IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received July 31, 2019, accepted August 8, 2019, date of publication August 16, 2019, date of current version August 30, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2935873*

# Smart Contract-Based Product Traceability System in the Supply Chain Scenario

**SHANGPING WANG**[1], **DONGYI LI**[1], **YALING ZHANG**[2], **AND JUANJUAN CHEN**[1]
[1]School of Science, Xi'an University of Technology, Xi'an 710048, China
[2]School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China

Corresponding author: Dongyi Li (dyli61610@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61572019, in part by the Key Research and Development Program of Shaanxi under Grant 2019GY-028, and in part by the Start-Up Fund for Ph.D. Teachers in Xi'an University of Technology under Grant 256081502.

**ABSTRACT** With the improvement of living standard, people begin to pay more attention to food safety and product quality. Therefore, for consumers, it is necessary to establish a reliable system that can trace the source of products. However, most existing traceability systems tend to lack transparency, data is primarily stored within the enterprise, and the cost of tampering with data is very low. Besides, the supply chain nodes are easy to evade responsibility when product safety or quality issues arise under the traditional centralized management model, and it is difficult to trace the root of issues. The development of blockchain technology provides us with new ideas for realizing the traceability of products in supply chain scenarios. Due to its characteristics of decentralization, transparency, and immutability, blockchain can be effectively used to alleviate the above problems. In this paper, we propose a product traceability system based on blockchain technology, in which all product transferring histories are perpetually recorded in a distributed ledger by using smart contracts and a chain is formed that can trace back to the source of the products. In particular, we design an event response mechanism to verify the identities of both parties of the transaction, so that the validity of the transaction can be guaranteed. And all events are permanently stored in the form of logs as a basis for handling disputes and tracking responsible entities. Furthermore, a system prototype is constructed based on the testing framework of Truffle. The contract code is deployed on a test network TestRpc that runs in local memory, and a decentralized web page interface is implemented based on the prototype. Finally, the system security analysis and experimental results show that our solution is feasible.

**INDEX TERMS** Blockchain, smart contract, supply chain, traceability, accountability.

## I. INTRODUCTION

The supply chain is a net-chain structure formed by independent or semi-independent economic entities in the process of product manufacturing and trading. It widely exists in many fields such as manufacturing industry, service industry, high-tech industry, and so on. The supply chain connects multiple entities such as suppliers, manufacturers, distributors, retailers, and customers. Complex supply chains can go through hundreds of stages, span months or even longer, and involve multiple regions around the world. In this scenario, it is difficult to find the root of the issue when the product has safety or quality problems because the supply chain involves a large number of entities. Especially in the food supply chain, ensuring food safety and traceability of sources can increase consumer trust. And for consumers, the government agencies or authorities need to respond more promptly and accurately to food scandals and accidents [1]. What's more, the quality, integrity, and availability of the product are critical for consumers, and product traceability ensures a high level of product [2]. At present, relevant departments and consumers in many countries advocate the traceability of the food supply chain, and have begun to improve the corresponding laws and regulations. For example, Canada enforces the use of bar codes, plastic hanging ear tags, or two electronic button ear tags to identify the initial herd. The National Livestock Identification System (NLIS) is Australia's permanent livestock identity system that tracks the entire process from birth to slaughter. It can be seen that the establishment and improvement of the traceability system is very necessary.

The supply chain is more emphasis on how to guarantee the long-term preservation and searchability of information,

The associate editor coordinating the review of this article and approving it for publication was Aniello Castiglione.

and ensure that the required history can be tracked in transactions that have already occurred. In the traditional supply chain system, however, the data is mostly recorded by each enterprise in a centralized ledger that is stored locally. When the information on the ledger is not conducive to the development of the enterprise itself, it may be falsified privately. As a result, mistrust between enterprises has become increasingly prominent, leading to an increase in communication costs. Moreover, due to the high possibility of tampering with data within the enterprise, the information between the supply chain nodes is inconsistent, which leads to the product traceability process being easily interrupted.

However, the emergence of Bitcoin [3] has brought a technical route to solve the above problems, and its underlying technology blockchain has been quickly recognized by all walks of life. A blockchain is a permanent, immutable record that is generated by superimposing encrypted data in chronological order. This distributed ledger technology has very significant features such as decentralization, traceability, tamper-proofing, and cryptographic security. Furthermore, smart contracts can be built based on blockchains [4], allowing transactions to be safely conducted between mutually untrusted parties. As an innovative Internet-based technology, blockchain can be used for a variety of applications such as tracking, auditing, and identity management, etc. The blockchain has come out of the back of Bitcoin and has begun to ferment into many other industries, for example, finance [5]–[9], electronic medical records (EMRs) [10]–[12], internet of things (IoT) [13]–[15], energy [16], [17] and many more. In particular, the application of blockchain technology in the supply chain industry will promote its more perfect development. As a distributed ledger database, the blockchain stores data in the form of blocks. The blocks are linked in chronological order by the Hash algorithm to ensure that the block contents cannot be changed. And each node in the network maintains a local copy of the blockchain ledger, which increases the cost of tampering with the data. This unique data storage method can effectively solve the problem of data can be easily modified in the supply chain. Therefore, the information of the upstream and downstream enterprises can be transparent and consistent, and all participating enterprises jointly maintain the ledger to prevent malicious node attacks.

**The main contributions of this paper are as follows.**

Firstly, to solve the problems that the product traceability process is easily interrupted and accountability is hard to achieve when disputes arise, in this paper, we propose a decentralized product traceability system based on blockchain technology. The process of product registration, transferring, and tracking is implemented through the collaboration of smart contracts. Our system breaks the situation that the information flow is maintained by a single or a small number of nodes in the traditional supply chain so that all nodes in the whole supply chain can participate in the process of maintaining the information flow. Since each node maintains a local copy of the entire network ledger, it dramatically reduces the possibility of privately tampering with data by enterprises.

Secondly, since the blockchain is inherently decentralized, the nodes in the network are mutually untrustworthy. So, how to ensure effective transactions between participants is very important. In the system proposed in this paper, we design an event response mechanism for KYC (know your customer) verification. And participants can determine the validity of the event by verifying the signature contained in the event to decide whether to continue trading with the other party. All events can be monitored and permanently stored in the blockchain, and regulators can implement the accountability function by querying the blockchain logs.

Finally, we build a decentralized application (DApp) based on the Truffle framework, deploy and test the contract code through a test network TestRpc running completely in local memory, and implement a decentralized web page interaction interface based on the prototype.

The rest of this paper is organized as follows. Section II states the related work, and Section III introduces the preliminaries used in this paper. In Section IV, the system architecture is described in detail. The concrete implementation of the smart contract is exhibited in Section V, and the experiment and evaluation are discussed in Section VI. Finally, the conclusion and future research direction are presented.

## II. RELATED WORK

With the advancement of science and technology and changes in market demand, understanding and mastering the research status and development trends of the supply chain plays a significant role in how to study the supply chain more deeply. Scholars and entrepreneurs at home and abroad have also made a lot of discussions and researches on how to combine the blockchain technology to improve the current management situation of the supply chain.

Information sharing accelerates product traceability process by enabling companies to understand useful information quickly. Nakasumi [18] proposed a new information sharing scheme based on blockchain technology. Users can manage their data and understand the data being collected about them and how to use it without trusting any third party. However, the scheme did not take into account the possibility of the enterprise itself tampering with data. Kim [19] *et al.* proposed a protocol for data interaction and sharing and described the integration details between supply chain nodes. The proposal of this scheme makes it possible to exchange and share data between upstream and downstream enterprises in the supply chain. Li *et al.* [20] proposed a peer-to-peer network architecture to support the increasing demand for visibility and timely delivery of information in the supply chain logistics phase, thus enabled the public to grasp the logistics information in real-time in a decentralized manner.

Data immutability can ensure the authenticity of the tracking data and eliminate the possibility of the company tampering with the data. Bocek *et al.* [21] proposed a modum.io model based on blockchain technology by using the Internet of Things sensor devices to ensure data invariance and public accessibility to temperature records while reducing operating costs for drug supply chain start-ups. As the model is in the preliminary stage of practice, the implementation details still need to be further improved. Ye *et al.* [22] designed and developed a supply chain prototype system based on blockchain, smart contracts, and Internet of things equipment, ensured that the data is transparent and tamper-proof, and provided different levels of data query functions for different users.

Tian [23] achieved traceability of trusted information throughout the agricultural product supply chain based on blockchain technology to ensure the collection, transfer, and sharing of real data and demonstrated the building process of the system. However, due to many uncertainties, it is necessary to improve and optimize the system itself in future research continuously. Lu and Xu [24] realized the product traceability process in a decentralized manner based on blockchain technology but did not consider the cost of integrating the blockchain into the existing system and the applicability of smart contracts. Malik *et al.* [25] proposed a permissioned blockchain framework managed by a food supply chain consortium, including government and regulatory agencies, used a three-tier architecture to ensure the availability of data to consumers. Salah *et al.* [26] proposed a solution that uses Ethereum blockchain and smart contracts to track and execute transactions without the need for a trusted third-party authority.

Also, Ye *et al.* [22] ensured that the supply chain data is transparent, traceable and immutable, while also taking into account the encryption protection of private data, enabling enterprises not to disclose sensitive information when sharing data. Lin *et al.* [27] designed a decentralized food safety tracking system based on blockchain and EPC information service network. The collaborative management of on-chain and off-chain data reduced the amount of data of a single node and alleviated data explosion problem. And enterprise-level smart contracts are used to replace traditional transaction records to preserve and manage food data to protect corporate privacy. In recent years, many enterprises at home and abroad are also actively exploring the application of blockchain in traceability and anti-counterfeiting, logistics, supply chain finance, and other scenarios. Blockchain technology is gradually penetrating traditional supply chain operations. For example, Maersk joint insurance institutions, blockchain companies, and other parties jointly created the world's first blockchain platform for marine insurance, forming a cross-professional chain alliance [28]. Wal-Mart Stores, the world's largest retailer, has completed a pilot program to track the whole process of pork production and sales using blockchain technology in China, resulting in a significant reduction in tracking time [29].

We also discussed other aspects of the application of blockchains in the supply chain to deepen the understanding of research topics. Chen *et al.* [30] discussed how to improve the quality management of the supply chain by using blockchain technology and proposed a quality management framework based on blockchain. Since the program is in the theoretical research stage, the actual implementation still needs a practical process. Leng *et al.* [31] proposed a public blockchain system based on the double-chain architecture in the agricultural supply chain. It mainly studied the double-chain structure and its storage method, resource rent-seeking matching mechanism, and consensus algorithm. Toyoda *et al.* [32] proposed a new product ownership management system for product anti-counterfeiting, and anyone can check the authenticity of the product. However, this solution does not address the issue of anonymity.

Since the research of blockchain applied to the supply chain field is still in the exploration stage, although there are many platforms to promote the supply chain information management based on blockchain technology, the theoretical research on the blockchain remains to be determined to improve. So far, blockchain technology still faces key challenges related to scalability, government governance, identity registration, user privacy, enforcement standards, and laws and regulations. This paper is based on the literature of the previous research, combined with the advantages and disadvantages of the scheme, proposed a decentralized product traceability system based on blockchain technology. Our design solves the problem in the current supply chain and finally achieves the product traceability process and information consistency among the supply chain nodes.

## III. PRELIMINARIES

In this section, we review some of the relevant background knowledge and detailed knowledge that will be used in this paper.

### A. BLOCKCHAIN

Blockchain is a new technology that is gradually emerging with the increasing popularity of digital currencies such as Bitcoin [3]. It is essentially a distributed ledger database. The blockchain records transactions that have been occurred by establishing a database maintained by all network nodes, and the entire process is open, transparent, and irreversible. It can be divided into the public blockchain, consortium blockchain, and private blockchain according to the participants, and the consortium blockchain and the private blockchain are collectively referred to as the permissioned blockchain. The public blockchain is completely decentralized in the real sense. That is, any node can join or exit the decision to create a new block at any time. However, in the permissioned blockchain, the decision to create a new block is performed by some trusted nodes and has been applied to copyright management, identity authentication, and data storage services.

### B. ETHEREUM AND SMART CONTRACT

Since Bitcoin is designed only for virtual currency scenarios, and it is not Turing-complete, the universality is not high. Hence, many other blockchain-based systems have emerged, and different types of applications have been allowed to be represented on the blockchain in the form of ''smart contracts''. Ethereum first realized the complete fit of blockchain and smart contracts [33]. Once a smart contract is deployed on the blockchain, it will be executed according to predefined rules, and no one can change it. The smart contract in Ethereum is written as a stack-based low-level bytecode language, which is performed by Ethereum Virtual Machine (EVM) and is called EVM code. These codes are stored in the blockchain in a binary form unique to Ethereum. Smart contracts can also be written in high-level languages, such as Solidity [34], and then compiled into EVM code.

The core of Ethereum is the Ethereum Virtual Machine, which can execute code with arbitrary algorithmic complexity. In computer terminology: Ethereum is Turing-complete. Developers can create contracts and a wide variety of DApps on Ethereum Virtual Machine based on existing programming languages (such as Java, Python). Smart contracts can handle a variety of operation logics, making full use of the Ethereum blockchain capabilities, enabling the blockchain more scalable and prompting Ethereum to develop into the most extensive blockchain development platform.

## IV. SYSTEM MODEL

In this section, we mainly introduce the system model of the product traceability process which is realized by the decentralization, data immutability, and time irreversibility of blockchain technology. Because the supply chain involves a large number of entities and the entire traceability process is very cumbersome, the system proposed in this paper considers dividing enterprise entities into suppliers, manufacturers, distributors and retailers to cover all the nodes involved in the supply chain. Any node can have both attributes of demand and supply. Therefore, by analyzing the net-chain structure generated in financial activities, we divide the supply chain nodes into different levels. By considering only the relationship between a node and its superior or subordinate in a single transaction, and then form a transaction chain consisting of transactions between nodes at various levels. Furthermore, in the process of product transaction formation, we design an event response mechanism to ensure that both parties of the transaction agree on the receipt and delivery of goods, and the process will be permanently stored in the blockchain.

### A. SYSTEM SECURITY REQUIREMENTS

The blockchain-based traceability system designed in this paper should meet the following security requirements.

Data Accessibility: As a design that supports the source traceable of the product, the system should be public to all users, and anyone can access the system to query product source data.

Data Immutability: In order to provide the public with true and reliable product source data, the system designed in this paper should have tamper-proof functionality, and no one can change the data.

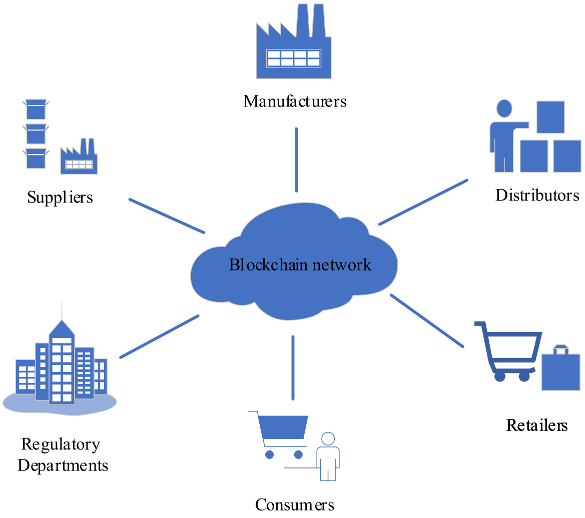

**FIGURE 1.** System overview.

System Autonomy: All data exchange in the system should follow a fixed and predictable algorithm. And all nodes can exchange, record, and update data in a trusted environment to prevent human interference.

Resistance to Man-in-the-middle Attacks: Since the blockchain itself is decentralized, the nodes in the system are mutually untrustworthy. Therefore, the system should resist man-in-the-middle attacks and prevent malicious nodes from forging transactions.

### B. SYSTEM OVERVIEW

As shown in Figure 1, the blockchain-based traceability system described in this paper mainly includes enterprise entities, consumers, and regulators in the supply chain, which are connected through a decentralized network. Each node in the network corresponds to an Ethereum account, which represents its identity in the system and can be used to deploy smart contracts. The enterprise entities in the supply chain can be summarized as suppliers, manufacturers, distributors, and retailers. And the functions and specific roles of each network node are described below.

**Suppliers**: The supplier is the owner of the raw materials and provides raw materials to the manufacturer. Raw materials as the source of products should be registered in the system by the supplier so that the consumer can trace back to the source of the raw materials of a specific product. Similar to the product, each raw material should also have a unique code.

**Manufacturers**: The manufacturer purchases raw materials from the supplier and processes the raw materials to produce the products. At the same time, the manufacturer supplies products to the distributor, which are sold all over the world by the distributor. As the owner of the product, the manufacturer is responsible for encapsulating the product information and registering it in the system. In the design of this paper, we use the international coding standard EAN/UCC-13 to encode the product, and the product has a unique code. We assume that in the process of product production, the products produced in the same batch are identical in quality and structure, and the difference can be ignored. Products that are produced in different batches need to be re-registered. Therefore, this design updates the product transfer process in batches, and each batch of products uniquely corresponds to a production batch number. For mass-produced products, they have the same batch number; for non-mass-produced products, the batch number represents the individual product.

**Distributors**: As the middleman in the process of transferring goods from the manufacturer to the end consumer, the main role of the distributor is to update the direction of the product flow in this process so that the product information remains uninterrupted during the traceability process.

**Retailers**: The retailer purchases products from the distributor in batches and sells them to consumers in retail, which is a participant that has direct trading relationships with consumers.

**Consumers**: The consumer is the individual who finally buys and uses the product. He can choose to participate in the network as a lightweight node, realizes the product traceability process by querying the data permanently stored in the block, or as a full node to jointly maintain the blockchain ledger data.

**Regulatory Departments**: Since the blockchain data is exposed to all nodes, it is highly transparent. Therefore, the regulatory department as a blockchain node can monitor product dynamics in real-time and can recall in time when safety or quality problems occur in the product and deal with issues quickly and efficiently.

To better show consumers the source traceability data of the product, when the product is transferred from one party to the other, we record the process by initiating a blockchain transaction. And add the product code and batch number to the transaction when the product is first transferred, and then in each subsequent transaction, by referring to the previous transaction hash to form a hash chain that traces the source of the product.

### C. SMART CONTRACT DESIGN

In order to realize the product traceability process, this paper designs three smart contracts, namely Product Registration Contract (PRC), Batch Addition Contract (BAC), and Transaction Update Contract (TUC). The PRC contract contains the addresses of the BAC contracts, and the BAC contract contains the addresses of the TUC contracts, so that the contracts can be linked and cooperated. For the convenience of expression, when we mention "product" in this subsection, the concept of the "product" includes both the product and the raw material. In the PRC contract, each product is assigned a BAC contract address when it is registered to add the production batch information of the product. In the BAC contract, each batch is assigned a TUC contract address to update the product transfer process of the batch. That is to say, each product corresponds to a batch list, and each batch corresponds to a product transfer process of this batch. The end consumer can query the transfer history of the purchased product by product code and batch number. Also, an authorization list is maintained in each contract, and only accounts added to the authorization list have the right to update the contract status. Once a malicious user occurs in the system, the regulator has the highest authority to remove the user from the authorization list. The specific functions of each contract are described below.

1) **Product Registration Contract (PRC)**. The PRC contract is deployed by the system manager and provides a product registration function `register()`, which permanently stores registration information for all products. The owner registers each product in the contract, including product code, product name, product owner, and raw materials or parts (see in Table 1). Moreover, when the product is registered, it will deploy a BAC contract at the new address using its owner's account and add the BAC contract address to the product registration information. The BAC contract provides a function to add production batches, which are detailed in the following. It should be noted that the process of registering raw materials is similar to this process and will not be repeated here.
2) **Batch Addition Contract (BAC)**. The BAC contract is deployed by the owner of each product when registering the product and provides the function `addBatch()` to add the production batch information of the product and saves all batches information permanently. The batch manager adds batch information to the contract for each batch of products, including the batch number, the batch manager, the timestamp, and the batch number of raw materials used to produce this batch of products, as well as deployed a TUC contract correspondingly (see in Table 2). The TUC contract provides the function of updating the product transaction record so that we can accurately grasp the nodes that the product has experienced in the supply chain, that is, which entities the product has gone through in the circulation process. The details are introduced in the following content.
3) **Transaction Update Contract (TUC)**. The TUC contract is deployed by the manager of each batch when adding the production batch information and provides the function `updateTr()` to update the transaction history for this batch of products. The following work is performed by the product sender after the product receiver receives the merchandise and launches a reception response event: i) First, create a transaction from the sender of the product to the recipient of the product, and attach relevant information to the transaction. That is, if the product is transferred for the first time, the product code and batch number are attached; if not, the hash of the previous transaction is referenced. ii) The transaction is then launched by the product sender and the relevant transaction information

**TABLE 1.** Example of product information list in PRC contract.

| id | Product Code | Product Name | Product Owner | Raw Materials | Timestamp | BAC Address |
|---|---|---|---|---|---|---|
| 1 | 109735813 | cocoa | 0xc07...f0e4 | / | 1562065155 | 0xbce0...d4c77 |
| 2 | 165456413 | milk | 0x32c...d192 | / | 1562065206 | 0x5695...9209c |
| 3 | 6928480334788 | coffee | 0x69a...0013 | 109735813, 165456413 | 1562065276 | 0xa911...0ac5a |
| 4 | 163113107 | tea | 0x38c...0d2a | / | 1562065411 | 0xa145...0e6d4 |
| ... | ... | ... | ... | ... | ... | ... |

**TABLE 2.** Example of the batch list in the BAC contract for a product with the product code of 6928480334788 in Table 1.

| id | Batch Number | Raw Materials and Their Batch Numbers | Batch Manager | Timestamp | TUC Address |
|---|---|---|---|---|---|
| 1 | 201907021605 | 109735813(201906281026),165456413(201906211139) | 0x69a...0013 | 1562622724 | 0xc246...1eebf |
| 2 | 201907031206 | 109735813(201906291536),165456413(201906211139) | 0x38c...0d2a | 1562722756 | 0xe1b2...d7059 |
| 3 | 201907031816 | 109735813(201906301538),165456413(201906211953) | 0x90f...c9c1 | 1562822789 | 0x8776...13dd4 |
| 4 | 201907041028 | 109735813(201906301538),165456413(201906221642) | 0xffc...09f0 | 1562922816 | 0x4994...cd779 |
| ... | ... | ... | ... | ... | ... |

**TABLE 3.** Example of product transaction list in the TUC contract corresponding to the batch with the batch number 201907021605 in Table 2.

| id | TrHash | Sender | Receiver | PreviousTr | Timestamp |
|---|---|---|---|---|---|
| 1 | 0xa6036...8f5718c | 0x3a0...c89b | 0xdde...08a0 | 6928480334788(201907021605) | 1563623747 |
| 2 | 0x0ade4...2193fe | 0xdde...08a0 | 0x2bb...9255 | 0xa6036...8f5718c | 1563823837 |
| 3 | 0x6c586...e409a5 | 0x2bb...9255 | 0xca7...c644 | 0x0ade4...2193fe | 1563623953 |
| 4 | 0xdfa0b...df2467 | 0xca7...c644 | 0xaa4...da63 | 0x6c586...e409a5 | 1563624067 |
| ... | ... | ... | ... | ... | ... |

is added to the list of transaction records maintained in the TUC contract, including the hash of the current transaction, the sender of the transaction, the recipient of the transaction, the hash of the previous transaction, and the timestamp (see in Table 3). It should be noted that the transaction list will only be updated if the current transaction is successfully packaged and chained to the blockchain and the hash of the previous transaction is valid; otherwise, an exception is thrown by the system. This work ensures that each transaction added to the blockchain is authentic and legitimate, preventing the product traceability process from being interrupted due to incorrect transaction information.

The three main methods mentioned above: `register()`, `addBatch()` and `updateTr()`, we will give the specific algorithm logic in the system implementation section.

### D. PRODUCT TRACEABLE FUNCTIONALITY

As demonstrated in Figure 2, the whole process of product registration, transferring, and tracking in the supply chain system is presented. We mainly summarize it as contract deployment, raw material registration, raw material procurement, product registration, product distribution, product wholesale, product purchase, and product source querying. Each step is described in detail as follows.

- Contract Deployment.

As shown in step 1 of Figure 2, the system manager deploys the PRC contract to the blockchain and public the contract address. The BAC contract and the TUC contract are automatically deployed by the caller's account when the `register()` function and the `addBatch()` function are invoked, respectively.

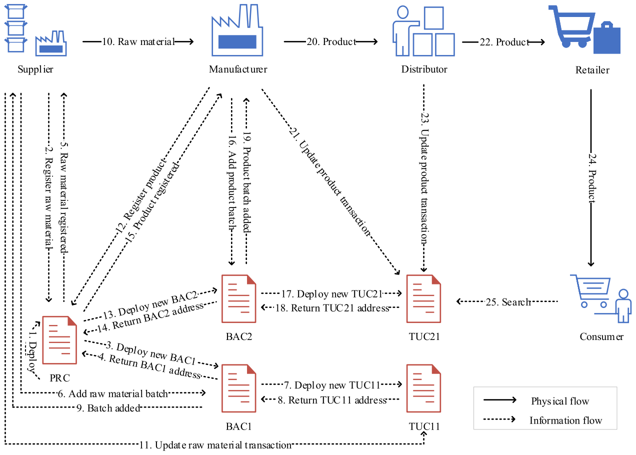

**FIGURE 2.** The process of product registration, transferring, and tracking.

- Raw Material Registration.

The raw material registration process is performed by the supplier, including the registration of raw material information and the registration of production batch information. As a type of product, raw materials have unique code, name, and other information which are similar to the product. First, the supplier invokes the `register()` function in the PRC contract to register the raw material information (Table 1) into the contract. At the same time, a BAC1 contract is automatically deployed by the supplier's account address, and the contract address is added to the registration information of the raw material, and then the registration result is returned to the supplier, as shown in step 2-5 of Figure 2. The supplier then registers the raw materials produced in batches and adds the batch information (Table 2) to the contract by invoking the

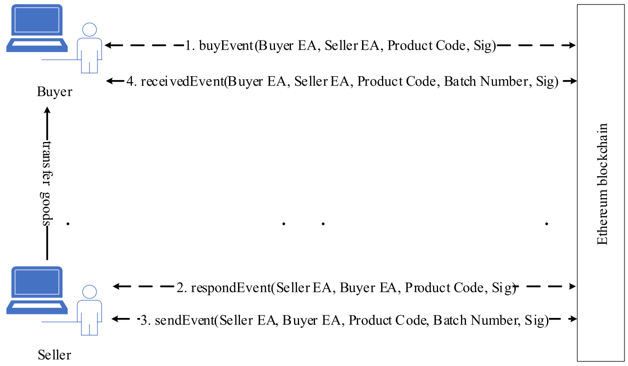
1. buyEvent(Buyer EA, Seller EA, Product Code, Sig)
4. receivedEvent(Buyer EA, Seller EA, Product Code, Batch Number, Sig)
Buyer
trans fer goods
Ethereum blockchain
2. respondEvent(Seller EA, Buyer EA, Product Code, Sig)
3. sendEvent(Seller EA, Buyer EA, Product Code, Batch Number, Sig)
Seller

**FIGURE 3.** **Event response mechanism.**

`addBatch()` function in the corresponding BAC1 contract. Meanwhile, a TUC11 contract is automatically deployed, and the contract address is added to the batch information of the raw material, and then the execution result is returned to the supplier, as shown in step 6-9 of Figure 2.

- Raw Material Procurement.

The raw material procurement process takes place between the supplier and the manufacturer. After both sides have reached a consensus, that is, the supplier confirms the delivery and the manufacturer confirms the receipt. The supplier will then upload the transaction information (Table 3) to the contract by invoking the `updateTr()` function in the TUC11 contract, as shown in step 11 of Figure 2.

As shown in Figure 3, the process of agreeing on the delivery and receipt of goods by both parties of the transaction is described, i.e., the actual process of step 10 in Figure 2. It is described below.

1) First, the buyer initiates a purchase request and the `buyEvent()` event is triggered, which includes the buyer's and seller's Ethereum address `Buyer EA` and `Seller EA`, the code of product to be purchased, `Product Code`, and the request initiator's Ethereum account public key `EPK`, as well as the signature `Sig` signed with its account private key to confirm the identity of both parties and the authenticity of the request. We specify the first Ethereum address in the event as the initiator's address and set the second Ethereum address in the event as an index entry so that other entities can query the log records related to themselves according to this entry and respond in time.
2) Then, the seller queries the log records related to himself according to his Ethereum address and verifies the validity of the signature contained in the events. If the verification is passed, an event `responseEvent()` is triggered by the seller to respond to the buyer's request.
3) Next, the seller transfers the goods to the buyer and an event `sendEvent()` is triggered to prove that the goods have been shipped, including the seller's and buyer's Ethereum address `Seller EA` and `Buyer EA`, the product code and the batch number of the shipped goods, `Product Code` and `Batch Number`, and the seller's Ethereum account public key `EPK`, as well as the signature `Sig` signed with its account private key.
4) Finally, an event `receivedEvent()` is triggered by the buyer upon receipt of the goods to certify that the goods have been received.

In the current scenario, the manufacturer acts as the buyer, and the supplier acts as the seller. Once the above process is completed, the supplier will update the transaction information according to the product code and batch number contained in the event to the corresponding TUC contract (i.e., TUC11 contract in Figure 2). We assume that both parties of the transaction will truthfully activate the above events because only then will the transaction information be updated, and the source of the product will be considered reliable.

- Product Registration.

The product registration process is performed by the manufacturer, including the registration of product information and the registration of batch information, and the corresponding deployment of BAC2 contract and TUC21 contract. Since this process is similar to the raw material registration process, it is not described here, as shown in steps 12-19 of Figure 2.

- Product Distribution.

The product distribution process takes place between the manufacturer and the distributor. After both parties reach a consensus according to the process shown in Figure 3, the manufacturer will upload the transaction information to the contract by invoking the `updateTr()` function in the TUC21 contract, as shown in steps 20 and 21 of Figure 2.

- Product Wholesale.

The wholesale product process takes place between the distributor and the retailer. After both parties reach a consensus according to the process shown in Figure 3, the distributor will upload the transaction information to the contract by invoking the `updateTr()` function in the TUC21 contract, as shown in steps 22 and 23 of Figure 2.

- Product Purchase.

The consumer purchases products from the retailer through supermarkets or hypermarkets. At this point, the products have arrived in the hands of the end consumer, as shown in step 24 of Figure 2. Because consumers are uncertain, we can't predict in advance which consumers will buy the product, so we don't consider uploading the transaction information to the blockchain in this step.

- Product Source Querying.

The consumer can choose to join the network as a lightweight node or a full node. All transaction histories of the purchased product can be inquired only by product code and batch number, as shown in step 25 of figure 2. Consumers view the source of the product by inputting product code and batch number and clicking the "OK" button on the page, and this operation does not consume gas.

**TABLE 4.** System test environment.

| | |
|---|---|
| Development Platform | Truffle v3.2.8 |
| Operating System | Windows 7 |
| Programming Language | Solidity |
| Editor | Atom 1.38.2 |
| Front-end Interaction | Web3.js API |
| Network Environment | EthereumJS TestRPC |

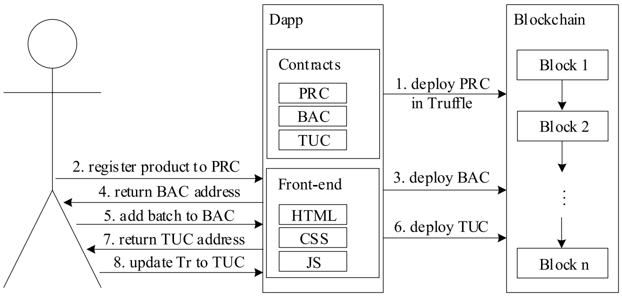

**FIGURE 4.** System construction flow diagram.

```
Algorithm 1: register()
Input: The massage sender's address(msg.sender),
       product code (productCode), product name
       (productName), raw materials
       (rawMaterials), current timestamp (now),
       BAC address (bacAddr), authorization list
       (AL), product count (productCount)
AL is the set of all authorized users' Ethereum address in
this contract;
if $msg.sender \in AL$ then
    if $productCode$ has not exist then
        register productCode, productName,
        msg.sender, rawMaterials, now, and
        bacAddr to the blockchain;
        productCount++;
    else
        Revert contract state and show an error.
    end
else
    Revert contract state and show an error.
end
```

## V. SYSTEM IMPLEMENTATION

In this section, the system proposed in this paper is implemented, and the compilation and deployment of smart contracts can be realized more easily by using the Truffle [35] framework. Truffle is a development tool for the Ethereum Solidity language and provides a test framework, which makes Dapp development easier. As shown in Table 4, the test environment for our solution is listed. Atom [36] is a powerful, free, and open-source text editor that can build a solidity IDE by downloading some plugins. Web3.js [37] is a JavaScript library provided by the Ethereum and offers a full set of JavaScript objects and functions that interact with the blockchain. In our implementation, we use the truffle default builder (truffle-default-builder 2.0.0) to help us integrate the contract abstraction into the script, so that we can use the contract abstraction provided by truffle directly in the JavaScript code.

The key to Dapp development lies in the writing of smart contracts and the interaction with contracts through front-end web pages. We mainly use HTML and CSS code to develop the web page, and then establish the connection with HTML code by writing front-end JavaScript code, to realize the interaction logic of the front-end web page. Users can interact with contracts by clicking on button events on the page, including deploying a new contract and invoke methods in the contract to write or read blockchain data. As depicted in Figure 4, the whole workflow of the system designed in this paper is shown. First, create a truffle project and write the project code in it. Smart contract code only works after it has been deployed on the blockchain. In system testing, we deploy only one PRC contract through truffle, and other contracts will be deployed automatically when front-end interactions are performed on web pages. Because this study case is used only for code testing, we use the local test network, TestRpc, to deploy contracts and send transactions. It is a network that runs entirely in local memory and exists only on the development device to quickly validate transactions and return execution results in real-time, without waiting for the default mining time like the main Ethereum network. Truffle takes advantage of this feature of the local test network and increases the test rate by nearly 90%. When developing Dapp, it is best to use TestRpc for code testing before deploying it on the main Ethereum network.

As we mentioned earlier, the product traceability system proposed in this paper mainly implements the following three smart contracts.

The PRC contract is equivalent to a platform that implements the registration of product information and displays all products that have been registered in the platform. The mapping relationship between the product code and the BAC contract address is maintained in the PRC contract. Manufacturers register their products on the platform so that other nodes can view the registered products through the platform. Algorithm 1 gives the pseudo-code implementation of the `register()` function.

BAC contract implements an update of the product batch, maintains a mapping relationship between the batch number and the TUC contract address, and can display all the batches that have been added to the contract. Each batch of product produced needs to be added to the contract so that other nodes can view all the product batches that have been produced for a product. Algorithm 2 gives the pseudo-code implementation of the `addBatch()` function.

TUC contract implements the product traceability process. Every transaction after the product leaves the manufacturers needs to be recorded to the contract. And the end consumer can view all transaction history of the product in

**Algorithm 2:** `addBatch()`

```
Input: The massage sender's address (msg.sender),
       batch number (batchNumber), raw materials
       used (materialBatchNumber), current
       timestamp (now), TUC address (tucAddr),
       authorization list (AL), batch count
       (batchCount)
AL is the set of all authorized users' Ethereum address in
this contract;
if msg.sender ∈ AL then
    if batchNumber has not exist then
        register batchNumber,
        materialBatchNumber, msg.sender,
        now, and tucAddr to the blockchain;
        productCount++;
    else
        Revert contract state and show an error.
    end
else
    Revert contract state and show an error.
end
```

**Algorithm 3:** `updateTr()`

```
Input: The massage sender's address (msg.sender),
       receiver's address (receiver), current
       timestamp (now), current tr (currentTr),
       previous tr (previousTr), authorization list
       (AL), tr count (trCount)
AL is the set of all authorized users' Ethereum address in
this contract;
if msg.sender ∈ AL then
    if previousTr has valid in the blockchain then
        register currentTr, msg.sender,
        receiver, previousTr, and now to the
        blockchain;
        productCount++;
    else
        Revert contract state and show an error.
    end
else
    Revert contract state and show an error.
end
```

**TABLE 5.** Smart contract deployment cost test(gasPrice = 2*Gwei*, 1*ether* = $296).

| Contract Create | Gas Used | Actual Cost(*ether*) | USD |
|---|---|---|---|
| Product Registration Contract | 1180217 | 0.002360434 | 0.6987 |
| Batch Addition Contract | 713513 | 0.001427026 | 0.4223 |
| Transaction Update Contract | 638125 | 0.001276250 | 0.3778 |

circulation. Algorithm 3 gives the pseudo-code implementation of `updateTr()` function.

## VI. EXPERIMENTAL EVALUATION

### A. EXPERIMENTAL RESULTS AND PERFORMANCE TEST

We analyze and evaluate the experimental results in this section. We uploaded all the experimental code to Github [38], and interested readers can access the link to download the code. Based on the front-end web page we implemented, the user can interact with the contract by clicking on the button event on the page, including the deployment and invocation of the contract. The gas cost required to deploy each smart contract is respectively listed in Table 5. The PRC contract only needs to be deployed once in the system, and the BAC contract and the TUC contract are deployed each time when the `register()` and `addBatch()` functions are invoked, respectively. The price of gas when we experimented was $2Gwei$ ($1Gwei = 10^9 wei = 10^{-9} ether$), and the price of Ether was $1ether = \$296$. The `register()`, `addBatch()`, and `updateTr()` operations were tested under different amount of raw materials, and the test results are shown in Figure 5. As seen in Figure 5, the different raw material quantities correspond to different gas costs. As the amount of raw materials increases, the costs of `register()` and `addBatch()` operations increases accordingly, and the cost of the `updateTr()` operation remains essentially unchanged. That means that the quantity of raw materials is independent of the cost of the `updateTr()` operation. The gas used per operation is obtained by using "web3.eth.getTransactionReceipt(TxHash).gasUsed" of the Web3.js API[39].

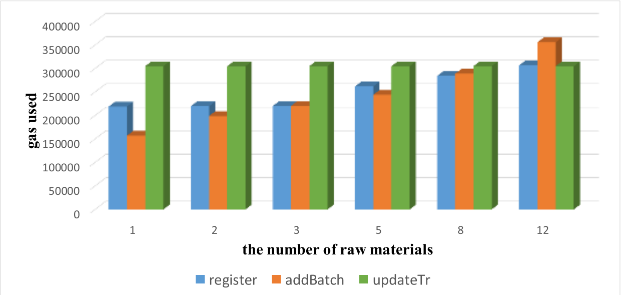

**FIGURE 5.** Smart contract operation costs under different number of raw materials.

As shown in Figure 6, a registration page for products and raw materials is displayed, listing all products and raw materials information already registered in the system. The enterprise can register the product into the blockchain to create a source for the product and generate an immutable record by entering the product name, product code, etc., and clicking the "Register Product" button. Meanwhile, a BAC contract was deployed at the new address by clicking the "Register Product" button. The product code is unique, using the international commodity coding standard EAN/UCC-13, consisting of 13 digits, and the last digit is the checksum. In order to distinguish raw materials from products, we use different raw material coding standards from products. The raw material code consists of 9 digits, the first position is "1",

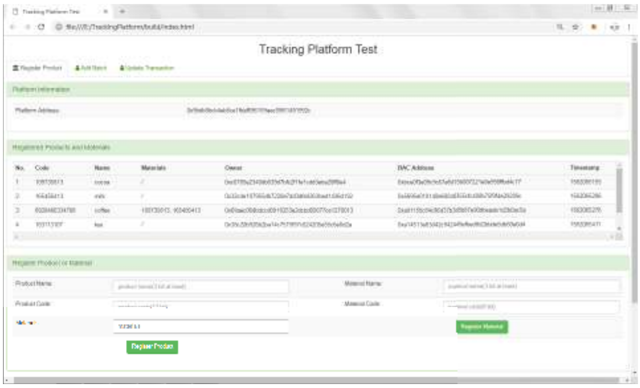

**FIGURE 6.** Product registration page.

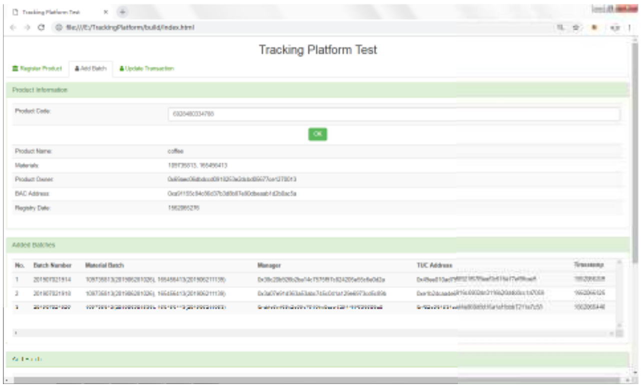

**FIGURE 7.** Product batch information page.

indicating that it is a raw material. Similar to the product code, the raw material code is unique. In addition, the page also provides the function to add users to the authorization list, and only authorized users can register products and raw materials.

Figure 7 shows the page for adding batches to a product or raw material. The user can query the registration information of the product by entering the product code and then clicking the ''OK'' button, and a list of all the batches that have been added is displayed on the page. The product batch manager, that is, the person responsible for the production of the batch, can add new batch information to the blockchain by entering the batch number and other items on the page. The batch number entered is unique and can identify this batch of products. Besides, the page also provides the function to add users to the authorization list only authorized users can add product batches.

As shown in figure 8, the page of the product transaction record update is displayed, and all the transferred histories of the product are listed, that is, which entities the product has gone through, and finally arrived in the hands of the end consumer. The authorized user can initiate a product transfer transaction by entering the referenced previous transaction hash and the product recipient's account address, and the transaction record will be added to the transaction list after the transaction is successfully packaged into the block. Consumers can query all transaction histories of the purchased product by inputting product code and batch number and clicking the ''OK'' button on the page, and the source of the product can be uniquely determined by the product code and batch number.

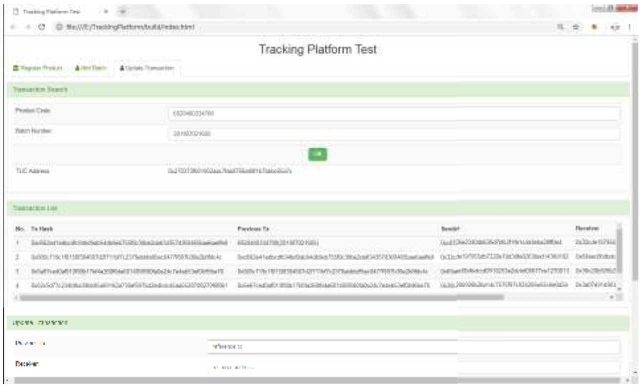

**FIGURE 8.** Product transaction update page.

**TABLE 6.** Comparing with other schemes.

| | [23] | [25] | [26] | [27] | Our System |
|---|---|---|---|---|---|
| Traceability | Yes | Yes | Yes | Yes | Yes |
| Degree of the Decentralization | High | Low | High | High | High |
| Accountability | No | Yes | No | Yes | Yes |
| Scalability | No | No | Yes | Yes | Yes |
| User-friendliness | No | Yes | No | No | Yes |

### B. SECURITY ANALYSIS

The blockchain-based traceability system designed in this paper meets the following security requirements.

Data Accessibility: In the design of this paper, the system user can choose to join the network as a full node or a lightweight node, and any node can query the data stored in the blockchain.

Data Immutability: Since the blockchain itself has the characteristics of tamper-proofing, and the system proposed in this paper is based on blockchain technology. Therefore, we say that our system data is also immutability.

System Autonomy: Through the smart contract deployed by the system manager on the platform, all information activities such as data exchange and storage generated on the blockchain can be considered as fixed and predictable, which ensures the accuracy, security, and autonomy of information exchange in a trusted environment.

Resistance to Man-in-the-middle Attacks: In this solution, we design an event response mechanism to ensure the identity of both parties of the transaction and the authenticity of the event. It is possible to resist man-in-the-middle attacks by verifying the signature carried in the event to determine whether the event is valid or not.

### C. SCHEME COMPARISON

We compare the traceability system designed in this paper with other systems. The comparison results are shown in Table 6, and the specific analyses are as follows.

- Traceability: Since we are comparing these traceability systems, these schemes all have the characteristics of traceability.
- Degree of the Decentralization: We introduce the concept of blockchain into the product source traceable system of the supply chain to eliminate the possibility of privately tampering with data within enterprises. Also, consumers can participate in the system to jointly maintain blockchain ledger data public and transparent. In comparison with other schemes, the framework proposed in [25] has the characteristics of partial decentralization, because the consumer acts as an off-chain participant, does not have the authority to maintain a copy of the ledger. Therefore, the degree of decentralization is a little lower than other schemes.
- Accountability: Although the absence of central authority can achieve complete decentralized storage of data information generated in the supply chain, no one can act as an authority to handle disputes and propose punishment when problems arise. Therefore, government agencies and regulators should also participate as nodes in the network to solve the above problems [25], [27]. They can verify the transaction information and locate the responsible person by querying the blockchain log, thus making the proof of accountability easier.
- System Scalability: For the platform framework proposed in this paper, we designed and retained enough extension points, and subsequent developments can easily add various functions. For example, add review functionality.
- User-friendliness: The web page we developed can easily realize the interaction between the user and the blockchain network and can attract more users in the form of web pages.

## VII. CONCLUSION

In this paper, a product traceability system based on blockchain technology is proposed, in which all product transferring histories are perpetually recorded in the unchangeable ledger by using smart contracts, and the process of product registration, transferring and tracking is realized through the collaboration of smart contracts. Consumers can also join the network as full nodes or lightweight nodes so that all nodes in the whole supply chain can participate in the process of maintaining information flows. Our system has obvious decentralized characteristics, which significantly reduces the possibility of privately tampering with data within enterprises. Moreover, we design an event response mechanism to verify the identity of both parties of the transaction, and by verifying the signature contained in the event to determine whether the event is valid. All events can be listened to and permanently stored in the blockchain in the form of a log. Finally, we build a decentralized application (DApp) based on the truffle framework, deploy the smart contract and test the contract code through a test network TestRpc running completely in local memory, and implement a decentralized web page interaction interface based on the prototype. The results of the security analysis show that our system is characterized by data accessibility, tamper-proofing, and resistance to man-in-the-middle attacks.

To optimize the traceability system proposed in this paper is our future research work: 1) Realize formatted upload of data by using IoT technology, reduce the possibility of manual input errors. 2) Through QR code technology to promote the process of product source querying, improve consumer consumption experience, and simplify the consumer operation process.

## REFERENCES

[1] M. M. Aung and Y. S. Chang, “Traceability in a food supply chain: Safety and quality perspectives,” *Food Control*, vol. 39, pp. 172–184, May 2014.

[2] M. A. Benatia, A. Remadna, D. Baudry, P. Halftermeyer, and H. Delalin, “QR-code enabled product traceability system: A big data perspective,” in *Proc. 16th Int. Conf. Manuf. Res.(ICMR)*, Skövde, Sweden: Univ. Skövde, Sep. 2018, pp. 323–328.

[3] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[4] *A Next-Generation Smart Contract Decentralized Application Platform*. Accessed: Jun. 28, 2019. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper/784a271b596e7fe4e047a2a585b733d631fcf1d4

[5] K. Fanning and D. P. Centers, “Blockchain and its coming impact on financial services,” *J. Corporate Accounting Finance*, vol. 27, no. 5, pp. 53–57, Jul. 2016.

[6] W.-T. Tsai, R. Blower, Y. Zhu, and L. Yu, “A system view of financial blockchains,” in *Proc. IEEE Symp. Service-Oriented System Eng. (SOSE)*, Oxford, U,K., Mar./Apr. 2016, pp. 450–457.

[7] B. Egelund-Müller, M. Elsman, F. Henglein, and O. Ross, “Automated execution of financial contracts on blockchains,” *Bus. Inf. Syst. Eng.*, vol. 59, no. 6, pp. 457–467, Dec. 2017.

[8] Q. K. Nguyen, “Blockchain—A financial technology for future sustainable development,” in *Proc. 3rd Int. Conf. Green Technol. Sustain. Develop. (GTSD)*, Kaohsiung, Taiwan, Nov. 2016, pp. 51–54.

[9] S. Singh and N. Singh, “Blockchain: Future of financial and cyber security,” in *Proc. 2nd Int. Conf. Contemp. Comput. Inform. (IC3I)*, Noida, India, Dec. 2016, pp. 463–467.

[10] A. Dubovitskaya, Z. Xu, S. Ryu, M. Schumacher, and F. Wang, “Secure and trustable electronic medical records sharing using blockchain,” in *Proc. AMIA Annu. Symp. Proc.*, pp. 650–659, Jan. 2017.

[11] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, “MedRec: Using blockchain for medical data access and permission management,” in *Proc. 2nd Int. Conf. Open Big Data (OBD)*, Vienna, Austria, Aug. 2016, pp. 25–30.

[12] K. Fan, S. Wang, Y. Ren, H. Li, and Y. Yang, “MedBlock: Efficient and secure medical data sharing via blockchain,” *J. Med. Syst.*, vol. 42, no. 8, p. 136, Aug. 2018.

[13] K. Christidis and M. Devetsikiotis, “Blockchains and smart contracts for the Internet of Things,” *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[14] Y. Zhang and J. Wen, “The IoT electric business model: Using blockchain technology for the Internet of Things,” *Peer-to-Peer Netw. Appl.*, vol. 10, no. 4, pp. 983–994, Jul. 2017.

[15] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, “Smart contract-based access control for the Internet of Things,” *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1594–1605, Apr. 2019.

[16] N. Z. Aitzhan and D. Svetinovic, “Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams,” *IEEE Trans. Depend. Sec. Comput.*, vol. 15, no. 5, pp. 840–852, Sep./Oct. 2018.

[17] C. Pop, T. Cioara, M. Antal, L. Anghel, L. Salomie, and M. Bertoncini, “Blockchain based decentralized management of demand response programs in smart energy grids,” *Sensors*, vol. 18, no. 1, p. 162, Jan. 2018.

[18] M. Nakasumi, “Information sharing for supply chain management based on block chain technology,” in *Proc. IEEE 19th Conf. Bus. Informat. (CBI)*, Thessaloniki, Greece, Jul. 2017, pp. 140–149.

[19] M. Kim, B. Hilton, Z. Burks, and J. Reyes, "Integrating blockchain, smart contract-tokens, and IoT to design a food traceability solution," in *Proc. 9th IEEE Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, Vancouver, BC, Canada: Univ. British Columbia, Nov. 2018, pp. 335–340.

[20] Z. Li, H. Wu, B. King, Z. B. Miled, J. Wassick, and J. Tazelaar, "A hybrid blockchain ledger for supply chain visibility," in *Proc. 17th Int. Symp. Parallel Distrib. Comput. (ISPDC)*, Geneva, Switzerland, Aug. 2018, pp. 118–125.

[21] T. Bocek, B. B. Rodrigues, T. Strasser, and B. Stiller, "Blockchains everywhere—A use-case of blockchains in the pharma supply-chain," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, Lisbon, Portugal, May 2017, pp. 772–777.

[22] X. Ye, Q. Shao, and R. Xiao, "A supply chain prototype system based on blockchain, smart contract and Internet of Things," *Sci. Technol. Rev.*, vol. 35, no. 23, pp. 62–69, 2017.

[23] F. Tian, "An agri-food supply chain traceability system for China based on RFID & blockchain technology," in *Proc. 13th Int. Conf. Service Syst. Service Manage. (ICSSSM)*, Kunming, China, Jun. 2016, pp. 1–6.

[24] Q. Lu and X. Xu, "Adaptable blockchain-based systems: A case study for product traceability," *IEEE Softw.*, vol. 34, no. 6, pp. 21–27, Nov./Dec. 2017.

[25] S. Malik, S. S. Kanhere, and R. Jurdak, "ProductChain: Scalable blockchain framework to support provenance in supply chains," in *Proc. IEEE 17th Int. Symp. Netw. Comput. Appl. (NCA)*, Cambridge, MA, USA, Nov. 2018, pp. 1–10.

[26] K. Salah, N. Nizamuddin, R. Jayaraman, and M. Omar, "Blockchain-based soybean traceability in agricultural supply chain," *IEEE Access*, vol. 7, pp. 73295–73305, 2019.

[27] Q. Lin, H. Wang, X. Pei, and J. Wang, "Food safety traceability system based on blockchain and EPCIS," *IEEE Access*, vol. 7, pp. 20698–20707, Feb. 2019.

[28] P. Baipai. (2017). *How IBM and Maersk Will Use the Blockchain to Change the Shipping Industry*. [Online]. Available: http://www.nasdaq.com/article/how-ibm-and-maersk-will-use-the-blockchain-to-change-the-shipping-industry-cm756797

[29] J. Nation. (2017). *Walmart Tests Food Safety With Blockchain Traceability*. [Online]. Available: https://www.ethnews.com/walmart-tests-food-safety-with-blockchain-traceability

[30] S. Chen, R. Shi, Z. Ren, J. Yan, Y. Shi, and J. Zhang, "A blockchain-based supply chain quality management framework," in *Proc. IEEE 14th Int. Conf. e-Bus. Eng. (ICEBE)*, Shanghai, China, Nov. 2017, pp. 172–176.

[31] K. Leng, Y. Bi, L. Jing, H.-C. Fu, and I. Van Nieuwenhuyse, "Research on agricultural supply chain system with double chain architecture based on blockchain technology," *Future Gener. Comput. Syst.*, vol. 86, pp. 641–649, Sep. 2018.

[32] K. Toyod, P. T. Mathiopoulo, I. Sasase, and T. Ohtsuk, "A novel blockchain-based product ownership management system (POMS) for anti-counterfeits in the post supply chain," *IEEE Access*, vol. 5, pp. 17465–17477, 2017.

[33] G. Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Yellow Paper. Accessed: Jun. 28, 2019. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf

[34] *Solidity—A Contract-Oriented, High-Level Language for Implementing Smart Contract*. Accessed: Jun. 28, 2019. [Online]. Available: https://solidity.readthedocs.io/en/develop/

[35] *Truffle Suite*. Accessed: Jun. 28, 2019. [Online]. Available: https://www.trufflesuite.com/truffle

[36] *Atom—A Hackable Text Editor for 21St Century*. Accessed: Jun. 28, 2019. [Online]. Available: https:/atom.io/

[37] *web3.js—Ethereum JavaScript API*. Accessed: Jun. 28, 2019. [Online]. Available: https://web3js.readthedocs.io/en/1.0/

[38] *Tracking Platform Test*. Accessed: Jul. 18, 2019. [Online]. Available: https://github.com/snowby-ldy/eth-traceabilityplatform

[39] J.-S. Park, T.-Y. Youn, H.-B. Kim, K.-H. Rhee, and S.-U. Shin, "Smart contract-based review system for an IoT data marketplace," *Sensors*, vol. 18, no. 10, p. 3577, Oct. 2018.

**SHANGPING WANG** received the B.S. degree in mathematics from the Xi'an University of Technology, Xi'an, China, in 1982, the M.S. degree in applied mathematics from Xi'an Jiaotong University, Xi'an, in 1989, and the Ph.D. degree in cryptology from Xidian University, Xi'an. He is currently a Professor with the Xi'an University of Technology. His current research interests include cryptography and information security.

**DONGYI LI** is currently pursuing the M.S. degree in applied mathematics with the School of Science, Xi'an University of Technology, Xi'an. Her research interests include information security and blockchain technology.

**YALING ZHANG** received the B.S. degree in computer science from Northwest University, Xi'an, China, in 1988, and the M.S. degree in computer science and the Ph.D. degree in mechanism electron engineering from the Xi'an University of Technology, Xi'an, in 2001 and 2008, respectively, where she is currently a Professor. Her current research interests include cryptography and network security.

**JUANJUAN CHEN** received the Ph.D. degree from Shaanxi Normal University, Xi'an, Shaanxi, China, in 2014. She is currently a Lecturer with the Xi'an University of Technology. Her main research interest includes cryptography.