audit process of participants of IoT devices in the process of private data sharing service.

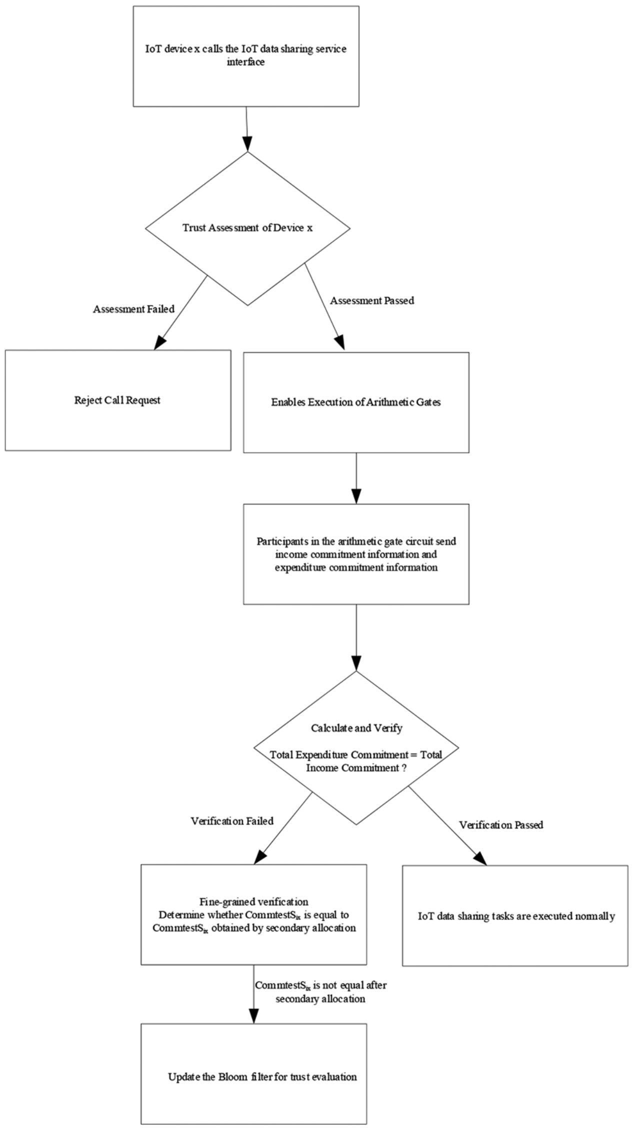
IoT device x calls the IoT data sharing service interface
Trust Assessment of Device x
Assessment Failed
Assessment Passed
Reject Call Request
Enables Execution of Arithmetic Gates
Participants in the arithmetic gate circuit send income commitment information and expenditure commitment information
Calculate and Verify
Total Expenditure Commitment = Total Income Commitment ?
Verification Failed
Verification Passed
Fine-grained verification Determine whether CommtestS$_{ix}$ is equal to CommtestS$_{ix}$ obtained by secondary allocation
IoT data sharing tasks are executed normally
CommtestS$_{ix}$ is not equal after secondary allocation
Update the Bloom filter for trust evaluation

Fig. 3 Audit Process for IoT Device Participant Behavior

Bloom filter design that can trace the trust state of the sensing layer

Storage structure design

In this research scheme, the on-chain smart contract layer is based on PKI architecture to control the access of participants of IoT devices. In addition, the on-chain smart contract layer can be embedded with other service interfaces for IoT devices to invoke due to flexibility considerations, and IoT devices can invoke other service interfaces on the chain with unique identity certificates except for IoT data sharing services. For the IoT data sharing service with high security requirements, the IoT device shall pass the verification of the Blum filter that can trace its trust status according to the device Id number in its certificate. After the verification is passed, the on-chain smart contract layer will process the IoT data sharing request of the IoT device of the task initiator. The invocation is shown in Fig. 4:

The Bloom filter [21] is a spatially efficient data structure that supports fast insertion and query and can be used to retrieve whether an element exists in a collection.

Specifically, the Bloom filter contains an array of bits, all of which are initially set to 0. When an element is to be added to the set, the element is hashed through multiple hash functions to obtain multiple hash values. Then set the position of the corresponding bit array to 1. When checking whether an element exists, the element is also hashed through the hash function to obtain multiple hash values. If the position of the corresponding bit array is 1, it indicates that the element may exist; If one of these

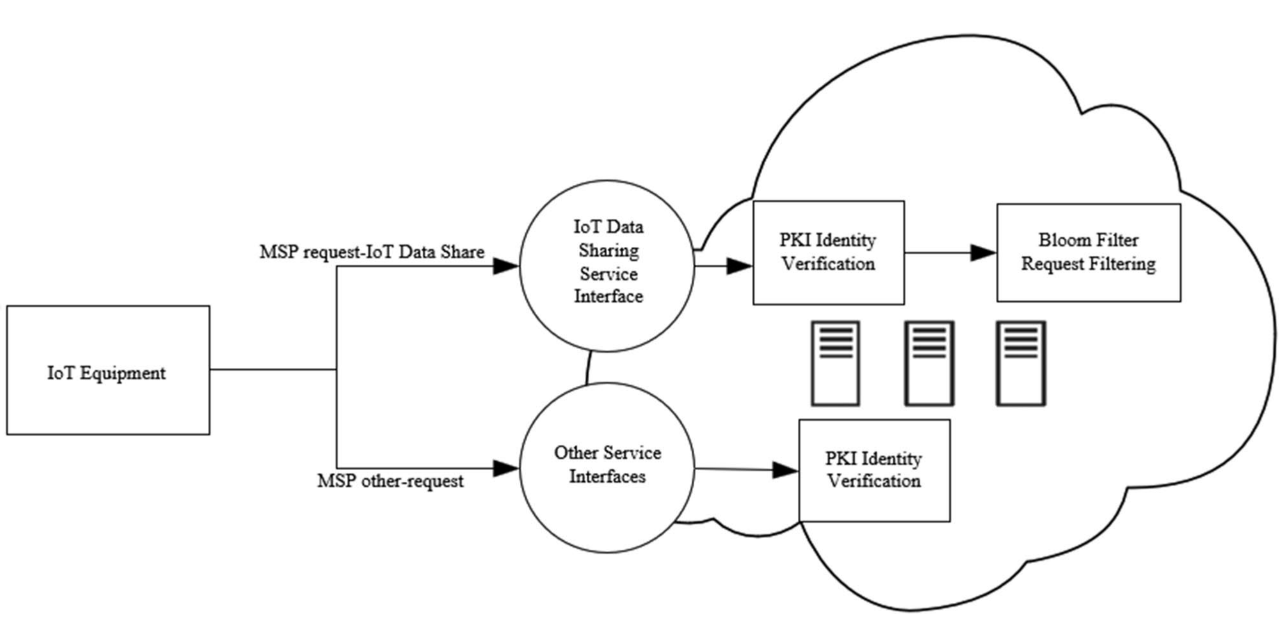
IoT Equipment
MSP request-IoT Data Share
MSP other-request
IoT Data Sharing Service Interface
Other Service Interfaces
PKI Identity Verification
Bloom Filter Request Filtering
PKI Identity Verification

Fig. 4 IoT device request verification

positions is 0, the element does not exist. The Bloom filter is suitable for situations where you can quickly determine the presence or absence of an element in large-scale data. The traditional Bloom filter only has a mechanical judgment of whether a certain element exists, and cannot adapt to the complex application scenarios of the real Internet of Things system. To this end, this paper constructs a Blum filter that can trace the trust status of IoT devices, dynamically updates the trust degree of IoT device participants according to the implementation of the IoT data sharing protocol of IoT devices, and synchronizes it to the on-chain smart contract layer of the global edge server through the consensus mechanism. The next time an IoT device with low trust initiates a request for private data sharing, The on-chain smart contract layer rejects the task request.

The Blum filter that can trace the trust status of the layer is composed of three layers of Blum filters. The Id of the IoT device goes through the first layer Blum filter to confirm whether the IoT device has registered the arithmetic gate circuit. After passing the first layer, it goes through the second layer Blum filter to confirm the trust status of the IoT device. If the IoT device can pass the first two layers of Blum filter screening, it indicates that it is a trusted IoT device, if it passes the first layer but does not pass the second layer of screening, it indicates that the IoT device is not trustworthy, and at the same time, the number of illegal access is recorded in the third layer of Blum filter.

The third layer Bloom filter essentially subscripts two bits of the original bit array to one bit of the third layer, so that the third layer represents three state values.

Table 1 Bloom filter state values definition for traceable perception layer trust states

3Layer 1,2Layer	**0 0**	**0 1**	**1 0**	**1 1**
0	Unregistered Device	Undefined	Suspicious Equipment	Trusted Device
1	Undefined	Undefined	Suspicious Equipment	Undefined
2	Undefined	Undefined	Malicious Device	Undefined

Table 1 below shows the status values of the three layers of Bloom filters on the chain that can be traced to the trust state of the sensing layer and their corresponding ones.

Non-registered device: The IoT device has not registered the arithmetic gate circuit at the on-chain smart contract layer, and the device has no need to call private data sharing services, and it can normally call other services at the on-chain smart contract layer with an identity certificate.

Trusted device: The IoT device is normally registered with the arithmetic gate and honestly executes the agreement whether it is the initiator or the participant of the private data sharing task.

Suspicious device: As a participant or initiator of a private data sharing task, a device that does not honestly execute in accordance with the agreement retains its right to call other services of the smart contract layer on the chain.

Malicious device: If a suspicious device illegally invokes the private data sharing interface twice or more, its PKI certificate is revoked and services are denied.

Figure 5 shows a storage structure of Bloom filter with the first two layers being 32-bit Bitmap and the third layer being 16-bit Bitmap, which is traceable and aware of device trust status:

Analysis of misjudgment rate

Because the Blum filter that can trace the trust state of the device at the sensing layer has three layers and the structure update mode of the third layer is different from that of the traditional Blum filter, the misjudgment rate needs to be re-analyzed and calculated.

Suppose n device ids are inserted into the first two layers as initial trusted IoT devices, and each device Id is input into k hash functions at each layer. Since the first layer is a traditional Bloom filter structure, the error rate of the first two layers is as follows:

$$f_{p1} = f_{p2} = \left(1 - e^{-kn/m}\right)^k \tag{3}$$

For the third layer Bloom filter, the quality function of the probability distribution of its subscript l is defined as:

$$b\left(l, kn, \frac{1}{m}\right) = \binom{kn}{l}\left(\frac{1}{m}\right)^l\left(1 - \frac{1}{m}\right)^{kn-l} \tag{4}$$

The misjudgment rate of the hash mapped device Id with the corresponding value of l is expressed as:

$$P_{f_p}(\theta, k, n, m) = \left(1 - \sum\nolimits_{l<\kappa} b\left(l, kn, \frac{1}{m}\right)\right)^k \tag{5}$$

Its θ represents the state value of the third layer.

The binomial distribution is replaced by the Poisson distribution and the Poisson distribution cumulative mass function is substituted

$$f_p(\theta = 1) = \left(1 - \frac{\Gamma(1, \kappa)}{\Gamma(1,0)}\right)^k \tag{6}$$

If the three state values of the third layer Bloom filter are 0,1,2, if an element exists in the set of Bloom filters, it will never be misjudged as not existing. So you only need to plug in two state values, 1,2.

$$f_p(\theta = 1) = \left(1 - \frac{\Gamma(1, \kappa)}{\Gamma(1,0)}\right)^k \tag{7}$$

$$f_p(\theta = 2) = \left(1 - \frac{\Gamma(2, \kappa)}{\Gamma(2,0)}\right)^k \tag{8}$$

Therefore, the misjudgment rate of the third layer Bloom filter is:

$$f_{p3} = \left(1 - \frac{\Gamma(1, \kappa)}{\Gamma(1,0)}\right)^k + \left(1 - \frac{\Gamma(2, \kappa)}{\Gamma(2,0)}\right)^k \tag{9}$$

Since each layer is related to the access control rights of the IoT device, the overall error rate is:

$$f_p = \max(f_{p1}, f_{p2}, f_{p3}) \tag{10}$$

Table 2 below shows the overall misjudgment rate of Blum filter in the traceable sensing layer device trust state:

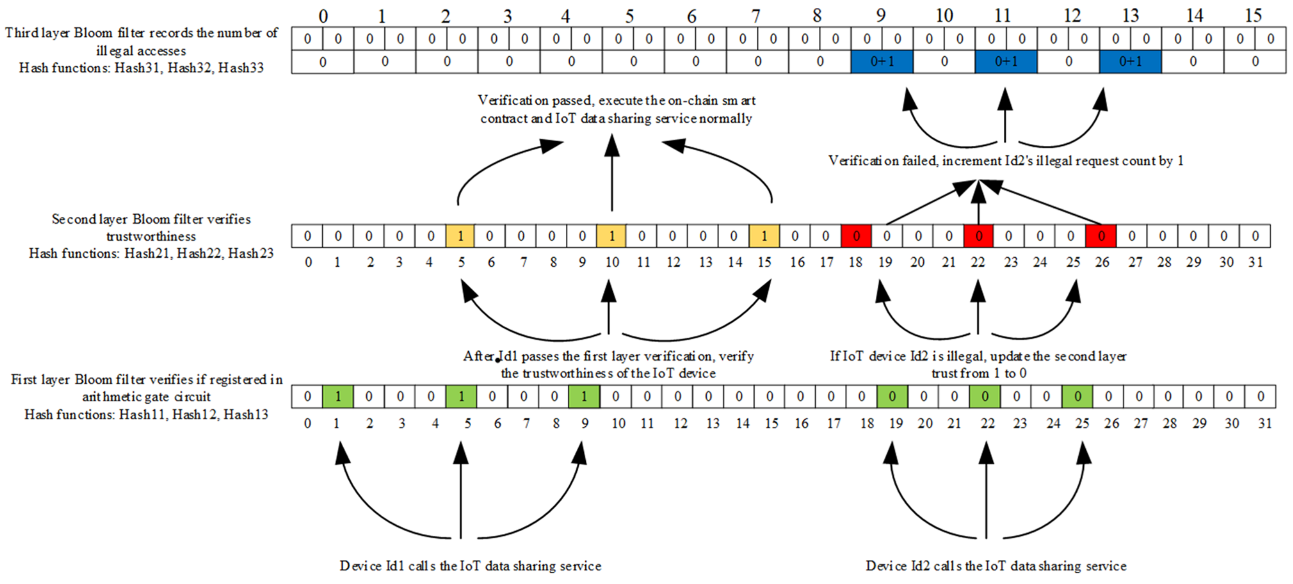
Third layer Bloom filter records the number of illegal accesses
Hash functions: Hash31, Hash32, Hash33
Verification passed, execute the on-chain smart contract and IoT data sharing service normally
Verification failed, increment Id2's illegal request count by 1
Second layer Bloom filter verifies trustworthiness
Hash functions: Hash21, Hash22, Hash23
After Id1 passes the first layer verification, verify the trustworthiness of the IoT device
If IoT device Id2 is illegal, update the second layer trust from 1 to 0
First layer Bloom filter verifies if registered in arithmetic gate circuit
Hash functions: Hash11, Hash12, Hash13
Device Id1 calls the IoT data sharing service
Device Id2 calls the IoT data sharing service

Fig. 5 Storage structure of the traceable trust state bloom filter for perception layer devices

Table 2 Analysis of the overall misjudgment rate

n/m	Misjudgment rate
0.1	3.563E-8
0.2	1.933E-6
0.3	1.869E-5
0.4	8.926E-5
0.5	2.897E-4

As can be seen from the table, when the occupancy rate n/m is low, the misjudgment rate is negligible. Therefore, selecting the appropriate size of Bloom filter in the actual application scenario can effectively and quickly manage the access control requests of IoT devices off the chain.

Program analysis and evaluation

Scheme analysis

Correctness Analysis: The IoT data sharing method based on secure multiparty computation is conducted in the form of data transactions. Among the N data transaction parties involved, during the data collection phase of IoT devices, each party shares its non-IoT data secret fragments with other IoT devices. The IoT device x of the sharing party generates the secret fragment S_{xi} $(\text{where } i \in \{1, 2, \ldots, N\})$and its Pedersen commitment $Comm_{xPay}\left(S_{xi}, r_{xi}\right) = g^{S_{xi}} h^{r_{xi}} \, mod \, q$ (where r_{xi} is a random number as the blinding factor for the Pedersen commitment, and (g, h, q) are the generators of the multiplicative group G of order q, which is a large prime number, with $G\ = ?g\ ? = ?h\ ?$ being publicly known).

During the secondary allocation phase, the IoT device x as the recipient generates an income commitment $Comm_{xAck}\left(S_{ix}, r_{xi}\right) = g^{S_{ix}} h^{r_{xi}} \, mod \, q$. The on-chain smart contract layer, acting as a trusted third-party auditor, verifies whether the total income and expenditure of the N data transaction participants are balanced and whether the commitment values satisfy:

$$Comm_{1Pay} \cdot \ldots \cdot Comm_{NPay} = Comm_{1Ack} \cdot \ldots \cdot Comm_{NAck}$$

If this holds true, the input and output values of the transaction are equal, and no specific transaction data is leaked to the on-chain smart contract layer, completing the audit. If not, the recipient of the data transaction restores the expenditure commitment value of the secret fragment using the blind factor and the secret fragment received during the expenditure phase. If the restored expenditure commitment value does not match the received expenditure commitment value, it confirms that the sender of the secret fragment did not follow the protocol as required, and the on-chain smart contract layer will adjust the sender's trust value. Therefore, this protocol ensures the correctness of IoT data sharing.

Simulation implementation and experimental analysis

This section presents the simulation implementation and introduces the specific process for the on-chain smart contract layer and the off-chain IoT device perception layer. For the multiple nodes involved in the IoT perception layer, Docker container technology is used to set up a single-machine multi-node microservice system for simulation. For the smart contract layer, the Hyperledger Fabric consortium blockchain platform is used to build a consortium network. The IoT data sharing protocol and the Bloom filter for the traceable perception layer trust states are integrated into Hyperledger Fabric as chaincode. Hyperledger Fabric uses Docker containers to run various types of nodes (Peer nodes, Orderer nodes, and CA nodes). Peer nodes typically run one or more chaincodes (smart contracts), and containerization ensures that each chaincode executes in its own isolated environment, thereby avoiding potential conflicts and security issues. Therefore, this section uses two computers as host machines to build and deploy the microservice simulation system.

On-chain Smart Contract Layer: The on-chain smart contract layer uses a single-machine multi-node setup to build a Hyperledger Fabric permissioned blockchain network configured with PBFT consensus. Docker is used to isolate the running environments of the nodes. One host is used as the edge server to build the consortium network, create a unique channel within the network, and set up different organizations to join the channel for simulation. Each organization is assigned a unique digital identity—an organizational key and root certificate, a TLS communication key, and a TLS root certificate. The organizational key and TLS key can issue identity certificates and TLS certificates for the nodes or users under that organization. The experiment is set up with one Peer node per organization, and certificates are issued for each. Each organization has its own certificate management module and Orderer node to manage the validity of its issued certificates and perform state synchronization during block creation. Chaincode for the IoT data sharing service is written, installed, and instantiated.

Off-chain IoT Device Perception Layer: Another computer is used to write a Vue-SpringBoot backend management program that generates different numbers of user IDs as IoT device IDs and registers corresponding MSP certificates with the on-chain smart contract layer. The backend management program uses the Fabric SDK for JAVA to manage the invocation of these users' IoT data sharing services. The on-chain smart contract layer's three-layer Bloom filter initializes these user IDs as trusted devices.

As shown in Table 3, the hardware and software configuration table, and Fig. 6, the simulation system architecture diagram.

Table 3 Hardware and software configuration

Hardware/Software environment	Configuration model/Version
Operating System	Ubuntu20.04
Processor	Intel Core i7
Memory	32GB
Hyperledger Fabric	V1.4.12
Hyperledger Fabric CA	V1.4.4
Fabric SDK for JAVA	V2.2.1
Paillier Key Length	512bit

Peer nodes are the core components of the Hyperledger Fabric network. They are responsible for maintaining the ledger state, executing smart contracts, validating transactions, and providing query and event notification functions. Orderer nodes are responsible for packaging transactions signed by Peer nodes into blocks.

This section defines the following standards to evaluate the performance of the solution:

Response Time: The time delay from the initiation of the data sharing request by the IoT device to the receipt of the response result. Based on the transaction process of Hyperledger Fabric, the response time consists of communication time and CPU time.

Execution Time: The time spent by IoT device participants in executing secure multiparty computation.

As shown in Fig. 7, the system's response time for completing all task requests when simultaneously initiating different numbers of secure multiparty computation tasks is tested. This test involves different numbers of IoT device users (n=3, 5, 10) performing (n, n) threshold secret sharing addition operations.

From Fig. 7, it can be seen that as the number of secure multi-party computation task requests increases, the response time for completing all secure multi-party computation tasks does not significantly increase when the number of participants remains the same. This is because the IoT data sharing protocol requires Fabric to generate multiple blocks. Each block carries task information and event information to advance the secure multi-party computation tasks. The amount of task information that can be contained in a single block in Fabric is controlled by the parameters *AbsoluteMaxBytes* and *PreferredMaxBytes*. By setting these parameters, each block can contain a large amount of task information. Therefore, when multiple secure multi-party computation requests are received simultaneously, Fabric's batching mechanism allows multiple task requests to be completed simultaneously during one round of the IoT data sharing protocol. However, the more participants there are, the longer it takes to reach consensus on task status among the participants, resulting in longer response times. In this experiment, the transaction frequency of the task requests has not yet reached *MaxMessageCount.* Therefore, the factor that most significantly affects the task response time is still the block generation interval *BatchTimeout*, leading to longer response times.

As shown in Fig. 8, based on the experiment in Fig. 7, *MaxMessageCount* is reset to 50 to test the average response time for completing a single task request when different numbers of IoT device users (n=3, 5, 10) simultaneously initiate 50 data sharing requests. From Fig. 8, it can be seen that when the number of task requests reaches 50, the transaction frequency for this number

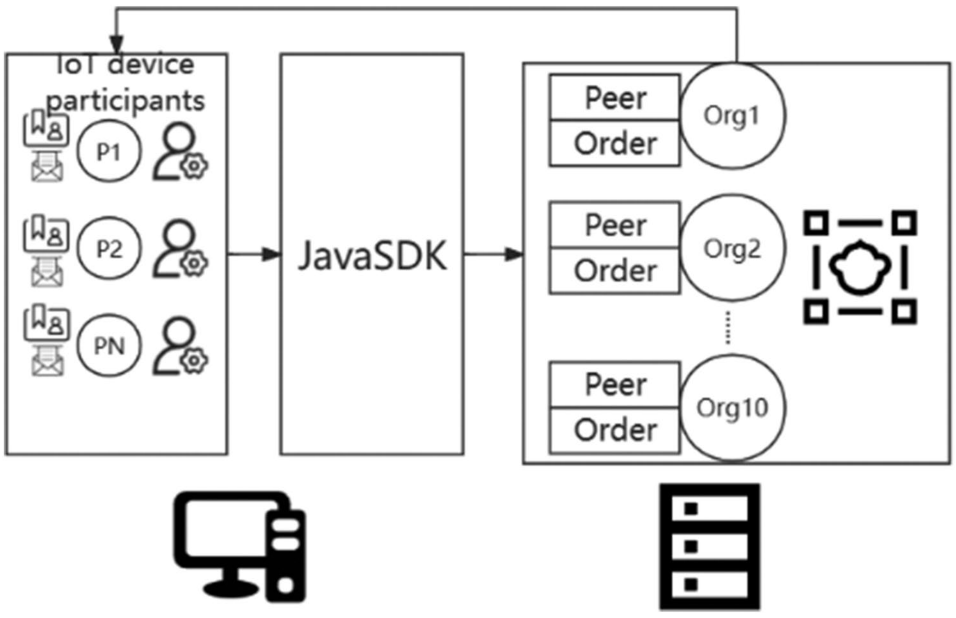

Fig. 6 Experimental architecture diagram

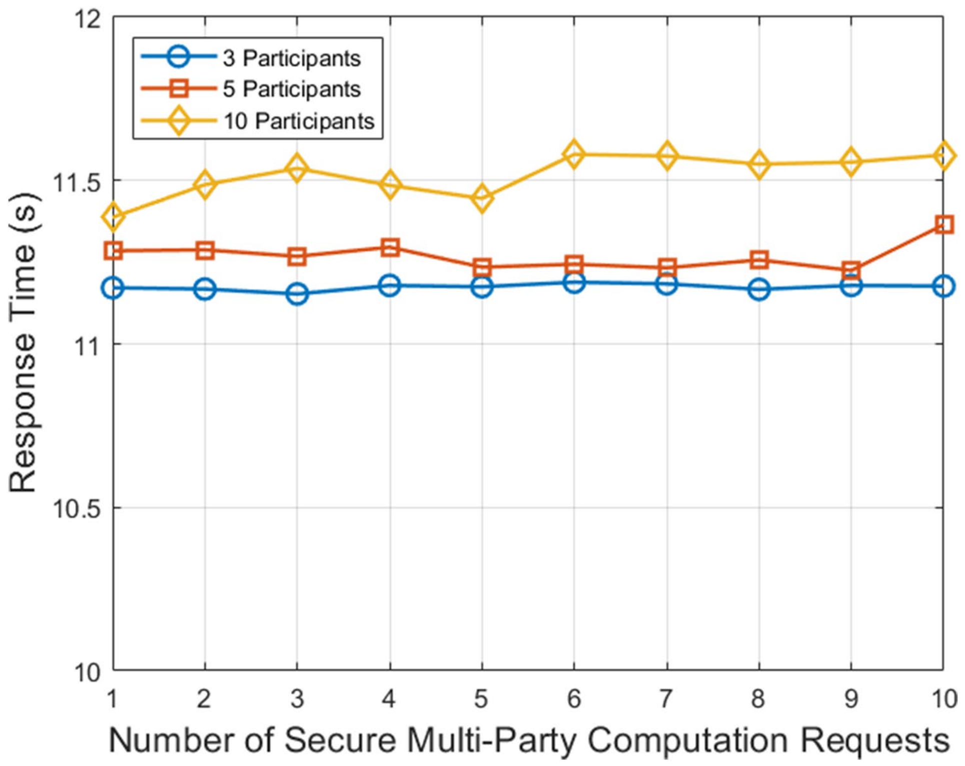

Fig. 7 Response time of secure multi-party computing task (Addition operation) under different task request quantities

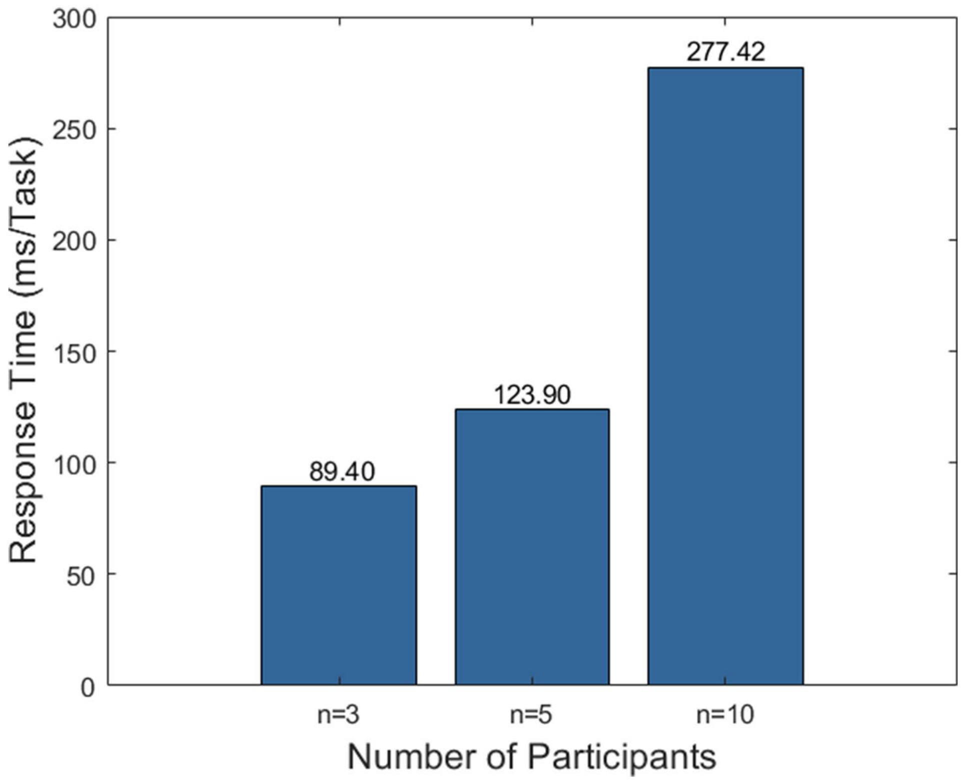

Fig. 8 Average response time of each secure multi-party computing task (Addition operation) with different numbers of participants

of tasks exceeds *MaxMessageCount*: 50. Therefore, the block generation time is primarily affected by the *MaxMessageCount* parameter. The block generation time is significantly shorter than under the conditions of the experiment in Fig. 7. In this case, the more participants there are, the higher the transaction frequency, and the longer the response time. The number of IoT device users has the most significant impact on the response time.

For multiplication, Shamir's secret sharing has limited homomorphism. This is because secret sharing is achieved by selecting points on a polynomial of fixed degree, and multiplication of linear secret shares generated by polynomials will result in a higher degree polynomial. Different schemes have different strategies to address this issue. As shown in Fig. 9, the execution time for different numbers of IoT device users $(n = 3, ?, 10)$ cooperating to complete a multiplication gate operation (i.e., n computing nodes cooperating to execute the multiplication operation between two (n, n) threshold IoT data secret shares) is presented under different schemes.

The blockchain-based secure multi-party computation scheme proposed by PS-MPC [23] uses a log-depth formula with constant-sized MPC gates to simulate n-party secure protocols, reducing the communication complexity of multiplication from quadratic to linear. This scheme is an off-chain computation scheme. The secure multi-party computation scheme proposed by Ghadamyari [21] adopts an on-chain computation approach and originally used Paillier homomorphic encryption to protect secret shares. However, Paillier encryption supports only scalar multiplication homomorphism for ciphertext shares, which limits the functionality of the scheme to some extent. To address this issue, this paper opts to replace Paillier with the BFV fully homomorphic encryption (FHE) algorithm. BFV is a more powerful fully homomorphic encryption algorithm that supports homomorphic multiplication, meaning that data can still be multiplied in its encrypted state without exposing the original information. Therefore, by introducing the BFV algorithm, this on-chain computation scheme is upgraded to perform more complex computations while protecting data privacy, with a key length set to 512 bits. BSCEN uses proactively generated Beaver triple shards to achieve homomorphic multiplication of secret shares, and BSCEN's scheme is an off-chain computation scheme.

As can be seen from Fig. 9, although PS-MPC [23] reduces the communication complexity of multiplication from quadratic to linear and, like the BSCEN scheme in this paper [20], belongs to off-chain computation, the trade-off is an increase in parallel computation complexity, resulting in lower execution efficiency compared to this paper's scheme. Ghadamyari-FHE [21] is an on-chain computation scheme, which involves on-chain node consensus, thus making its communication efficiency lower than that of off-chain computation. Furthermore, while

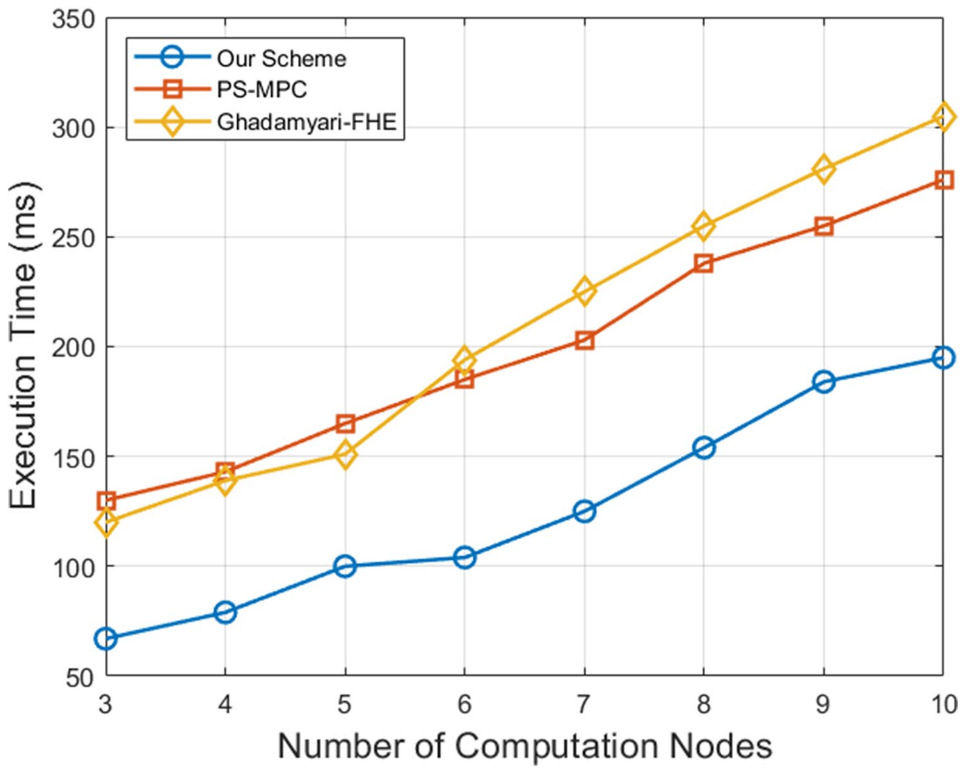

Fig. 9 Execution time of different numbers of IoT user nodes cooperating to complete a multiplication gate operation

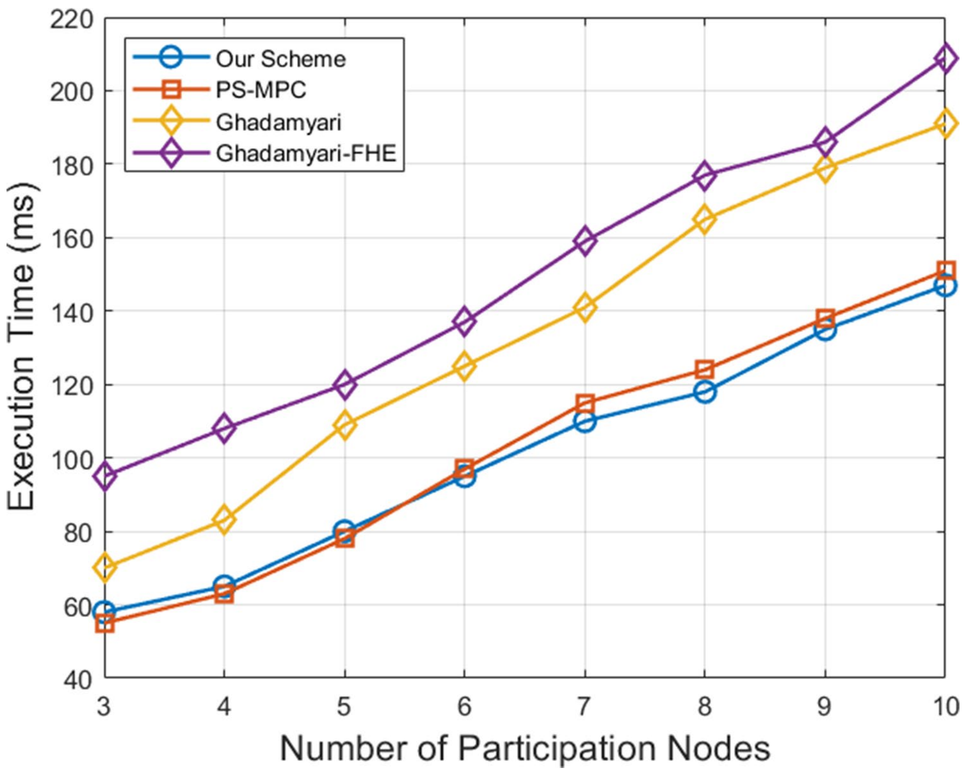

Fig. 10 The Execution time of different numbers of iot user nodes cooperating to complete an addition gate operation

Ghadamyari-FHE [21] uses homomorphic ciphertext multiplication for multiplication operations, this paper's scheme directly uses plaintext secret shares to execute multiplication operations based on Beaver triples. Consequently, the computational complexity of Ghadamyari-FHE [21] is significantly higher than that of this paper's scheme. Therefore, the scheme proposed in this paper has a certain computational efficiency advantage in performing multiplication operations.

As shown in Fig. 10, the execution time for different numbers of IoT device users cooperating to complete an addition gate operation is presented. For addition operations, since Shamir's secret sharing inherently possesses homomorphism [22], when executing an addition gate operation, the scheme proposed in this paper, like PS-MPC [23] and Ghadamyari [21], only requires IoT nodes to complete their respective computation tasks in parallel. However, to test the execution time, it is still necessary to interact once to restore the common addition gate operation result from the secret shares among the execution parties. We plan to further explore how to optimize the data sharing and model training processes through advanced mechanisms such as hierarchical Stackelberg games, to achieve higher efficiency and fairness [21].

From the perspective of computational efficiency, both the IoT nodes in this paper's scheme and PS-MPC [23] perform secure multi-party addition operations on plaintext secret shares off-chain, while Ghadamyari [21] performs secure multi-party addition operations on ciphertext secret shares based on Paillier homomorphic encryption, which has higher computational complexity. Therefore, in terms of computational efficiency, this paper's method and PS-MPC [23] are more efficient than Ghadamyari [21]. For Ghadamyari-FHE, which replaces Paillier homomorphic encryption with the BFV fully homomorphic encryption algorithm, the computational complexity of fully homomorphic encryption is the highest, making its computational efficiency the lowest under this method.

In summary, considering the overall computational efficiency of IoT nodes, the method proposed in this paper has higher computational efficiency compared to other methods.

Conclusion

This paper proposes a feasible IoT data sharing scheme based on the application scenario of IoT combined with edge computing, utilizing blockchain and secure multi-party computation technology. The scheme constructs an off-chain collaborative secure multi-party computation protocol for IoT data sharing, allowing trusted IoT devices to exchange and share IoT data securely and controllably in a multi-party computing environment. Through comparative analysis and performance evaluation, this scheme can effectively integrate IoT data into off-chain secure data sharing, with relatively high computational efficiency in the application scenario of IoT systems, and has practical value. In the future, we plan to further explore how to optimize the IoT data-sharing model through advanced mechanisms such as hierarchical Stackelberg games, to achieve higher efficiency and security [24].

Author contributions
Ma is responsible for coordinating the title and research plan, Duan and Zhang is responsible for implementing the plan and writing the paper, Li , Fu and Ma are responsible for revising the paper.

Funding
Declaration: This work was supported by The National Natural Science Foundation of China (No.62272007), The Natural Science Foundation of Beijing (No. 4234083).

Data availability
No datasets were generated or analysed during the current study.

Declarations

Competing interests
The authors declare no competing interests.

Received: 24 January 2024 / Accepted: 2 September 2024
Published online: 13 September 2024

References
1. Khan WZ, Ahmed E, Hakak S et al (2019) Edge computing: a survey [J]. Future Generation Comput Syst 97:219–235
2. Ansari MS, Alsamhi SH, Qiao Y et al (2020) Security of distributed intelligence in edge computing: Threats and countermeasures [J]. The Cloud-to-Thing Continuum: Opportunities and Challenges in Cloud, Fog and Edge Computing, : 95–122
3. Conti M, Dragoni N, Lesyk V (2016) A survey of man in the middle attacks [J]. IEEE Commun Surv Tutorials 18(3):2027–2051
4. Pedersen TP (1991) Non-interactive and information-theoretic secure verifiable secret sharing [C]//Annual international cryptology conference. Berlin, Heidelberg: Springer Berlin Heidelberg, : 129–140
5. Huigui R, Jinxia M, Bingguo C, Guang S, Fei L Key distribution and recovery algorithm based on Shamir Secret sharing [J]. J Commun 2015, 36(03):64–73
6. Liu Y, Yu W, Ai Z, Xu G, Zhao L, Tian Z (2023) A blockchain-empowered Federated Learning in Healthcare-based Cyber Physical systems. IEEE Trans Netw Sci Eng 10(5):2685–2696. https://doi.org/10.1109/TNSE.2022.3168025
7. Liu Y, Zhang Y, Su S et al (2024) BlockSC: a Blockchain empowered spatial Crowdsourcing Service in Metaverse while preserving user location privacy [J]. IEEE J Sel Areas Commun 42(4):880–891. https://doi.org/10.1109/JSAC.2023.3345416
8. Sikeridis D, Bidram A, Devetsikiotis M et al (2020) A blockchain-based mechanism for secure data exchange in smart grid protection systems [C]//2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC). IEEE, : 1–6
9. Sengupta J, Ruj S, Bit SD (2020) A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT [J]. J Netw Comput Appl 149:102481
10. Ouaddah A, Abou Elkalam A, Ait Ouahman A (2016) FairAccess: a new Blockchain-based access control framework for the internet of things [J]. Secur Communication Networks 9(18):5943–5964
11. Liu CH, Lin Q, Wen S (2018) Blockchain-enabled data collection and sharing for industrial IoT with deep reinforcement learning [J]. IEEE Trans Industr Inf 15(6):3516–3526
12. Liang X, An N, Li D A Blockchain and ABAC Based Data Access Control Scheme in Smart Grid [C]//2022 International Conference on Blockchain Technology and, Security I et al (2022) (ICBCTIS). IEEE, : 52–55
13. Yang Y, Wu J, Long C et al (2022) Blockchain-enabled multiparty computation for privacy preserving and public audit in Industrial IoT [J]. IEEE Trans Industr Inf 18(12):9259–9267
14. Zelbst PJ, Green KW, Sower VE et al (2020) The impact of RFID, IIoT, and Blockchain technologies on supply chain transparency [J]. J Manuf Technol Manage 31(3):441–457
15. Alladi T, Chamola V, Parizi RM et al (2019) Blockchain applications for industry 4.0 and industrial IoT: a review [J]. Ieee Access 7:176935–176951
16. Yu K, Tan L, Aloqaily M et al (2021) Blockchain-enhanced data sharing with traceable and direct revocation in IIoT [J]. IEEE Trans Industr Inf 17(11):7669–7678
17. Wu Hongyue C, Zhiwei SHI, Bowen DENG, Shuiguang C, Shizhan XUE, Xiao FENG, Zhiyong (2019) A decentralized service request distribution method for edge computing environment [J]. Chin J Comput 46(05):987–1002
18. Cao K, Liu Y, Meng G et al (2020) An overview on edge computing research [J]. IEEE Access 8:85714–85728
19. Chen B, Wan J, Celesti A et al (2018) Edge computing in IoT-based manufacturing [J]. IEEE Commun Mag 56(9):103–109
20. Permission L, Yanbiao X Gaogang Efficient data name search method based on mixed Count Bloom Filter [J]. J Comput Res Dev 2023, 60(05):1136–1150
21. Ghadamyari M, Samet S (2019) Privacy-Preserving Statistical Analysis of Health Data Using Paillier Ho momorphic Encryption and Permissioned Blockchain [C].In:2019 IEEE International Conference on Big Data(Big Data),5474–5479
22. Jianhua H, Huiya J,Zhongcheng L.Constructing fair secure multi-party computation based on blockchain [J]. Appl Res Comput 2020, 37:225–244
23. Zyskind G, Nathan O,Pentland A Enigma:Decentralized computation platform with guaranteed privacy [J].arXiv preprint arXiv:1506.03471,2015
24. Kang H, Ji S Hierarchical Stackelberg Game Swarm learning incentive method for Wireless Edge Network [J/OL]