International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 17 (2025) pp. 1-17

https://doi.org/10.70917/ijcisim-2025-0316
Article

Blockchain-based IoT device identity authentication and data traceability

Weibing Li *, Shenggang Wu and Wei Han

Academic Affairs Office, Beijing Polytechnic University, Beijing, 100176, China;
liweibing0756@163.com (W.L.); wushenggang@bpu.edu.cn (S.W.); hanwei@bpu.edu.cn (W.H.)

Abstract: With the advent of the Internet of Things (IoT) era, IoT data has become increasingly important. Due to risks such as data forgery in network data, there is an even greater need for identity authentication and data traceability. In response to this, a research proposal for IoT device identity authentication and data traceability based on blockchain technology has been designed. First, by combining IoT devices with blockchain technology, the technical approach for IoT device identity authentication and data traceability is determined. Then, using development tools, the proposed technical approach is designed and implemented. Finally, the proposed research solution is subjected to empirical analysis from multiple dimensions. Under the proposed traceability method, as data volume increases, the time overhead approaches zero, with a traceability tracking time range of 280–300 ms and minimal variability, indicating that the proposed research solution demonstrates excellent practical effectiveness.

Keywords: IoT devices; blockchain; identity authentication; data traceability

1. Introduction

With the rapid development of the Internet of Things (IoT), identity verification and data traceability for IoT devices have become important research topics [1-2]. Literature [3] reviews machine learning techniques used to identify IoT devices and methods for detecting tampered or counterfeit devices, categorizing IoT device identification and detection into device-specific pattern recognition, anomaly detection, and other methods, and analyzes the key technologies for identifying IoT devices and detecting tampered or counterfeit devices. Traditional centralized authentication methods face challenges in data security, trustworthiness, and real-time performance [4-5]. Blockchain-based authentication and traceability technologies can overcome these challenges, providing IoT devices with more secure and trustworthy authentication and data traceability [6-7]. Reference [8] proposes a trusted product traceability system based on Hyperledger Fabric and electronic product code information services, which ensures product traceability and can also authenticate and authorize IoT devices used for data collection.

Blockchain is a distributed ledger technology that stores data in a chain-like structure across multiple nodes, ensuring the immutability and decentralized nature of the data [9-10]. Literature [11] traces the evolution of blockchain systems, aiming to highlight the importance of decentralized applications (dApps) and the future value of blockchain, while conducting research on advanced dApps to explore the direction of blockchain development. Blockchain-based identity verification and traceability technology leverages the advantages of blockchain to store the identity information and data of IoT devices on the blockchain, ensuring the authenticity of identities and the reliability of data [12-14]. Literature [15] proposes an IoT blockchain gateway (BCoT) to record authentication transactions within the blockchain network and introduces a new device identification model suitable for blockchain-based identity authentication, validating the feasibility of the aforementioned framework.

First, blockchain-based identity verification technology enables unique device identification [16]. Each IoT device has a unique identity on the blockchain, which can be used to verify the device's identity [17-18]. Reference [19] proposes an identity verification system and method based on IoT and blockchain technology, which detects anomalies by comparing the traffic characteristics and behavioral information of nodes with the node activities observed on the blockchain.

Secondly, blockchain-based traceability technology has enabled the traceability of IoT device data [20]. In traditional IoT applications, there are certain challenges in tracing device data [21]. However, blockchain-based traceability technology achieves data immutability and traceability by storing device data in the blockchain in the form of transactions [22-23]. Literature [24] explores the design philosophy of developing an IoT data management platform, as well as the implementation of data management and linear traceability on this platform using blockchain and smart contracts, verifying the platform's effectiveness in ensuring data privacy and integrity. Literature [25] proposed a blockchain-based storage system named “Sapphire” for data analysis in IoT, and achieved interaction between IoT devices and the blockchain through the adoption of an object-storage-based smart contract method as the transaction protocol within “Sapphire.”

Similarly, each device's data has a corresponding identity identifier on the blockchain, which can be traced back to the source of the data [26-27]. This traceability technology has broad application prospects in fields such as food safety and pharmaceutical traceability [28]. Reference [29] introduces AgriBlockIoT, a fully decentralized, blockchain-based agricultural food supply chain management traceability solution, and evaluates its main advantages and disadvantages. Additionally, blockchain-based identity verification and traceability technology can enhance the efficiency of information sharing and interaction among IoT devices [30-31]. Traditional centralized identity verification and traceability methods require validation and authorization through a central authority, leading to bottlenecks and delays in information exchange [32-33]. In contrast, blockchain-based technology enables direct interaction and information sharing between devices, improving data real-time performance and efficiency [34-35]. Reference [36] proposes a blockchain-based trusted interaction architecture for smart manufacturing devices at the device layer of industrial IoT, and simulates experiments to validate the feasibility of this architecture. IoT devices can perform real-time identity verification and traceability queries through mechanisms such as smart contracts, significantly enhancing the efficiency and security of IoT applications [37-39]. Reference [40] proposes a decentralized access control framework based on smart contracts and analyzes the performance of access strategies based on smart contracts using blockchain transaction gas consumption data. It also examines the system's security, scalability, and other performance aspects.

This paper combines relevant research materials to design an identity authentication scheme for IoT devices based on blockchain technology from two aspects: the physical layer and the blockchain layer. The overall identity authentication process can be divided into three stages: the initialization stage, the registration stage, and the identity authentication verification stage. Building upon the existing framework, this paper expands the identity authentication functionality for IoT devices and designs a blockchain-based data traceability method for IoT devices. The overall framework is divided into three layers: the traceability information collection layer, the traceability information storage layer, and the traceability information display layer. Using development tools, the research theoretical framework and process are designed and implemented, and the research solution is analyzed.

2. Exploring Identity Authentication and Data Traceability from a Blockchain Perspective

2.1. Blockchain-based identity authentication for IoT devices

Identity authentication refers to the process of confirming a user's true identity through certain means. This section studies the issue of device authentication in the Internet of Things (IoT). Given that most IoT devices have limited computing resources, the identity authentication process consumes a large amount of resources, and the authentication process is overly dependent on third-party trusted centers, this section introduces blockchain to establish trust among IoT devices and achieve secure anonymous authentication between them.

2.1.1. Physical Layer

The physical layer plays a crucial role in the IoT architecture. It primarily consists of various IoT devices and KGCs. These IoT devices form the infrastructure of the entire network, possessing the capabilities to sense, process, and execute tasks. They can collect environmental information in real time, perform predefined tasks, and communicate with other devices or systems. However, these devices are often constrained by hardware resources, with relatively limited storage and computing capabilities, making them ill-equipped to handle complex data encryption and authentication tasks. Therefore, when IoT devices boot up or join the network, they need to request their own private key from the KGC. This private key will serve as the device's identity within the network, used for authentication and

encryption/decryption operations during subsequent communications. As the trust center of the entire network, the KGC is responsible for generating and managing device private keys. When an IoT device requests a private key from the KGC, the KGC verifies the device's identity and generates a unique private key for it, which is then securely transmitted to the device.

2.1.2. Blockchain Layer

The blockchain layer serves as a core component of the IoT cross-domain authentication system, primarily composed of blockchain agent nodes (BANs) and the underlying blockchain technology [41]. These elements collaborate to provide IoT devices with a secure, reliable, and efficient cross-domain authentication environment. Each KGC in a management domain requires a BAN to maintain the blockchain's global distributed ledger, with each BAN corresponding to a single KGC. The presence of BANs enables KGCs to focus on their core functions of key management and identity authentication. BAN indexes domain-internal information and uploads it to the blockchain, with the index pointing to the specific storage location of the information, significantly improving blockchain write and read speeds and ensuring authentication efficiency. The blockchain's global distributed ledger is composed of blocks encapsulating transactions, with each block containing an index of domain-specific information. During the authentication process, devices across different management domains can establish trust relationships and complete authentication with the assistance of the blockchain. Due to the unique properties of the blockchain, information uploaded to it cannot be arbitrarily tampered with, ensuring its authenticity.

2.1.3. Initialization Phase

Assuming that during the deployment of various entities, the system public parameters used in the subsequent text have been securely placed in advance in various entities participating in identity authentication. System parameter $params = \{BP, g, P_1, P_2, P_{rub}, H_1, H_2, hid, eid\}$ is pre-configured within entities in each domain to facilitate identity-based signature and verification. This includes the generators $P_i, P_2, \varphi(P_2) = P_1$ of the bilinear groups $BP = (G_1, G_2, G_T, e, N)$, $N > 2^{\lambda}$, G_1, and G_2; two cryptographic hash functions $H_1 : \{0,1\}^* \to Z_N^*$, $H_2 : \{0,1\}^n \times G_T \times N \to Z_N^*$, and n for signature data length; and the bilinear pairing identifier $eid : G_1 \times G_2 \to G_r$ of e. Within each domain, the KGC selects a random number $\beta \in [1, N-1]$ as the signature master private key, calculates $P_{Pub} = \beta P_2$ as the KGC signature master public key, and then calculates $g = e(P_1, P_{Pub})$. Next, the signature private key generation function is selected, with hid as the marker, and finally the parameters are made public.

2.1.4. Registration stage

Each management domain contains terminal devices participating in authentication, intermediate systems, and various subnets. Since each management domain is often considered secure, i.e., the entities within each management domain are mutually trusted, this solution assumes that IoT devices, KGC, and BAN within the same management domain are mutually trusted.

In the IBC system, each device's identity identifier is directly used as a public key. For security and privacy reasons, this solution does not directly use the device's real identity identifier as a public key. The device's unique identity identifier ID_i is hidden. Instead, a pseudo-anonymous temporary identity identifier PID_i distributed by the KGC is used as the public key. Additionally, the KGC generates corresponding private keys for IoT devices and uploads this information to the blockchain ledger to ensure immutability and traceability.

d_i Select a random number R_i, calculate $S_1 = Hash(ID_i \,||\, R_i)$ and ID_{d_i} as the unique identifier for d_i, where ID_{d_i} primarily contains the real identity information of d_i, such as IP address, serial number, expiration time, etc. d_i Send the registration request message and (S_1, R_i) to KGC. Upon receiving the message, KGC searches for a value that satisfies the condition $S' = S_1$ for $ID_{di}^{'}$, where:

$$S' = Hash(ID_{d_i}^{'} \,||\, R_i) \tag{1}$$

If found, it means that d_i has a legal identity, KGC can generate a private key and

pseudo-anonymity for d_i. Then KGC randomly selects $k \in [1, N-1]$, calculates $PID_{i,1} = kP_1$, and then KGC performs the following calculation:

$$PID_{i,2} = ID_{d_i} \oplus H_1(\beta \cdot PID_{i,1} + P_{Pub}) \quad (2)$$

$$PID_{d_i} = (PID_{i,1}, PID_{i,2}) \quad (3)$$

$$t_1 = H_1(PID_{d_i} \| hid, N) + \beta \quad (4)$$

$$t_2 = \beta \cdot t_1^{-1} \quad (5)$$

$$sk_{d_i} = [t_2]P_2 \quad (6)$$

KGC sends a request to BAN to update the information of d_i and update the index information written to the blockchain ledger. After the write is successful, BAN sends a write success response to KGC, and then KGC sends (PID_{d_i}, sk_{d_i}) to d_i through a secure channel. d_i obtains (PID_{d_i}, sk_{d_i}), which is pseudo-anonymity and the signature private key.

2.1.5. Identity Authentication Phase

(1) Authentication mechanism

The IBS digital signature verification algorithm was used in the authentication process. First, the one-way identity authentication process is introduced, as shown in Figure 1. Device d_i initiates the authentication request, device d_i performs the verification, and the authentication information of d_i is named $Tocken_i$.

$$Token_{ij} = T_i \| N_i \| PID_{d_i} \| Text \| s_{sk_{d_i}}(T_i \| N_i \| PID_{d_i} \| Text) \quad (7)$$

Among these, T_i is a timestamp, N_i is a random number, PID_{d_i} is a pseudo-anonymity generated by KGC for d_i, Text is the information transmitted by device d_i to device d_i via $Tocken_{ij}$, with the communication key negotiation portion placed in this field, and $s_{sk_i}(X)$ refers to the signature of message X by the private key sk_{d_i} of device d_i requesting authentication.

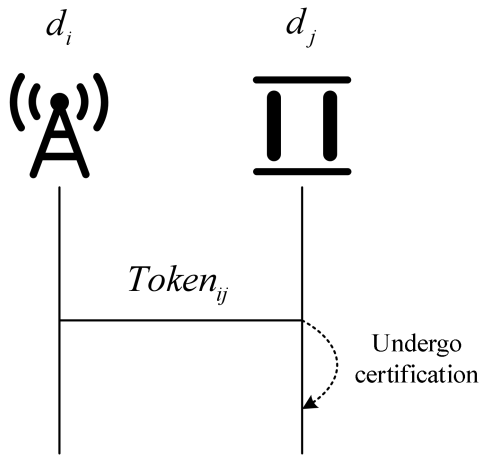

Figure 1. One-way Authentication.

Device d_i sends $Tocken_i$ to Device d_i. After Device d_i receives $Tocken_i$, it first retrieves the public key of Device d_i. Device d_j uses the public key of Device d_i to verify $Tocken_{ij}$, signs the unsigned message, and compares it with the signature received in $Tocken_{ij}$ to complete the verification process. If it is a two-way verification, as shown in Figure 2, then devices d_i and d_j swap roles, device d_i becomes the verifier, and device d_i becomes the identity verification requester, repeating the above

process.

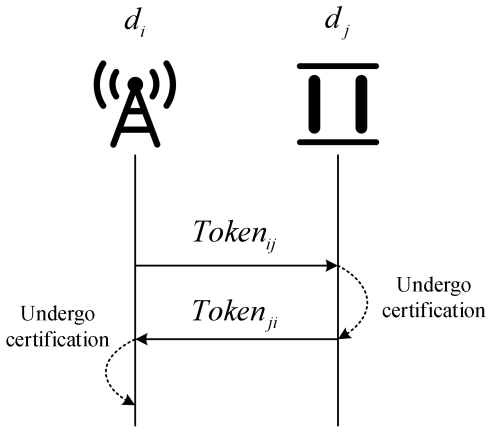

Figure 2. Two-way Authentication.

(2) Cross-domain authentication

Assume that device d_i^A initiates an authentication request to device d_i^B. This is cross-domain authentication, as shown in Figure 3.

Cross-domain authentication requires the use of BAN to retrieve the latest information from each domain for identity verification. During the cross-domain authentication process, two-way identity verification is completed, and a session key is established. Device d_i^A generates a message $M = T_i \| N_i \| PK_i \| PD_{d_i^A}$, T_i is the timestamp, N_i is a random number, $PK_i = r_i G$, PK_i is the session public key, where $r_i \in Z_p$ is the session private key randomly selected by d_i^A. d_i^A sends the message and signature request to KGC_A, KGC_A After receiving the message and signature request, KGC_A searches the local database for $sk_{d_i^A}$. If it does not exist or has expired, it will regenerate the private key for d_i^A. KGC_A performs the following calculation to obtain the signature (h, S). The calculation is as follows:

$$w = g^r \tag{8}$$

$$h = H_2(M_{d_i^A} \| w, N) \tag{9}$$

$$l = (r - h) \bmod N \tag{10}$$

$$S = [l]\varphi sk_{d_i^A} \tag{11}$$

Subsequently, KGC_A sends the signature back to d_i^A. To perform authentication, d_i^A sends an authentication request with M and signature (h, S) to d_i^B. The verification process is as follows: when device d_i^B receives the authentication message from device d_i^A, d_i^B first checks whether T_i is fresh. If the requirement is met, d_i^B sends the authentication request with $M_{d_i^A}$ and signature $(h, S)_{d_i^A}$ to KGC_B. KGC_B first searches the local database for the public key of d_i^A (i.e., pseudo-anonymous $PID_{d_i^A}$). If it is not found or $PID_{d_i^A}$ has expired, KGC_B requests the latest information about domain A from BAN_B, BAN_B queries the latest record of domain A from the blockchain by calling the query chaincode, and sends the specific information of the latest domain A back to KGC_B, KGC_B then performs the following calculations based on the received latest information:

$$t = g^h \tag{12}$$

$$h_1 = H_1(PID_{d_i^A} \| hid, N) \tag{13}$$

$$P = [h_1]P_2 + P_{Pub_A} \tag{14}$$

$$u = e(S, P) \tag{15}$$

$$w^{'} = u \cdot t \tag{16}$$

$$h_2 = H_2(M_{d_i^A} \| w^{'}, N) \tag{17}$$

Finally, verify $h_2 = h$. If it is valid, the verification is successful, and one-way authentication is completed. Next, perform two-way authentication. Then, d_i^B randomly selects $n_i \in Z_q$ as its own session private key, calculates the public key $PK_j = n_i G$, and calculates the session key $TK_{ii} = n_i PK_i = n_i r_i G$. d_i^B generates the message $M = T_i \| N_i \| PK_i \| PID_i$, T_i is the timestamp, and N_i is the random number. d_i^B sends the message and signature request to KGC_B . KGC_B Upon receiving the message and signature request, it searches its local database for $sk_{d_i^B}$. If it does not exist or has expired, it regenerates the private key for d_i^B . KGC_B It generates a signature on the message M_i and sends the signature $(h, s)_{a^n}$ back to d_i^B , d_i^B It sends an authentication request to d_i^A with the message $M_{d_i^B}$ and signature $(h, s)_{d_i^B}$ attached.

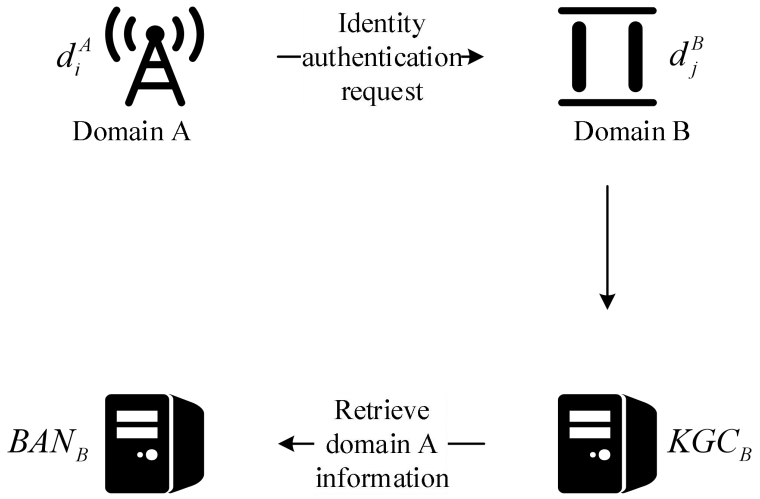

Figure 3. Cross-domain Authentication.

2.2. Blockchain-based IoT device data traceability

In the current process of IoT data flow, as discussed in the previous research of this paper, traceability information is often subject to forgery and tampering due to the drawbacks of centralized storage models, making it impossible to trace the true data source and data flow process through data traceability [42]. Therefore, to address the above issues, this section will propose an IoT data traceability system model based on blockchain technology and provide a detailed description of each layer. The management of traceability information primarily encompasses three aspects: collection, storage, and querying. Therefore, the blockchain-based IoT data traceability model is divided into three layers: the traceability information collection layer, the traceability information storage layer, and the traceability information display layer. The specific contents of each layer are as follows:

2.2.1. Traceability Information Collection Layer

The collection of traceability information is related to data origin tracking methods. If a data origin tracking method based on annotation is used, traceability information related to the data is recorded directly using annotations. If a data origin tracking method based on inverse functions is used, the inverse function is simply set up, and there is no need to store the entire traceability information process; only a small amount of metadata knowledge information is recorded. The data origin tracing method used in

this paper is a DAG-based tracing method. If annotation is used to record the traceability information of data during data flow, it is difficult to implement, and the inverse function-based method is not applicable in this environment. Therefore, the method for collecting traceability information during the IoT data flow process should refer to the DAG-based tracing method and the specific flow environment of IoT data. Regarding the issue of traceability information collection methods, the participants in the IoT data flow process can be broadly categorized into four key roles: providers, processors, sellers, and users. These four roles have a many-to-many relationship. To better collect traceability information for data entities, it is necessary to integrate the actual users of these four roles into the same ecosystem to facilitate traceability information collection. Data flows between users, and the key to this flow is enabling users to transact data. Therefore, the collection of traceability information cannot be separated from the data platform that provides data services. Traceability information can be collected by monitoring the types of operations data undergoes within the platform (e.g., data upload, processing, download, transactions, etc.). IoT datasets are stored using various methods, such as cloud storage and traditional databases. By monitoring user operations on IoT datasets, traceability information regarding data changes can be obtained. Additionally, traceability information can be collected by gathering key data from web scripts related to data operations and by analyzing log files within the system.

2.2.2. Traceability Information Storage Layer

The traditional centralized traceability information storage model poses risks such as tampering and falsification of traceability information. Blockchain technology can effectively mitigate these risks. The primary reasons for using blockchain to store traceability information are as follows:

(1) Preventing single points of failure: Each user in the data flow process joins the blockchain network as a node, and traceability information is stored in various blocks of the blockchain and carried by a series of transactions between user nodes. Since each node in the blockchain network backs up all on-chain transaction information, data loss will not occur due to the failure of a single node.

(2) Tamper-proof traceability information: Traceability information recorded on the blockchain is jointly maintained by all nodes, and each node within the blockchain network can obtain the traceability information of a specific data entity. This method leverages the advantages of blockchain technology, such as tamper-proofing and open consensus, to achieve reliable storage of traceability information.

(3) Easy data recovery: In traditional centralized storage models, once the central database is damaged or maliciously attacked, resulting in data loss or destruction, if there is no comprehensive data recovery and backup strategy in place, it will be impossible to recover the data. In a blockchain network, each node continuously updates and stores all block information. If a node loses all block information, it can obtain all previous block information from neighboring nodes.

Based on the above three reasons, this paper proposes to store all traceability information on the blockchain, with the specific content stored on the chain including each traceability record and the transaction hash values associated with the additional traceability information. Additionally, through the traceability information management contract, the mapping relationship between the data ID written on the blockchain and the transaction hash values corresponding to the traceability information associated with this data ID is established. The specific detailed information about the data entity, data activity information, and data agent detailed information are also written onto the blockchain through the corresponding smart contracts.

2.2.3. Traceability Information Display Layer

The traceability information display layer primarily displays the results of traceability information queries. This layer serves as the top-level component in the layered model of a blockchain-based IoT data traceability system, providing users with a visual representation of the flow process and operational processing information of traceability objects. When conducting specific traceability queries, the system directly traces the information from the blockchain. To visualize the traceability information, the traceability results obtained from the blockchain are mapped into RDF-formatted data during the visualization process, and this RDF-formatted data is validated and stored. Subsequently, the SPARQL query language recommended by the W3C Working Group is used to query the RDF-formatted data. After the query, the results are uploaded to ProvStore to generate a visual representation for users. The specific process of traceability information visualization is shown in Figure 4. SPARQL query language, RDF, and ProvStore are all within the scope of traceability information visualization research but are beyond the scope of this paper, so further details are not provided here.

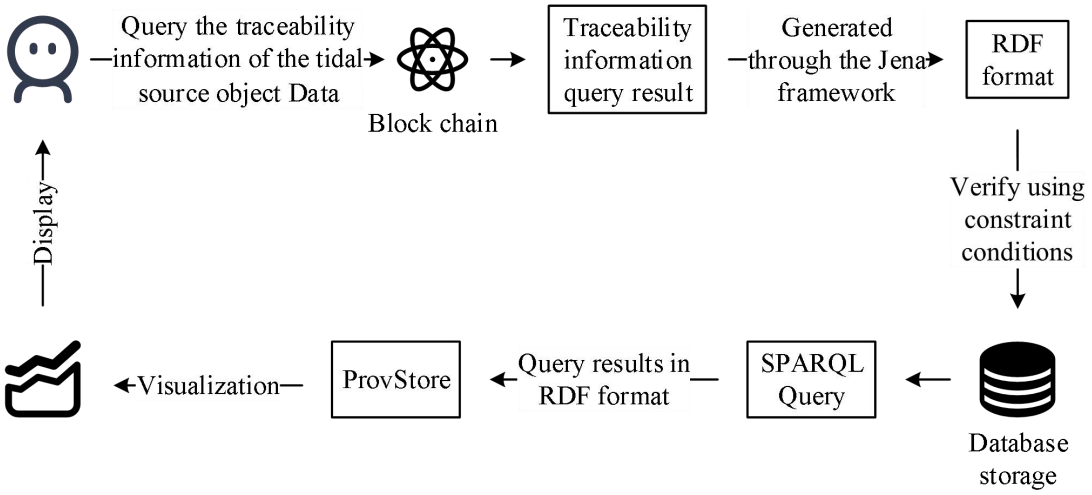

Figure 4. Specific flow chart of provenance information visualization.

3. Design and implementation of the research plan

3.1. Functional Design of the Research Plan

3.1.1. Identity Authentication Function Design

Trusted identity verification is the first line of defense for secure data traceability. Data traceability involves numerous participants, including data sources, data transmitters, data reviewers, and data users, all of which are important components of the system and play their respective roles in maintaining normal service and stable operation. It is necessary to ensure the uniqueness and legitimacy of each role's identity and to verify the identity of each role at every step of the process. IoT identity verification can be divided into trustworthy authentication of IoT devices and trustworthy identity verification of operators. IoT devices possess certain sensing, computing, and communication capabilities and serve as the "source" of data. Users are the operators of IoT devices, who may use the devices directly or indirectly through device control. This section will describe feasible trustworthy identity verification processes in conjunction with the primary roles involved in the system. This paper primarily focuses on user-centric identity verification scenarios, including scenarios where users directly interact with IoT devices for identity verification, as well as scenarios where users verify each other's identities.

3.1.2. Traceability Data Storage Function Design

To achieve secure and trustworthy data traceability, the definition of traceability data and its secure and reliable storage are critical. This paper implements traceability data storage through two parts. First, based on the PROV data model, a traceability data model is established in an IoT environment to describe traceability records, enabling the tracking of data changes and the identification of entities causing such changes. Then, based on the traceability data model, a set of smart contracts for traceability data management is designed. The contract encapsulates the logic for writing and reading traceability data on the blockchain by all parties involved in the traceability process, including the structure of the traceability data. After deploying the compiled contract on the blockchain, when a transaction meets the predefined conditions, the contract is automatically executed to achieve data storage. Due to the characteristics of the blockchain, once data is uploaded to the chain, it is difficult to tamper with, thereby ensuring the reliability of the traceability data.

3.1.3. Design of traceability data verification functions

The traceability data verification function can validate the correctness of the proposed solution. After all parties involved store the data on the blockchain, other nodes can retrieve the traceability data of the traceability object from the blockchain to verify the authenticity of the traceability data. The contract includes the logic for traceability data verification. Here, we consider two scenarios: normal data that has not been tampered with and abnormal data that may have been tampered with. If the data has not been tampered with, other nodes can retrieve authentic traceability data from the blockchain, including attributes of the traceability object, agents, and a series of operations performed. Conversely, if the data has been tampered with, the retrieved traceability data will be empty.

3.2. Overall Implementation Plan

This paper combines the functional structure of the Internet of Things (IoT) with the division of functional modules in the solution to design a functional implementation architecture for blockchain-based data traceability. The architecture consists of four parts: IoT devices, a blockchain network built with blockchain nodes, smart contracts, and the front end of the application. The following sections will elaborate on each part.

(1) IoT devices

Looking from the bottom up, IoT devices (mainly various types of data collection devices, including sensors, RFID tags, readers, etc.) are used to collect data information from the environment and transmit the data to gateway nodes or user nodes for processing, serving as the source of blockchain data.

(2) Blockchain Network

The blockchain network is jointly built by geth clients to store traceability data and provide basic traceability data query services. Blockchain nodes participate in and maintain the blockchain, are responsible for generating and storing block data, and provide the computing power required for the blockchain network.

(3) Smart Contract Section

In the smart contract section, Solidity language can be used to flexibly write smart contract scripts that are applicable to the application and must be strictly executed by all nodes on the network. These scripts describe the business logic implemented based on the blockchain data scheme, mainly including the logic of traceability data storage and traceability data query (i.e., verification).

(4) Front-end page display

On the front-end display page, based on the normal operation of the blockchain network and the successful deployment of contracts, all parties involved can use the front-end page to store traceability data and verify traceability records.

3.3. Implementation of functional modules

3.3.1. Identity Authentication

The identity verification module primarily includes device identity verification and identity verification for all parties involved in data traceability. Only after the device's initial identity is established is the collected information valid. Once the device is detected to already have a public key and private key, it cannot be initialized again, as re-initialization would result in changes to the public key and private key, effectively altering the identity and leading to the loss of data ownership. The public key can be derived from the private key, but the private key cannot be inferred from the public key. Additionally, the Keccak256 hash algorithm ensures that data in transit cannot be easily tampered with. This effectively addresses issues such as identity spoofing at the perception layer and data tampering during network transmission, thereby ensuring the normal operation of application-layer functions.

3.3.2. Storage of traceability data

Traceability data storage is implemented via smart contracts. Ethereum contracts offer two data storage methods: one utilizes account storage, and the other employs events, also known as log storage. Both methods are implemented within the contract code. After the contract is written, it is first compiled, then the compiled smart contract is deployed to the blockchain. Upon successful deployment, the smart contract's address and ABI are returned, with the ABI including variables, events, and callable methods. In the application frontend, uploading a file yields its hash value for subsequent data authenticity verification. The uploader can add descriptive information about the file based on the actual business scenario. Subsequently, the authenticity of the traceability data can be verified by comparing the hash value of the re-calculated traceability object with the hash value obtained from the blockchain.

3.3.3. Verification of traceability data

The verification of traceability data is based on the storage of traceability data. Once traceability data is stored on the blockchain network, traceability records can be retrieved from the blockchain to verify the traceability data. The contract provides service interfaces to process data. After the contract is compiled, it generates an ABI. By interacting with the contract, traceability data can be stored and queried. According to the contract design, the information contained in the traceability data should include the displayed content. This paper uses a front-end application to query traceability data in a visual manner.

4. Empirical Research Analysis

4.1. Analysis of IoT Device Identity Verification

This section examines the effectiveness of the research design proposed in this paper from the perspectives of IoT devices and blockchain identity authentication services. The specific details are described below:

4.1.1. Performance testing of IoT devices

(1) IoT device end testing environment

The 11oT devices used for testing in this section are based on the STM32F103ZET6 microcontroller, featuring a Cortex-M3 core, a clock speed of 72 MHz, 64 KB of RAM, and 512 KB of FLASH memory. In the identity authentication system designed in this paper, the certificate-less signature algorithm is implemented using the GMP library and the JediPairing library, with the elliptic curve being BLS12-381, which has a security level of 128 bits.

(2) Calculation of overhead analysis

During the registration phase and identity verification phase, the two IIoT devices participating in the verification process, along with their respective domain-specific KGCs and BSNs, must perform a certain number of cryptographic operations. These cryptographic operations are the primary factors influencing the computational overhead performance of this scheme. This section summarizes the main cryptographic operations involved in the identity verification process, including scalar multiplication (SM) and point addition (PA) on the BLS12-381 curve. Some cryptographic operations with relatively minimal performance overhead have not been considered, as they only have a negligible impact on computational performance.

Generally speaking, KGC and BSN are servers with higher configurations within their respective domains, making them suitable for handling complex computational tasks. In contrast, IIoT devices are typically cost-effective and require reliable, stable participation in production tasks, so they should not be burdened with computationally intensive cryptographic operations. To assess the computational overhead of this scheme, we analyzed the computational burden borne by entities in each domain under real-world operating conditions based on the identity authentication process described earlier, as shown in Table 1. In this scheme, the majority of the computational burden generated by identity authentication is borne by KGC and BSN, making this scheme suitable for IIoT devices with limited computational capabilities.

Table 1. Cryptographic operations performed by each entity and the number of times.

Symbol	Total number	In the registration stage	During the identity verification stage
D_i	4SM+PA	2SM+PA	2SM
KGC	2SM+2PA	2SM+2PA	

To further demonstrate the advantages of this scheme in terms of computational overhead, this section compares identity authentication schemes D and E, which also use certificate-free technology, summarizes the number and types of cryptographic operations, and compares them with this scheme, as shown in Table 2.

Table 2. Comparison of operation types and frequencies.

Entity	D	E	This plan
A	3SM+2PA	2SM	2SM
B	3SM+4PA	3SM	2SM

When testing with IoT devices in a test environment, the average time required for the two cryptographic operations PA and SM is 3.42 ms and 309.11 ms, respectively. Based on this, the computational time required by the two entities participating in the identity authentication phase can be compared as shown in Figure 5. The results show that in a single identity authentication process, the total computational time for the two IoT devices in this scheme is 65.37% of that in Scheme D and 79.46% of that in Scheme E. This is because this scheme does not require IoT devices to directly verify the results of certificate-less signatures, and it integrates the certificate-less signature algorithm and key exchange process, enabling IoT devices to perform key negotiation simultaneously during identity authentication. Furthermore, as shown in the table, the computational overhead for the two devices undergoing mutual authentication is nearly identical in this scheme. This is because both IoT devices use the same certificate-less signature technology and perform the same cryptographic operations during the

authentication process.

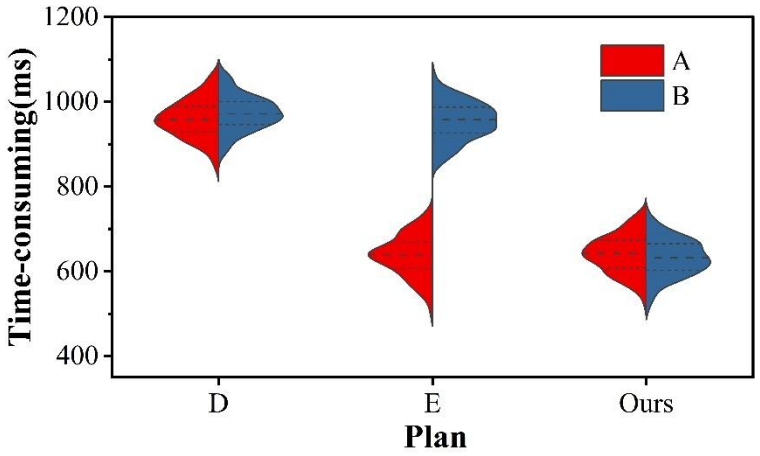

Figure 5. Comparison of computational costs.

(3) Communication Overhead Analysis

This section summarizes and compares the communication overhead of IoT devices during the identity authentication process for identity authentication schemes D and E, which also use certificate-free technology, and this scheme, as summarized in Table 3.

Table 3. Comparison of communication overhead of authentication schemes.

Identity authentication stage		
Plan	Send	Receive
D	$\left\|Z_q^*\right\|+2\left\|G\right\|+\left\|t\right\|$	$\left\|MAC\right\|$
E	$3\left\|Z_q^*\right\|+4\left\|G\right\|+\left\|t\right\|$	$\left\|MAC\right\|+\left\|G\right\|$
Ours	$\left\|Z_q^*\right\|+\left\|G\right\|+\left\|t\right\|$	$\left\|S\right\|+\left\|G\right\|+\left\|t\right\|$

According to the IoT endpoint testing conditions described, this solution implements cryptographic operations on the BLS12-381 elliptic curve using the Jedi Pairing library. $\left|G\right|=96bytes$, $\left|Z_q^*\right|=64bytes$. Additionally, $\left|MAG\right|=64bytes$, $\left|t\right|=4bytes$, this solution includes $\left|M\right|=\left|t+TID\right|=4bytes+16bytes=20bytes$, $\left|S\right|=64bytes$. Based on the above parameters, the comparison results between this solution and Solutions D and E are shown in Figure 6. In this scheme, during a single authentication process, a single IoT device consumes 85 bytes and 89 bytes of communication bandwidth for sending and receiving, respectively. The total communication consumption for schemes D, E, and this scheme is 158 bytes, 365 bytes, and 174 bytes, respectively. The communication overhead of this scheme is 9.20% higher than that of scheme D and 47.67% of that of scheme E.

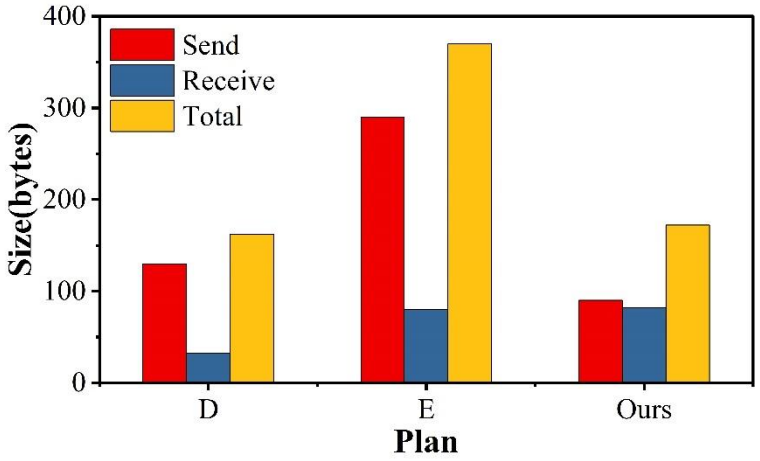

Figure 6. Comparison of communication overhead.

4.1.2. Blockchain Identity Authentication Server Performance Testing

(1) Server-side test environment

This paper establishes a test environment based on Hyperledger Fabric 1.8.2, deployed in Solo mode, i.e., a single-node ordering mode. In this environment, only one order node is used, which is responsible for ordering and generating blocks for peer nodes. During system testing, 2, 3, 4, 5, and 6 organizations were set up, representing the domains participating in the cross-domain identity authentication process. Each domain contains one peer node and one client node, with the peer node's functionality handled by the BSN in this system and the client node's functionality handled by KGC. All these nodes run in Docker containers on a local computer. Additionally, the benchmarking tool used in this test is Hyperledger Caliper, and the deployment status of each blockchain node is shown in Table 4.

Table 4. Deployment of blockchain nodes.

Affiliated organization	Name	Port
Org1	Peer0.org1	7026
Org2	Peer0.org2	8026
Org3	Peer0.org3	9026
Org4	Peer0.org4	10026
Org5	Peer0.org5	11026
Org6	Peer0.org6	12026
Org1	Ca.org1	7027
Org2	Ca.org2	8027
Org3	Ca.org3	9027
Org4	Ca.org4	10027
Org5	Ca.org5	10027
Org6	Ca.org6	11027
-	Orderer	7025

(2) Identity authentication service performance testing

This section tests the performance of the blockchain identity authentication service when 2, 3, 4, 5, and 6 domains participate in the blockchain. Each domain contains one KGC and one BSN, where the BSN acts as a Peer node and the KGC acts as a Client node. In Fabric, the Peer participates in smart contract execution and maintains the blockchain ledger, while the Client initiates transactions. All these nodes run in Docker containers. The average latency and throughput of the identity authentication service when calling smart contract 2 were measured at different send rates (35–135 TPS), as shown in Figures 7 and 8. In Figure 7, when the number of domains is 4, 5, and 6, the average transaction latency is initially low and stable, but then suddenly increases as the send rate increases. In Figure 8, when the number of domains is 5 or 6, the throughput rate first increases linearly and then remains stable. In summary, the more domains in the blockchain network, the lower the maximum throughput and the higher the average latency. As the number of nodes in the blockchain network increases, the number of nodes required to endorse and validate transactions also increases, making the aforementioned phenomena reasonable and

validating the effectiveness of the design scheme proposed in this paper.

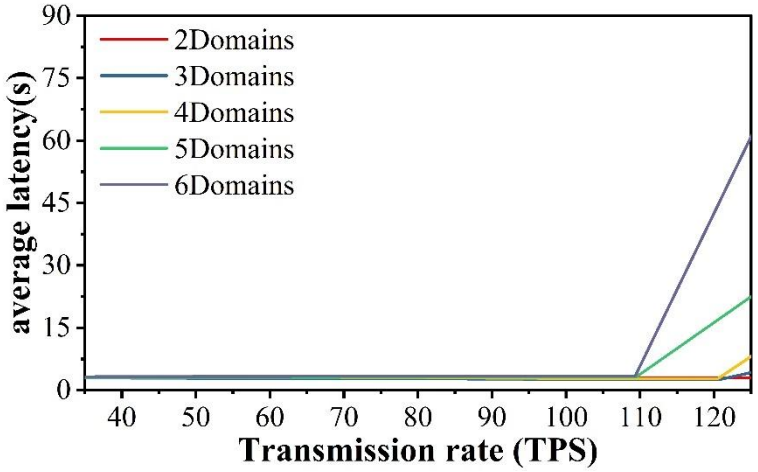

Figure 7. Authentication average latency.

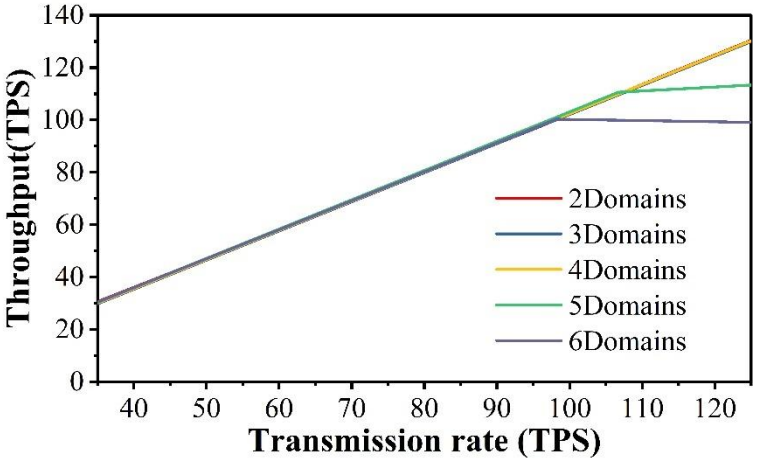

Figure 8. Authentication throughput.

4.2. Traceability and analysis of IoT device data

4.2.1. Traceability Method Verification Analysis

To test the impact of the data tracing method proposed in this paper on performance, this section conducts tests from two aspects: the additional time overhead of tracing information collection on HQL execution and the efficiency of tracing tracking.

(1) Traceability information collection time consumption experiment

This paper extends blockchain technology by adding a traceability collection module to IoT device authentication. Therefore, this experiment aims to conduct a comparative test of Hive before and after the extension to examine the impact of the traceability collection process on overall performance. The experiment performs the same HQL task on data of different scales, testing the comparison of HQL execution time before and after adding the traceability information collection module under different input data scales, as well as the time overhead of traceability collection during the HQL process. The test experiment is set up in 8 groups, with data volumes ranging from 128MB to 1024MB, each tested for execution time, with the average value of three executions in the same experiment taken for each. The experimental results are shown in Figure 9. As shown in Figure 9, both the HQL execution time with and without the trace collection module increases linearly with the input data volume, but the time overhead difference between the two is small. Additionally, throughout the entire execution process, the time overhead of trace collection approaches zero. Therefore, the additional time overhead of trace collection on HQL can be neglected.

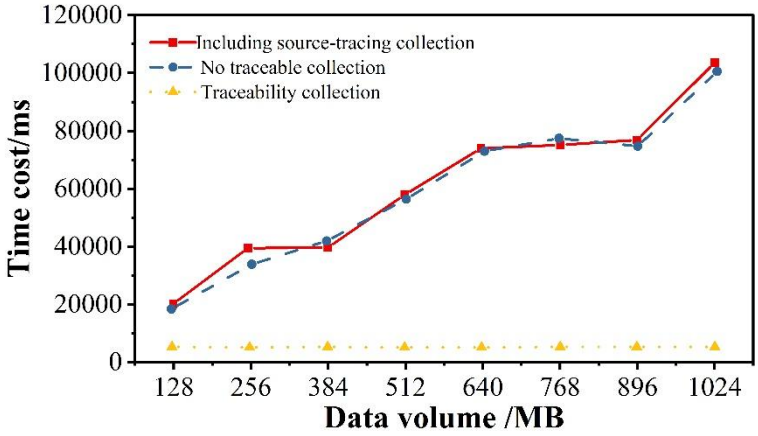

Figure 9. Experiment on the time cost of traceability information collection.

(2) Traceability efficiency experiment

To achieve traceability tracking based on given data items, this paper proposes a traceability tracking method based on directed acyclic graphs (DAGs), utilizing graph theory query algorithms to track the origin and generation process of data. The time complexity of traceability tracking was tested under different input data scales, with the execution time for each test being the average of three runs of the same experiment. The experimental results are shown in Figure 10. As can be seen, as the data scale increases, the traceability tracking time remains consistently around 280–300 ms, with minimal fluctuations. This indicates that the traceability tracking method proposed in this paper demonstrates overall good performance.

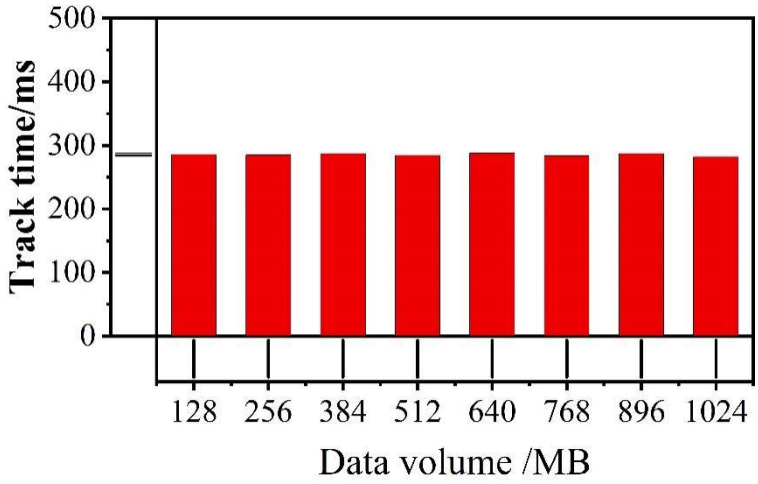

Figure 10. Experiment on traceability and tracking efficiency.

4.2.2. Performance Testing

Use the Caliper testing framework to test Hyperledger Fabric. The Caliper testing framework is a tool developed specifically for testing Hyperledger consortium chains. After setting the configuration parameters, performance testing can be conducted on the traceability system network established in this article. The testing primarily includes two types: write and query. The performance testing results are shown in Table 5. The write test evaluates the performance of writing to the blockchain ledger, while the query test assesses the performance of reading from the ledger. Based on actual application scenarios, the number of traceability queries on the blockchain ledger exceeds the number of write operations. In the test benchmark configuration, the number of write operations is set to 4,000, and the number of query operations is also set to 4,000.

Table 5. Performance test results.

Test round	Type	Success	Failure	Delivery rate /TPS	Average delay /s	Throughput /TPS
1	write	4000	0	49	1.64	46
2	write	4000	0	105	10.05	58
3	write	4000	0	188	16.49	111
4	write	3996	4	405	18.18	151
5	write	3998	2	425	19.08	135
6	write	3996	4	408	21.08	129
7	query	4000	0	24	0.14	45
8	query	4000	0	96	0.18	94
9	query	4000	0	197	4.08	177
10	query	4000	0	379	12.36	319
11	query	4000	0	436	18.59	288
12	query	4000	0	446	17.59	304

(1) Analysis of throughput and success rate test results

The write-type test was conducted over six rounds, with each round initiating a total of 4,000 transactions on the network. The number of requests per second was set to 100, 200, 400, 800, 1,600, and 3,200 for each round. Based on the experimental results, the overall network transaction success rate exceeded 95%, ensuring the system could operate normally. Network congestion occurred when the actual send rate approached approximately 425 TPS. Under the tested local machine configuration, the send rate upper limit was 450 TPS, and when the send rate reached approximately 425 TPS, the transaction throughput reached a peak of 151 TPS. Continuing to increase the transmission rate does not result in further increases in the actual transmission rate, and the throughput remains largely stable. The query type test was conducted over six rounds. Each round involved initiating a total of 4,000 transaction requests to the network, with the number of requests per second set to 100, 200, 400, 800, 1,600, and 3,200 respectively. Based on the experimental results, the query request success rate was 100%, ensuring the system operated normally. When the system send rate reached approximately 450 TPS, it reached the actual send rate limit. The results showed that when the request rate reached 379 TPS, the actual send rate was 319 TPS, and the throughput reached a peak of approximately 320 TPS.

(2) Analysis of time delay test results

For write-type tests, when the send rate is below 25 times per second, the average transaction latency is within 1 second, enabling quick response to user operations and meeting the requirements of a large number of write operations. However, when the request send rate exceeds 1,000 times per second, the average transaction latency increases to over 8 seconds, with the highest average latency reaching 21.08 seconds in the test. For query-type tests, when the request rate does not exceed 100 requests per second, the maximum average transaction latency is around 8 seconds, while lower send rates result in transaction delays of less than 2 seconds, providing fast response times and meeting user requirements for querying product data during traceability. When the send rate exceeds 430 requests per second, transaction latency approaches 18.6 seconds.

5. Conclusion

This paper conducts research on identity authentication and data traceability for IoT devices based on blockchain technology, supported by relevant theoretical knowledge, and thoroughly examines the research plan from multiple perspectives. The main research conclusions are as follows:

(1) The communication consumption of the three different IoT device authentication schemes is 158 bytes, 365 bytes, and 174 bytes, respectively. The value of the scheme in this paper is 9.20% higher than that of Scheme D and only 47.67% of that of Scheme E. In addition, the authentication time of the scheme in this paper is 65.37% and 79.46% of that of Schemes D and E, respectively. Overall, the proposed scheme outperforms Schemes D and E, validating the applicability of the proposed IoT device authentication scheme.

(2) The traceability method described in this paper was tested and analyzed in terms of time consumption and traceability efficiency. As the data volume increased from 128 MB to 1024 MB, the time consumption tended toward zero, and the corresponding data traceability time remained at around 280–300 ms with very little fluctuation, proving the application value of the IoT data traceability method from a blockchain technology perspective.

Fundings
Major Intra-School Science and Technology Project: Research and Application of Visual Perception Technology for Multimodal User Behavior Data(No.2024X010-KXD).

References

1. Salman, O., Abdallah, S., Elhajj, I. H., Chehab, A., & Kayssi, A. (2016, June). Identity-based authentication scheme for the Internet of Things. In 2016 IEEE Symposium on Computers and Communication (ISCC) (pp. 1109-1111). IEEE.
2. Houhamdi, Z., & Athamena, B. (2020). Identity identification and management in the internet of things. Int. Arab J. Inf. Technol., 17(4A), 645-654.
3. Liu, Y., Wang, J., Li, J., Niu, S., & Song, H. (2021). Machine learning for the detection and identification of Internet of Things devices: A survey. IEEE Internet of Things Journal, 9(1), 298-320.
4. Kahtyat, S. (2024, November). Designing a Decentralized Identity Verification Platform. In Norsk IKT-konferanse for forskning og utdanning (No. 3).
5. Bhattacharya, S., Najana, M., & Khanna, A. (2024). Decentralized identity verification via smart contract validation: Enhancing PKI systems for future digital trust. International Journal of Global Innovations and Solutions (IJGIS).
6. Saravanan, V., Sumalatha, A., Reddy, D. N., Ahamed, B. S., & Udayakumar, K. (2024, October). Exploring Decentralized Identity Verification Systems Using Blockchain Technology: Opportunities and Challenges. In 2024 5th IEEE Global Conference for Advancement in Technology (GCAT) (pp. 1-6). IEEE.
7. Liu, Y., He, D., Obaidat, M. S., Kumar, N., Khan, M. K., & Choo, K. K. R. (2020). Blockchain-based identity management systems: A review. Journal of network and computer applications, 166, 102731.
8. Li, L., Qu, H., Wang, H., Wang, J., Wang, B., Wang, W., ... & Wang, Z. (2022). A blockchain-based product traceability system with off-chain EPCIS and IoT device authentication. Sensors, 22(22), 8680.
9. Li, X., Jiang, P., Chen, T., Luo, X., & Wen, Q. (2020). A survey on the security of blockchain systems. Future generation computer systems, 107, 841-853.
10. Sheth, H., & Dattani, J. (2019). Overview of blockchain technology. Asian Journal For Convergence In Technology (AJCT) ISSN-2350-1146.
11. Cai, W., Wang, Z., Ernst, J. B., Hong, Z., Feng, C., & Leung, V. C. (2018). Decentralized applications: The blockchain-empowered software system. IEEE access, 6, 53019-53033.
12. Sabrina, F., Li, N., & Sohail, S. (2022). A blockchain based secure IoT system using device identity management. Sensors, 22(19), 7535.
13. Babu, E. S., Dadi, A. K., Singh, K. K., Nayak, S. R., Bhoi, A. K., & Singh, A. (2022). A distributed identity-based authentication scheme for internet of things devices using permissioned blockchain system. Expert Systems, 39(10), e12941.
14. Namasudra, S., Sharma, P., Crespo, R. G., & Shanmuganathan, V. (2022). Blockchain-based medical certificate generation and verification for IoT-based healthcare systems. IEEE Consumer Electronics Magazine, 12(2), 83-93.
15. Gong, L., Alghazzawi, D. M., & Cheng, L. (2021). BCoT sentry: A blockchain-based identity authentication framework for IoT devices. Information, 12(5), 203.
16. Venkatraman, S., & Parvin, S. (2022). Developing an IoT identity management system using blockchain. Systems, 10(2), 39.
17. Sharma, P., Moparthi, N. R., Namasudra, S., Shanmuganathan, V., & Hsu, C. H. (2022). Blockchain-based IoT architecture to secure healthcare system using identity-based encryption. Expert Systems, 39(10), e12915.
18. Dorri, A., Roulin, C., Pal, S., Baalbaki, S., Jurdak, R., & Kanhere, S. S. (2022). Device identification in blockchain-based internet of things. IEEE Internet of Things Journal, 9(24), 24767-24776.
19. Laroiya, C., Bhatia, M. K., Madan, S., & Komalavalli, C. (2022, September). IoT and blockchain-based method for device identity verification. In International Conference on Innovative Computing and Communications: Proceedings of ICICC 2022, Volume 1 (pp. 269-280). Singapore: Springer Nature Singapore.
20. Feng, H., Wang, X., Duan, Y., Zhang, J., & Zhang, X. (2020). Applying blockchain technology to improve agri-food traceability: A review of development methods, benefits and challenges. Journal of cleaner production, 260, 121031.
21. Zhu, P., Hu, J., Li, X., & Zhu, Q. (2021). Using blockchain technology to enhance the traceability of original achievements. IEEE Transactions on Engineering Management, 70(5), 1693-1707.
22. Fan, Z. P., Wu, X. Y., & Cao, B. B. (2022). Considering the traceability awareness of consumers: should the supply chain adopt the blockchain technology?. Annals of Operations Research, 309(2), 837-860.
23. Centobelli, P., Cerchione, R., Del Vecchio, P., Oropallo, E., & Secundo, G. (2022). Blockchain technology for bridging trust, traceability and transparency in circular supply chain. Information & Management, 59(7), 103508.
24. Cui, H., Chen, Z., Xi, Y., Chen, H., & Hao, J. (2019, August). IoT data management and lineage traceability: A blockchain-based solution. In 2019 IEEE/CIC International Conference on Communications Workshops in China (ICCC Workshops) (pp. 239-244). IEEE.
25. Xu, Q., Aung, K. M. M., Zhu, Y., & Yong, K. L. (2017). A blockchain-based storage system for data analytics in the internet of things. In New Advances in the Internet of Things (pp. 119-138). Cham: Springer International Publishing.

26. Siddiqui, M. S., Syed, T. A., Nadeem, A., Nawaz, W., & Albouq, S. S. (2020). BlockTrack-L: A lightweight blockchain-based provenance message tracking in IoT. International Journal of Advanced Computer Science and Applications, 11(4).
27. Zhao, Y., Yang, X., Yu, Y., Qin, B., Du, X., & Guizani, M. (2021). Blockchain-based auditable privacy-preserving data classification for Internet of Things. IEEE Internet of Things Journal, 9(4), 2468-2484.
28. Kowsalya, K., Rani, R. P. J., Bhiyana, M., Saini, M., & Patil, P. P. (2023, May). Blockchain-Internet of things-Machine Learning: Development of Traceable System for Multi Purposes. In 2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE) (pp. 1112-1115). IEEE.
29. Caro, M. P., Ali, M. S., Vecchio, M., & Giaffreda, R. (2018, May). Blockchain-based traceability in Agri-Food supply chain management: A practical implementation. In 2018 IoT Vertical and Topical Summit on Agriculture-Tuscany (IOT Tuscany) (pp. 1-4). IEEE.
30. Wang, D., Wang, H., & Fu, Y. (2021). Blockchain-based IoT device identification and management in 5G smart grid. EURASIP Journal on Wireless Communications and Networking, 2021(1), 125.
31. Goyat, R., Kumar, G., Alazab, M., Conti, M., Rai, M. K., Thomas, R., ... & Kim, T. H. (2020). Blockchain-based data storage with privacy and authentication in internet of things. IEEE Internet of Things Journal, 9(16), 14203-14215.
32. Nag, A., Hassan, M. M., Das, A., Kaushal, C., & Thakur, D. (2024). Blockchain-based identity authentication for Internet of Things systems: A comprehensive survey. Artificial intelligence and Internet of Things based augmented trends for data driven systems, 163-180.
33. Bao, Z., He, D., Khan, M. K., Luo, M., & Xie, Q. (2022). PBidm: Privacy-preserving blockchain-based identity management system for Industrial Internet of Things. IEEE transactions on industrial informatics, 19(2), 1524-1534.
34. Zhang, A., Zhang, P., Wang, H., & Lin, X. (2020). Application-oriented block generation for consortium blockchain-based IoT systems with dynamic device management. IEEE Internet of Things Journal, 8(10), 7874-7888.
35. Liu, Y., Liu, A., Xia, Y., Hu, B., Liu, J., Wu, Q., & Tiwari, P. (2023). A blockchain-based cross-domain authentication management system for IoT devices. IEEE Transactions on Network Science and Engineering, 11(1), 115-127.
36. Wang, Y., Zhang, K., Zhao, X., & Hu, X. (2024). BTIA-IME: A blockchain-based trusted interactive architecture for intelligent manufacturing equipment. Internet of Things, 25, 101026.
37. Kamboj, P., Khare, S., & Pal, S. (2021). User authentication using Blockchain based smart contract in role-based access control. Peer-to-Peer Networking and Applications, 14(5), 2961-2976.
38. Gaur, R., Prakash, S., Kumar, S., Abhishek, K., Msahli, M., & Wahid, A. (2022). A machine-learning–blockchain-based authentication using smart contracts for an IoHT system. Sensors, 22(23), 9074.
39. Wang, S., Li, D., Zhang, Y., & Chen, J. (2019). Smart contract-based product traceability system in the supply chain scenario. IEEE access, 7, 115122-115133.
40. Hasan, M. R., Alazab, A., Joy, S. B., Uddin, M. N., Uddin, M. A., Khraisat, A., ... & Talukder, M. A. (2023). Smart contract-based access control framework for internet of things devices. Computers, 12(11), 240.
41. Jifang Wang,Shangping Wang,Jin Sun,Xin Zhao & Bintao He. (2025). Blockchain assisted public audit with cross-authentication for shared data in Ad Hoc networks. Ad Hoc Networks,178,103960-103960.
42. Qiyuan Wang & Wentao Fan. (2025). Traceability of logistics electronic data based on mobile sensors and blockchain technology in the context of low-carbon energy. Results in Engineering,27,105806-105806.