IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON MISSION CRITICAL SENSORS AND SENSOR NETWORKS (MC-SSN)

Received November 14, 2019, accepted November 28, 2019, date of publication December 16, 2019,
date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2959771

Smart Contract-Based Data Commodity Transactions for Industrial Internet of Things

YUNA JIANG, (Student Member, IEEE), YI ZHONG, (Member, IEEE), AND XIAOHU GE, (Senior Member, IEEE)

School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China

Corresponding author: Xiaohu Ge (xhge@hust.edu.cn)

This work was supported by the National Key Research and Development Program of China under Grant 2017YFE0121600.

ABSTRACT The evolution of Industrial Internet of things (IIoT) boosts the amount of IIoT data. Machine learning promotes the progress of data analytics services. In order to facilitate the flow and explore the economic value of IIoT data, it is crucial to consider data packet transactions (DPTs) and data analytics service transactions (DASTs) simultaneously. Centralized data trading platforms emerge to realize transactions of data commodities. However, centralized platforms lack trust and robustness. How to realize DPTs and DASTs in a decentralized way is a challenging issue. In this paper, a new transaction solution based on the smart contract-enabled blockchain technology is proposed, which consists of the DPT smart contract and DAST smart contract. The DPT smart contract is implemented to trade data packets. The DAST smart contract provides a competitive way to trade data analytics services. Both smart contracts are designed to enable entities in IIoT to execute DPTs and DASTs automatically and honestly. Moreover, the transaction disputes between different IIoT entities are solved by the big data center off-chain, and the treatment results will be recorded on the blockchain by the big data center. The DPT smart contract and DAST smart contract are implemented and tested on Remix integrated development environment to achieve DPTs and DASTs. The gas costs of smart contracts are estimated and the security of the proposed solution is analyzed. The performance analysis demonstrates that the proposed solution is secure and feasible.

INDEX TERMS Industrial Internet of Things, data packet transaction, data analytics service transaction, smart contract, blockchain.

I. INTRODUCTION

Industrial Internet of things (IIoT) has attracted extensive attention in recent years. IIoT focuses on connections of various kinds of smart industrial devices, including manufacturing, agriculture, petroleum, medical system, transportation, etc [1]–[3]. The development of fifth generation wireless communication networks and IIoT boosts the amount of industrial data [4], [5]. Moreover, with the development of the machine learning and data mining technologies, data analytics services become popular to explore the potential value of industrial data [6]. In order to utilize industrial data to implement the intelligent manufacturing and meet the various requirements of traffic [7], an efficient big data market should be designed to allow different IIoT entities to trade data commodities, including data packets and data analytics services in a secure way [8]. Both data packets and data analytics services are precious commodities among industry circles, as data analytics services based on collected data packets have brought economic profits to various kinds of industries [9]. The data transaction mode relying on third-party platforms becomes the mainstream. However, third-party centralized data transaction platforms have problems of trust and robustness. For one thing, enterprises have security concerns about whether big data platforms would precipitate data or the data stored on big data platforms would be brute-forced. For another thing, the denial of service and single point of failure are common problems in the existing centralized systems. The service quality and system security might be affected if the centralized server is attacked [10]. How to design a decentralized data packet transaction (DPT) and data analytics service transaction (DAST) solution in IIoT is a challenging issue.

There has been an increasing interest in the blockchain technology [11]. The earliest application of blockchain technology is Bitcoin [12]. However, the design of Bitcoin only considers the application of digital currency, which cannot support many business applications. As Bitcoin and

The associate editor coordinating the review of this manuscript and approving it for publication was Yonghui Li.

blockchain technologies gradually attract attentions, some cryptocurrencies based on the blockchain technology have emerged. Ethereum [13], proposed by Vitalik Buterin, has become the most popular blockchain application platform at present. The blockchain technology is considered as the main candidate technology for the decentralization of Internet of Things (IoT) [14]. Transaction data stored on the blockchain needs to be maintained by the whole network, so data commodities can be transferred between untrusted nodes. Applications that previously could only be implemented on trusted third-party platforms can be performed in a distributed manner with the blockchain technology at present [15]. The blockchain technology makes the market highly decentralized, and blockchain-based markets are trusted, robust, and secure [16]. Nick Szabo first proposed the concept of smart contracts, which means that legal provisions can be written in executable code [17]. Vitalik Buterin introduced smart contracts into Ethereum, indicating that procedure on Ethereum could be implemented automatically and could not be interfered with [13]. A smart contract is a modular, reusable and automated script that runs on the blockchain. Smart contracts are stored on the blockchain, so each node can call functions in smart contracts and view the log of each interaction recorded on the blockchain [18]. The distributed consensus and tamper-proof characteristics of blockchain technology can build trust between different IIoT entities, which promotes IIoT entities to send transactions. Moreover, different IIoT entities can trade data packets or data analytics services automatically through smart contracts. The smart contract-enabled blockchain technology is expected to be a key technology to build a decentralized big data market.

In order to facilitate the flow of data and explore the potential value of data in IIoT. A transaction solution containing DPTs and DASTs is proposed based on the blockchain technology and smart contract in this paper. IIoT entities execute transactions through smart contracts stored on the blockchain without the needs of centralized systems. The main contributions of this paper are summarized as follows:

1) Based on the smart contract-enabled blockchain technology, a new transaction solution consisting of DPTs and DASTs in IIoT is proposed.
2) The DPT smart contract is designed to trade data packets while the DAST smart contract provides a competitive way to trade data analytics services. Transaction disputes between different IIoT entities can be solved by the big data center (BDC) off-chain and treatment results would be recorded on the blockchain by the BDC.
3) The DPT smart contract and DAST smart contract are written in the Solidity language and implemented in Remix integrated development environment (IDE). The gas costs of smart contracts are estimated, and the security of the proposed solution is analyzed.

This paper is organized as follows. A review of related work is presented in Section II. The system framework is described in Section III. Smart contracts are designed in Section IV. Implementation and testing results are given in Section V. Finally, conclusions are drawn in Section VI.

II. RELATED WORK

In this section, some related works about DASTs and DPTs are reviewed and discussed. Transactions of data analytics services were studied in [9], [19]–[21]. In [9], data analytics services are considered as digital items. The optimal pricing and profit maximization model based on Bayesian digital commodity auction were proposed. A smart data pricing method for IoT providers was proposed in [19], which determined the purchase price of data packets for the data owner and the subscription fee for the service user. A profit-driven data acquisition framework for the crowdsourcing-aware data trading market was proposed in [20]. In [21], a business model of the service platform between the wireless sensor network and users was proposed. The service platform provides data analytics services to users based on data purchased from the wireless sensor network.

There exist some studies investigating data packet transactions or sharing based on the blockchain technology [22]–[27]. An electronic medical data sharing framework based on the blockchain technology, smart contract and interplanetary file system (IPFS) was proposed in [22], which implemented a trusted access control scheme. Electronic medical data can be shared between different patients and healthcare providers in a secure way. A distributed data storage and sharing framework was proposed based on the blockchain technology, attribute-based encryption (ABE) technology and IPFS distributed storage [23]. The proposed framework executed the function of the ciphertext keyword search, which solved the problem that the cloud server could not return all search results or return incorrect results in the traditional cloud storage system. A method of trading IoT data without third-party verification was proposed in [24], which analyzed the interaction and authentication between entities involved in data transactions. A secure data storage and sharing scheme for vehicle edge networks based on the consortium blockchain and smart contract was proposed in [25], which could prevent unauthorized data sharing effectively. Consider the lack of trust, security and transparency in existing digital asset transfer systems, a distributed digital asset delivery proof was proposed in [26], the blockchain technology and smart contract are used to provide immutable, tamper-proof transaction logs. A distributed data vending framework based on data embedding and similarity learning is proposed in [27], which was analyzed by a real case for sharing electronic medical records. Some open-source projects started researching data packet transactions. AAAChain aims to build a decentralized and autonomous data open platform that is composed of a large number of vertical scene applications by using the blockchain technology [28]. GXChain is a fundamental chain serving the

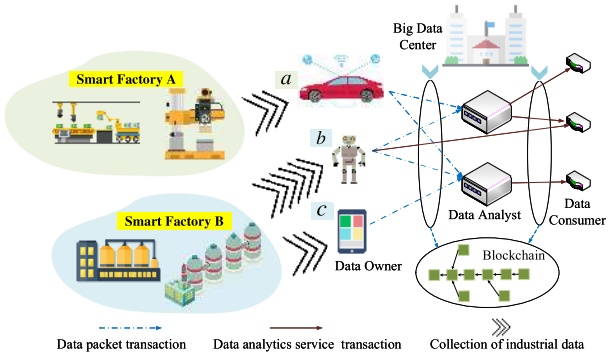


FIGURE 1. System framework.

global data economy, which aims to build a network of value for trusted data [29]. DXChain is a big decentralized data and machine learning network driven by a computing-centric blockchain [30]. XChain is committed to constructing a distributed data trading platform that enables data of small and medium-sized enterprises to be circulated in an effective and legal way [31]. However, how to design a decentralized trade market for DPTs and DASTs based on the smart contract-enabled blockchain technology has not been studied.

III. SYSTEM FRAMEWORK

In this section, a system framework for DPTs and DASTs among various entities is proposed. The proposed data transaction system framework consists of four entities: data owner (DO), data analyst (DA), data consumer (DC) and big data center (BDC). All participating entities have Ethereum addresses and can send transactions by calling functions defined in smart contracts. All legal transactions will be recorded on the blockchain. The system framework is shown in Figure 1. The main components of the proposed transaction system framework are presented as follows.

Data Owner: DOs are entities that can gather data from smart factories. The DO can be an intelligent industrial robot or a smart handheld terminal, etc. When the DO wants to publish data packets that are intended to be sold, the DO needs to call the corresponding function to add the information of data packets into the distributed ledger of blockchain. Then the information of data packets will be recorded on the blockchain. All entities joining in the blockchain can see the information of the data packets that DO wants to sell. Data packets are desensitized by the DO. Different DOs are possessed by different smart factories. As shown in Fig.1, the DO *a* is controlled by the smart factory A while the DO *b* and DO *c* are controlled by the smart factory B.

Data Analyst: DAs are smart entities that are equipped with data modeling and analysis capabilities, such as prediction and verification accuracy [9]. DAs can use machine learning algorithms to analyze data packets bought from DOs. When DCs send transactions on the blockchain to request data services, including data packets that DAs need to buy from DOs and the required analysis results, DAs make decisions on whether to provide data analytics services for DCs. If the DA is willing to provide the data analytics service, the DA will publish the major service information on the blockchain. There is a competition between DAs, as only the DA who provides the data analytics service with the minimum price will be chosen by the DC. Meanwhile, the service time of the DA needs to meet DC's requirement. The DA that is finally selected by the DC can provide the data analytics service based on the requirements of DC.

Data consumer: DCs are entities that need either data packets or data analytics services. On the one hand, DCs can buy data packets from DOs directly if data analytics services are not required. On the other hand, DCs can request data analytics services from DAs, and the data analytics service requirements of DCs will be responded by DAs. The DC chooses one DA from DAs who are willing to provide the data analytics service. The selected DA buys data packets from the DO based on the requirement of DC and provides the requested data analytics service at the minimum price. Finally, the DC buys the data analytics results from the selected DA by calling functions in the smart contract.

Big data center: The BDC is responsible for the supervision of DPTs and DASTs on the blockchain. Duties of the BDC are as follows:

1) The BDC creates smart contracts that consist of transaction rules and penalty rules. The BDC can also destruct and recreate the smart contract when the security flaw is found. DOs, DAs and DCs can send transactions based on rules regulated in smart contracts.
2) The BDC needs to check the data packet information given by DOs. If the data packets published by the DO have already existed, the DO will be punished automatically based on penalty rules in smart contracts.
3) The BDC is responsible for solving DPT disputes and DAST disputes off-chain. Results of disputes will be recorded on the blockchain by the BDC, and the entities with malicious behaviors will be punished automatically based on the penalty rules of smart contracts.

Blockchain: The blockchain and smart contract are used to establish the decentralized data trading market between different entities. By designing smart contracts, DPTs and DASTs can be performed automatically without the involvement of a trusted-third-party. Moreover, DPTs and DASTs can be recorded on the blockchain, which can guarantee the non-tampering of transactions in a trustless environment.

IV. SMART CONTRACT DESIGN

The smart contract consisting of functions and data is a piece of code that resides at a specific address on the blockchain [32]. The behavior of smart contracts is controlled by the contract code. Since the contract code can be seen and executed by all consensus nodes in the blockchain network, the smart contracts are trustworthy and can work exactly based on preset rules [33]. Besides, all execution records are kept on the blockchain and cannot be tampered [34].

In this section, two smart contracts are designed. As shown in Fig.2, the BDC creates two smart contracts including DPT

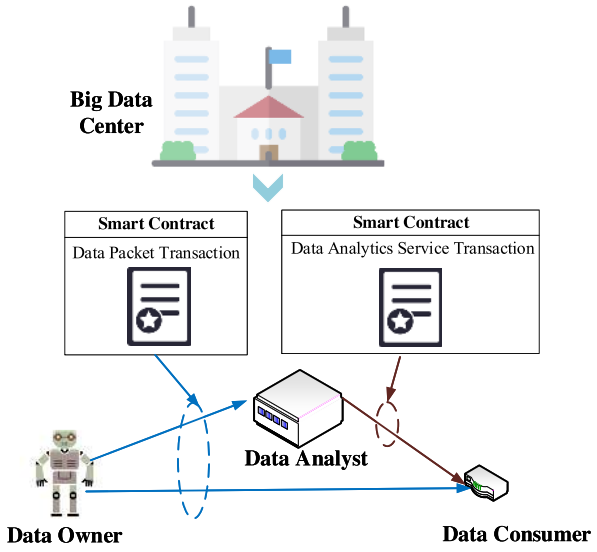


FIGURE 2. Smart contracts.

smart contract and DAST smart contract. DAs and DCs buy data packets from DOs through the DPT smart contract, and DCs buy data analytics services from DAs through the DAST smart contract. Smart contracts mainly contain the following parts:1) Function: A function is a unit of code executable in the smart contract. All participating entities can perform transactions by calling functions in smart contracts. There are functions that allowing DAs to request data packets, DOs to sell data packets, etc; 2) Function modifier: The function modifier can change the behavior of functions to some extent. For example, the function modifier can automatically check whether a parameter is valid before the execution of functions; 3) Event: Events can notify all participating entities about the updates of transactions taking place. State changes of smart contracts can be obtained externally through events.

A. DPT SMART CONTRACT

In this section, the DPT smart contract is described. DAs and DCs buy data packets from DOs by calling functions in the DPT smart contract. The main functions in the DPT smart contract are shown below.

1) PUBLISHING DATA PACKETS FOR SALE

The procedure of DOs publishing data packets through the DPT smart contract is shown in Figure 3. Two functions including *AddDataItem()* and *DataCenterCensor()* are designed to describe how DOs publish data packets for sale.

AddDataItem(): The DO calls *AddDataItem()* when the DO wants to publish data packets that are intended to be sold. The DO needs to publish the information of the data packet on the blockchain, which is denoted as *DataItem(ItemHash,TopicName,ItemPrice, ItemSequence)*. *ItemHash* is the Hash value of the data packet that is computed by the DO, *TopicName* is the name of the data packet, *ItemPrice* is the price of the data packet and *ItemSequence* is the number of data packets that have already

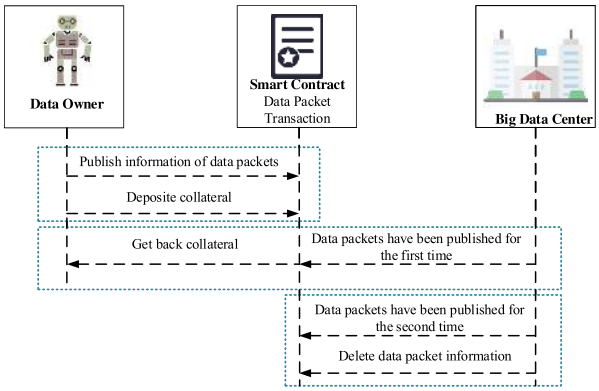


FIGURE 3. Publishing data packets for sale.

been published by the DO. Besides, the DO needs to deposit the collateral that is qual to the amount of *ItemPrice*.

DataCenterCensor(): The BDC needs to censor whether the information of the data packet has already existed on the list of data packet items recorded on the blockchain. If the data packet information is published for the first time, the deposited collateral will be returned to the DO. If the data packet information is published for the second time, the deposited collateral of the DO will not be returned. Besides, the data packet information will be removed from the list of data packet items. The deposited collateral help incentivize DOs to publish their own data packet items honestly.

2) PURCHASING DATA PACKETS

The procedure of DAs purchasing data packets from DOs through the DPT smart contract is shown in Algorithm 1. The process of DCs purchasing data packets from DOs is the same as that of DAs. Three functions including *PurchaseRequest()*, *AccessPermission()* and *DataTradeResult()* are designed to describe how DAs purchase data packets from DOs. *msg.value* in Algorithm 1 is the Ether attached to the current transaction.

PurchaseRequest(): The DA calls *PurchaseRequest()* when the DA wants to buy data packets from the DO. The data packet information that the DA wants to buy will be given. Besides, the DA deposits the collateral and payment *ItemPrice*. The collateral is the same as the amount of *ItemPrice*. Then, *PurchaseRequest()* creates a unique token that is a hash for the DA automatically. The hash is calculated using the *keccak256()*, which is a one-way hash function algorithm selected as the SHA-3 standard and is built in the function of Solidity. The token is generated as *token* = *keccak256 (msg.sender, NeedOwnerID, ItemSequence, block.timestamp, ItemHash)*. *msg.sender* is the Ethereum address of the transaction sender, *block.timestamp* is the timestamp of the current block. The DA who gets the valid token is permitted to download data packets from the DO.

AccessPermission(): The DO decides whether to permit DAs to download data packets by calling *AccessPermission()*. The DO authenticates whether the DA has the token. If the DA has the token, the Ethereum address of the DA will be

Algorithm 1 Purchasing Data Packets

Input: *DataItem()*, deposit of DA *msg.value(DA)*, deposit of DO *msg.value(DO)*, *downloading result*

require(*msg.value(DA)* = 2 × *ItemPrice*)
require(*msg.value(DO)* = *ItemPrice*)
A token is created for DA based on *DataItem()* DA wants to buy.
if *(token = true)* **then**
  Ethereum address of the DA is put into the permission list *PermissionList[]*.
  if *(downloading result = true)* **then**
    DO gets the payment *ItemPrice* and the deposited collateral;
    DA gets the deposited collateral;
    Event *DataTradefinished()*is triggered;
  else
    Transaction would be processed by the BDC off-chain;
    Event *DataTrademistake()* is triggered;
  end
else
  DA cannot download data packets from DO;
end

Algorithm 2 Selecting DA to Provide Data Analytics Service

Input: *Expectprice*, *Expectfinishedtime*, *msg.value(DC)*, *price*, *Finishtime*, *msg.value(DA)*

require(*msg.value(DC)* = 2 × *Expectprice*)
require(*msg.value(DA)* = *Expectprice* + *price*)
require(*Finishedtime*<*Expectfinishedtime*)
if $t < T$ **then**
  foreach *DA* **do**
    if *(msg.value(DA)< = Minimumprice)* **then**
      Minimumprice = *msg.value*;
      Minimumbidder = *msg.sender*.
    end
  end
else
  Event *SelectionsEnd() is triggered;*
  Minimumprice, *Minimumbidder* and *Finishedtime* are returned.
end

put into the permission list of the DO. Then, the DO needs to deposit the collateral that is the same amount of *ItemPrice* in order to guarantee the honesty of the DO.

DataTradeResult(): Once the DA downloads the data packet whose hash value is equal to *ItemHash* from the DO, the DPT is finished. The event *DataTradefinished()* is triggered and the message: ''The data packet transaction between DO and DA is finished'' can be seen from the transaction log. Meanwhile, the payment *ItemPrice* and the collateral deposited by the DO will be transferred to the Ethereum account of the DO. Meanwhile, the DA gets back the deposited collateral. Otherwise, the event *DataTrademistake()* is triggered and the message: ''The data packet transaction between DO and DA is unfinished'' can be seen from the log. The transaction will be processed by the BDC off-chain if the transaction is unfinished.

The BDC solves DPT disputes between DAs and DOs off-chain. DPT disputes may be caused by malicious behaviors of DAs, malicious behaviors of DOs or other reasons. The BDC judges reasons for disputes off-chain and calls different functions to solve disputes. Meanwhile, malicious behaviors will be recorded on the blockchain by the BDC, which have a negative effect on reputations of entities involved in transactions. Three methods designed to solve DPT disputes between DAs and DOs are described as below:

1) Malicious behaviors of DAs: The DA can download the data packet exactly, but the DA claims that it is unable to download the data packet from the DO or the hash value of the data packet is wrong. On this occasion, the collateral of the DA will not be returned. Meanwhile, the payment *ItemPrice* and the collateral deposited by the DO will be transferred to the Ethereum account of the DO.
2) Malicious behaviors of DOs: The DO prohibits the DA in the permission list from downloading data packets. The collateral of the DO will not be returned. Meanwhile, the DA gets back the deposited collateral and the payment.
3) Other factors: The data packet fails to be downloaded by the DA due to some other reasons such as communication interruption, etc. The DA and DO will not be punished. The DA and DO will get back their deposits, respectively. The DO re-send the transaction to purchase the data packet.

B. DAST SMART CONTRACT

In this section, the DAST smart contract is described. DCs buy the data analytics services from DAs by calling functions in the DAST smart contract. The main functions in the DAST smart contract are shown below.

1) SELECTING DA TO PROVIDE DATA ANALYTICS SERVICES

The DC publishes requirements of the data analytics service on the blockchain. DAs respond to the DC by publishing the information of data analytics services they can provide. DAs compete with each other as the DC chooses the data analytics service with the minimum price. The selected DA provides the data analytics service for the DC. Four functions including *ServicesNeed()*, *Selection()*, *SelectionEnd()* and *Refund()* are designed to implement the process. The procedure is shown in Algorithm 2.

ServiceNeed(): The DC publishes requirements of the data analytics service on the blockchain, which contains the information of data packets needed to be analyzed,

the expected service price *Expectprice* and required service time *Expectfinishedtime*. Besides, the DC deposits the payment *Expectprice* and the collateral which is the same amount of *Expectprice*. Then, the event *DataServiceNeed()* is triggered and the data analytics service requested by the DC can be seen from the transaction log.

Selection(): During the period T the DC waits for responses from DAs. DAs who are willing to provide data analytics services for the DC give the service price *price* and the request service time *Finishedtime*. In order to ensure DAs to provide real service information, DAs are required to deposit collateral that is equal to the sum of *Expectprice* and *price*. *Selection()* judges whether the current *msg.value* is lower than the previous collateral and whether *Finishedtime* is lower than *Expectfinishedtime*. If the above conditions are met, the collateral expressed as *Expectprice*+*price* and the address of the DA is updated with *msg.value* and *msg.sender*. Moreover, the previous collateral and address of the DA are recorded in the public variable *CollateralReturns[]* and used as a basis for refunds. The parameter t in algorithm 2 represents the time DCs have already spent on waiting for responses from DAs.

SelectionEnd(): The DC stops waiting for responses from DAs. The DC calls *SelectionEnd()* to select the DA who provides the data analytics service with the minimum service price *Minimumprice*. The event *SelectionsEnd()* is triggered and the information of the selected DA is given, including the Ethereum address of the DA *Minimumbidder*, the minimum service price *Minimumprice* and *Finishedtime*.

Refund(): DAs who are not selected can get back the deposited collateral by calling *Refund()*. Based on the public variable *CollateralReturns[]*, only DAs who had deposited in the smart contract can refund their collaterals.

2) TRANSACTION RESULTS OF DATA ANALYTICS AERVICE

The DC gives the result of the DAST *getservice* by calling function *Servicetraderesult()*. The implementation process is shown in Algorithm 3. *Realfinishedtime* in algorithm 3 represents actual service time spent by the DA. The function *Servicetraderesult()* is shown below.

Servicetraderesult(): If the DA transmits the service result to the DC within the service time *Finishedtime*, the Ether that the DA would receive is expressed as $E^1_{Analyst}$ = *Expectprice*+2×*Minimumprice*. Meanwhile, the Ether that the DC would receive is expressed as $E^1_{Consumer}$ = 2×*Expectprice-Minimumprice*. The event *DataServicefinished()* is triggered, and the message "The data analytics service transaction between DA and DC is finished on time" can be seen from the transaction log. If the DC receives the service result during the time beyond *Finishedtime*, the DC only needs to pay half of the minimum service price *Minimumprice* to the DA and the Ether that the DC would receive is expressed as $E^2_{Consumer}$ = 2× *Expectprice* - 0.5× *Minimumprice*. The Ether that the DA would receive is expressed as $E^2_{Analyst}$ = *Expectprice*+1.5×*Minimumprice*. The event *DataServiceDelayed()* is triggered and the message

Algorithm 3 Transaction Results of Data Analytics Service

Input: *getservice*, *Realfinishedtime*, *Minimumprice*, *Expectprice*, *Finishedtime*

if *(getservice = true)* **then**
  if *(Realfinishedtime< = Finishedtime)* **then**
    Ether transferred to DA: $E^1_{Analyst} = Expectprice + 2 \times Minimumprice$;
    Ether transferred to DC: $E^1_{Consumer} = 2 \times Expectprice - Minimumprice$;
    Event *DataServicefinished()* is triggered;
  else
    Ether transferred to DA:$E^2_{Analyst} = Expectprice + 1.5 \times Minimumprice$;
    Ether transferred to DC:$E^2_{Consumer} = 2 \times Expectprice - 0.5 \times Minimumprice$; Event *DataServiceDelayed()* is triggered;
  end
else
  BDC solves disputes between DC and DA;
  Event *DataServicemistake()* is triggered.
end

"The data analytics service transaction between DA and DC is delayed" can be seen from the transaction log. If the DC does not receive the result of the data analytics service, the DAST dispute is solved by the BDC off-chain. Besides, the event *DataServicemistake()* is triggered and the message "The data analytics service transaction between DA and DC is unfinished" can be seen from the transaction log.

The BDC solves DPT disputes between DAs and DCs off-chain. DAST disputes may be caused by malicious behaviors of DAs, malicious behaviors of DCs or some other reasons. The BDC judges reasons for disputes off-chain and calls different functions to solve disputes. Meanwhile, malicious behaviors will be recorded on the blockchain by the BDC, which have a negative effect on reputations of entities involved in transactions. Three methods designed to solve DAST disputes between DAs and DCs are described as below:

1) Malicious behaviors of DAs: The DA claims to be able to provide the data analytics service but does not provide. The collateral of DA will not be returned. The DC would get back the deposited collateral and payment.
2) Malicious behaviors of DCs: The DC receives the data analytics result but claims that it does not receive. The collateral of DC would not be returned. The DA will get back the deposited collateral and the payment of the data analytics service.
3) Other factors: The data analytics result fails to be transmitted to the DO due to some other reasons such as communication interruption, etc. The DA and the DO will not be punished. The DA will get back the

deposited collateral and the payment of the data analytics service. The DC will get back the deposited collateral. The DA needs to resend data analytics results to the DA.

V. IMPLEMENTATION AND TESTING

DPT and DAST smart contracts are written in the Solidity language and tested in Remix IDE. The Remix IDE that integrates the Solidity compiler, runtime environment, debugging and publishing tools is a browser-based Solidity IDE developed by Ethereum. The runtime environment provided by Remix IDE includes JavaScript virtual machine (JavaScript VM), Injected Web3 and Web3 Provider. In this paper, DPT smart contract and DAST smart contract are tested in the Web3 Provider. Ethereum accounts are provided by Ganache, which is a personal blockchain for Ethereum development and can be used to run tests. The initial balance of all accounts in the Ganache is 100 *ether*. Ether is the currency of the Ethereum network and *ether* is the unit of Ether. In this section, DPT and DAST smart contracts are compiled and tested. The testing results are described below.

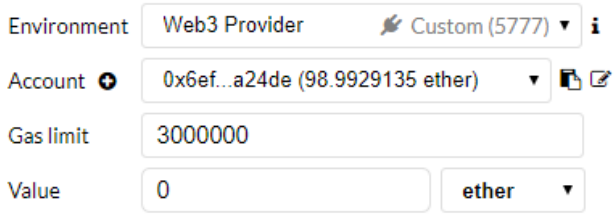
A. DPT SMART CONTRACT

In this section, details of testing the DPT smart contract are analyzed, including running results of functions, outputs of transaction logs and the transfer between accounts, etc. The Ethereum addresses of the DO, BDC and DA are ''0x6ef10eb2884e9dfb74dbcd51bc883827deaa24de'', ''0x5e19ac775caac4cdf306b9b9a2262dab2b75c9fb'',''0x18743d90fb5985e8713ecf93085f8f8af1a6bc51'', respectively. Functions in the DPT smart contract can be executed correctly in Remix IDE. The status of the smart contract will be reset, and the error messages will be returned if the running conditions of functions are not met.

The information of the data packet can be seen from the transaction log when the DO published it on the blockchain. As shown by Fig4 (a), the hash value of the data packet is ''WmSXRSuFMP3aCVSZKpEjPHPUZN2NjB3YrhJTHsV4X3vb2td''. The name of the data packet is ''The machine speed of smart factory A in June 2019''. The identity number of the DO that provides the data packet is ''0''. The sequence number of the data packet is ''0''. The price is ''1 *ether*''. Besides, the message ''This *DataItem* has been added'' can be seen from the transaction log. Fig4(b) shows the balance of the DO as almost 99 *ether*. This indicates that the DO has published the data packet and deposited the collateral of 1 *ether*. The gap between the balance and 99 *ether* is the transaction fee.

The BDC censors whether the information of the data packet published by the DO has already existed. If the data packet is ''publishable'', the collateral of 1 *ether* previously deposited by the DO would be refunded. As shown in Fig 5, the balance of the DO is almost 100 *ether* which is the initial balance of the DO. If the data packet is ''unpublishable'', which means that the data packet has been published for the second time. The deposited collateral of the DO will not

(a) Transaction log



(b) Balance of DO

FIGURE 4. Publishing data packets by DO.

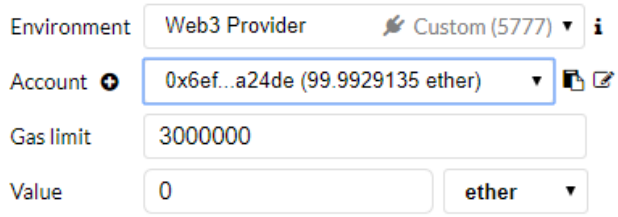


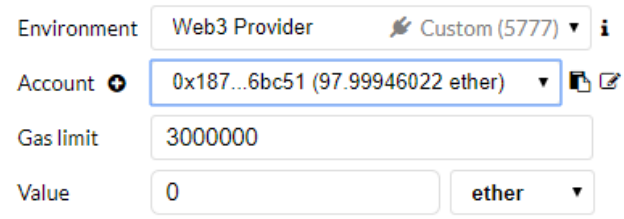
FIGURE 5. Balance of DO with successful data packets publication.

be returned and the information of data packets published will be removed from the data packet items list. Such censorship of the BDC can prevent entities from selling data packets bought from other DOs to some extent. The gap between 100 *ether* and the balance of DO is the transaction fee paid by the DO.

The DA publishes the data packet purchase request and deposits the collateral of 2 *ether*. Then, the DA gets a token which can be used to download the data packet from the DO. As can be seen from Fig 6(a), the unique token generated for the DA with Ethereum address ''0x18743d90fb5985e8713ecf93085f8f8af1a6bc51'' is represented as ''0xc28e6028ffb36745892a9802fc348551b4a590ca3f79da5796890030d9b6681c''. The current timestamp is 1575943863. Fig 6 (b) shows that the balance of the DA is almost 98 *ether*. This indicates that the DA has published the data packet purchase request and deposited 2 *ether*, including the collateral of 1 *ether* and payment of 1 *ether*. The gap between the balance of DA and 98 *ether* is the transaction fee.

The length of the array *PermissionList()* increases by 1 when the DO permits the DA to download the data packet. In Fig 7, the decoded output shows that the length of array *PermissionList[]* is 1.

(a) Transaction log



(b) Balance of DA

FIGURE 6. Publishing data packet purchase request by DA.

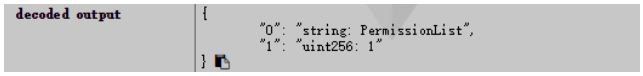


FIGURE 7. Decoded output of downloading permission for DA.

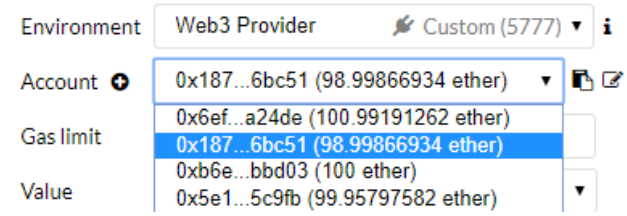
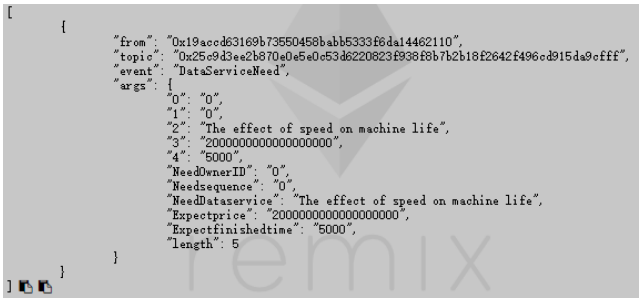


FIGURE 8. Balance of DO and DA after successful downloading.

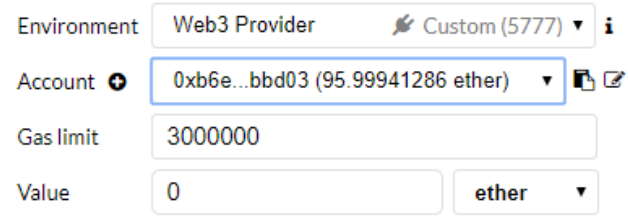
The deposited collateral of the DA and DO is refunded when the DA can download the data packet successfully. The DA pays 1 *ether* to the DO for the data packet. Fig 8 shows the balance of the DO and DA when the DA can download the data packet from the DO successfully. As shown in Fig 8, the balance of the DO is 100.99191262 *ether* and that of the DA is 98.99866934 *ether*. The balance of the DO increases by 2 *ether*, which consists of the collateral of 1 *ether* and the payment of 1 *ether*. The balance of the DA increases by 1 *ether*, which is the collateral deposited by the DA. When the DA cannot download the data packet from the DO successfully, the dispute between the DA and DO would be solved by the BDC off-chain.

B. DAST SMART CONTRACT

In this section, details of testing the DAST smart contract are analyzed. The Ethereum address of the DA and



(a) Transaction log



(b) Balance of DC

FIGURE 9. Publishing data analytics service requirements by DC.

DC are "0x18743d90fb5985e8713ecf93085f8f8af1a6bc51" and "0xb6e386f91010182dbcdce3886762f538d2cbbd03", respectively.

DCs send transactions to publish their data analytics service requirements on the blockchain. As can be seen from the transaction log shown in Fig.9 (a), the data analytics service requested by the DC includes *NeedOwnerID*, *Needsequence*, *NeedDataservice*, *Expectprice* and *Finishedtime*. *NeedOwnerID* is the ID of the DO, *Needsequence* is the requested data packet sequence, *NeedDataservice* is the name of the requested data analytics service, *Expectprice* is the expected price of the data analytics service and *Expectfinishedtime* is the expected time to wait for analytics results. As shown in Fig 9 (b), the balance of the DC has reduced 4 *ether*. It means that the DC has deposited 4 *ether*, which consists of the collateral of 2 *ether* and the payment of 2 *ether*.

DAs send transactions to publish data analytics services they can provide based on service requirements of the DC. The provided data analytics services contain the service price *price* and service time *Finishedftime*. In order to prevent the DA from providing invalid service information, the DA needs to deposit the collateral which is the sum of *Expectprice* and *price*. As can be seen from Fig 10, the balance of the DA has reduced 4 *ether* that is the collateral deposited by the DA.

The DC selects a DA who provides the data analytics service with the minimum service price. As can be seen from the transaction log shown in Fig.11, the DC select the DA whose Ethereum address is "0x18743d90fb5985e8713ecf93085f8f8af1a6bc51". The price of the data analytics service provided by the DA is 2 *ether* and the service time is 4000 seconds.

The DC sends the transaction to claim whether it has gotten the data analytics service in the expected service time. Fig.12 (a) shows the transaction log and the message that

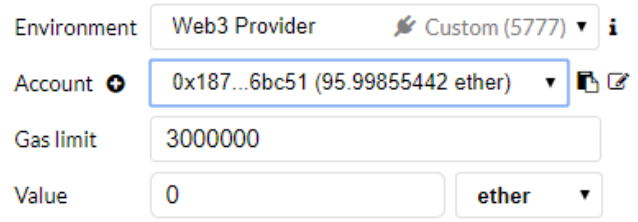
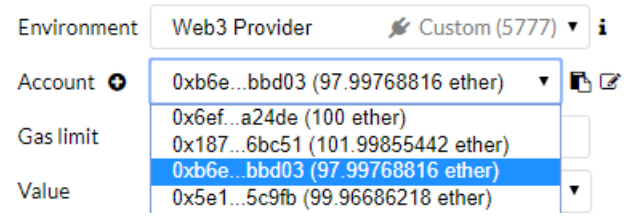


FIGURE 10. Balance of DA after responding to DC.

FIGURE 11. Transaction log of data analytics service chose by DC.

(a) Transaction log



(b) Balances of DA and DC

FIGURE 12. Data analytics service transaction.

is "The data analytics service transaction between DA and DC is finished on time". It means that the DA has finished the data analytics service and send the service result to the DC on time. Fig.12(b) shows the balances of DA and DC, the balance of the DA is 101.99855442 *ether*, and that of the DC is 97.99768816 *ether*. It means that the DA receives the deposited collateral of 4 *ether* and the payment of 2 *ether*. The DC receives the deposited collateral of 2 *ether*.

C. COST ANALYSIS

In this section, the gas costs of the DPT smart contract and DAST smart contract are analyzed. Each transaction running on the blockchain network costs a certain amount of gas, which limits the amount of work required to execute a transaction. The transaction fee on the Ethereum platform is related to two factors, i.e., the gas limit and gas price. The gas limit that is determined by the sender indicates the maximum amount of gas that the sender is willing to pay before sending the transaction. The setting of gas price affects the speed that miners package transactions into blocks. Transactions with high gas price can be packaged into blocks more quickly than the transactions with low gas price. MetaMask makes a best guess at the setting of gas limit. The value of gas price is usually denoted by $10^9 wei$ and $1\ ether = 10^{18} wei$, where *wei* and *ether* are units of Ether. Gas can be purchased by Ether. The transaction fee is calculated as: $transaction\ fee = gas\ used \times gas\ price$.

TABLE 1. Gas costs of functions in DPT smart contract.

Function Caller	Function Name	Transaction Gas	Gas cost (*ether*)
DO	*AddDataItem*	354325	3.54325×10^{-4}
BDC	*DataCenterCensor*	31284	3.1284×10^{-5}
DA	*PurchaseRequest*	26989	2.6989×10^{-5}
DO	*AccessPermission*	50044	5.0044×10^{-5}
DA	*DatatradeResult*	39544	3.9544×10^{-5}

TABLE 2. Gas costs of functions in DAST smart contract.

Function Caller	Function Name	Transaction Gas	Gas cost (*ether*)
BDC	*ServicesNeed*	29357	2.9357×10^{-5}
DA	*Selection*	72279	7.2279×10^{-5}
DC	*SelectionEnd*	44956	4.4956×10^{-5}
DA	*Refund*	21947	2.1947×10^{-5}
DC	*Servicetraderesult*	41279	4.1279×10^{-5}

Table 1 shows gas costs of all functions in the DPT smart contract. Transaction gas includes the gas cost by contract execution, value transfer and data transfer. The first column in Table 1 indicates that the callers of functions are the DO, DC, DA and BDC. From table 1, the function with the most amount of gas cost is *AddDataItem()*, which is 3.54325×10^{-4} *ether*. The overall costs of other functions are small and all of them are below 5.0044×10^{-5} *ether*. Table 2 shows gas costs of all functions in the DAST smart contract. As can be seen from table 2, the overall costs of all functions are small which are below 7.2279×10^{-5} *ether*. Based on the analysis, the gas costs of smart contracts proposed in this paper are small.

D. SECURITY ANALYSIS

In this section, the security of the proposed transaction solution is analyzed. The distributed consensus of the blockchain technology eliminates the distrust between entities and centralized transaction platforms. Instead of selling data packets or data analytics services through centralized big data platforms, entities can send transactions in a decentralized way, which can also eliminate the single point of failure in centralized transaction platforms. Moreover, the tamper-proof and traceable characteristics of the blockchain technology prevent all entities in IIoT from denying their actions saved in the tamper-proof logs. The DA or DC accesses data packets of DOs off-chain by using the unique token that is calculated using timestamp and unique identifier. Thus, only authorized entities can download data packets from DOs, which can prevent replay and Man-in-the-Middle attacks.

TABLE 3. Comparison of proposed solution with existing solutions.

Characteristic	[9]	[20]	[24]	[26]	Proposed solution
Decentralized	×	×	√	√	√
Traceable	×	×	√	√	√
Unforgeable	×	×	√	√	√
DPT	√	√	√	√	√
DAST	√	×	×	×	√
Proof of delivery	×	×	×	√	√

The smart contract that is a piece of code resides on the blockchain allows entities in IIoT send transactions in an automatic and honest way. Transaction rules and penalty rules are stipulated in both DPT smart contract and DAST smart contract. Once entities go against transaction rules, the entities will be punished based on penalty rules, which will bring financial loss to the entities. Moreover, the blockchain technology is combined with the management of BDC. The transaction disputes between different entities in IIoT will be processed by the BDC, and handling results will be recorded on the blockchain by the BDC. Modifiers in the smart contract code limit entities to executing functions. Only if the Ethereum address of the sender matches the authorized Ethereum address can the function be executed. So modifiers can restrict malicious users from calling specific functions in smart contracts at random.

E. COMPARISON

In this section, the proposed transaction solution is compared with other transaction solutions, and comparison results are shown in Table 3. In general, the proposed transaction solution is more advantageous than other four solutions in terms of trading data packets and data analytics services in a decentralized way.

VI. CONCLUSION

In this paper, a new transaction solution consists of DPTs and DASTs in IIoT was proposed. The smart contract-enabled blockchain technology was introduced to realize a decentralized transaction solution. The DPT smart contract was designed to trade data packets while the DAST smart contract provided a competitive way to trade data analytics services. Different entities in IIoT execute transactions by calling functions in smart contracts automatically and honestly. Disputes between different IIoT entities can be solved by the BDC off-chain, and results of handling would be recorded on the blockchain by the BDC. The DPT smart contract and DAST smart contract were written in the Solidity language and tested on Remix IDE. Gas costs of functions in the DPT smart contract and DAST smart contract were estimated, all of which are small. The security of the proposed transaction solution was also analyzed, which is resilient against some known attacks. In the future, we will investigate distributed consensus and reputation problems in the decentralized and distributed IIoT data market.

REFERENCES

[1] H. Yao, T. Mai, J. Wang, Z. Ji, C. Jiang, and Y. Qian, "Resource trading in blockchain-based industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3602–3609, Jun. 2019.

[2] Y. Jiang, X. Ge, Y. Zhong, G. Mao, and Y. Li, "A new small-world IoT routing mechanism based on Cayley graphs," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10384–10395, Dec. 2019, doi: 10.1109/JIOT.2019.2938800.

[3] X. Ge, "Ultra-reliable low-latency communications in autonomous vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5005–5016, May 2019.

[4] Y. Zhong, T. Q. S. Quek, and X. Ge, "Heterogeneous cellular networks with spatio-temporal traffic: Delay analysis and scheduling," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 6, pp. 1373–1386, Jun. 2017.

[5] X. Ge, Y. Sun, H. Gharavi, and J. Thompson, "Joint optimization of computation and communication power in multi-user massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4051–4063, Jun. 2018.

[6] S. Xiao, H. Yu, Y. Wu, Z. Peng, and Y. Zhang, "Self-evolving trading strategy integrating Internet of Things and big data," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2518–2525, Aug. 2018.

[7] Y. Zhong, X. Ge, H. H. Yang, T. Han, and Q. Li, "Traffic matching in 5G ultra-dense networks," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 100–105, Aug. 2018.

[8] F. Liang, W. Yu, D. An, Q. Yang, X. Fu, and W. Zhao, "A survey on big data market: Pricing, trading and protection," *IEEE Access*, vol. 6, pp. 15132–15154, 2018.

[9] Y. Jiao, P. Wang, S. Feng, and D. Niyato, "Profit maximization mechanism and data management for data analytics services," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2001–2014, Jun. 2018.

[10] B. Yu, J. Wright, S. Nepal, L. Zhu, J. Liu, and R. Ranjan, "IoTChain: Establishing trust in the Internet of Things ecosystem using blockchain," *IEEE Cloud Comput.*, vol. 5, no. 4, pp. 12–23, Aug. 2018.

[11] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data*, Jun. 2017, pp. 557–564.

[12] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: Nov. 28, 2017. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[13] V. Buterin, "A next-generation smart contract and decentralized application platform," Ethereum, Zürich, Switzerland, White Paper 3, 2014, vol. 3. [Online]. Available: https://cryptorating.eu/whitepapers/Ethereum/Ethereum_white_paper.pdf

[14] M. S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli, and M. H. Rehmani, "Applications of blockchains in the Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1676–1717, 2nd Quart., 2019.

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

[16] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, "Blockchain technologies for the Internet of things: Research issues and challenges," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2188–2204, Apr. 2019.

[17] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9, pp. 1–25, Sep. 1997. [Online]. Available: http://journals.uic.edu/ojs/index.php/fm/article/view/548

[18] Y. Hanada, L. Hsiao, and P. Levis, "Smart contracts for machine-to-machine communication: Possibilities and limitations," in *Proc. IEEE Int. Conf. Internet Things Intell. Syst. (IOTAIS)*, Bali, Indonesia, Nov. 2018, pp. 130–136.

[19] D. Niyato, D. T. Hoang, N. C. Luong, P. Wang, D. I. Kim, and Z. Han, "Smart data pricing models for the Internet of Things: A bundling strategy approach," *IEEE Netw.*, vol. 30, no. 2, pp. 18–25, Feb. 2016.

[20] Z. Zheng, Y. Peng, F. Wu, S. Tang, and G. Chen, "Trading data in the crowd: Profit-driven data acquisition for mobile crowdsensing," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 2, pp. 486–501, Feb. 2017.

[21] L. Guijarro, V. Pla, J. R. Vidal, and M. Naldi, "Maximum-profit two-sided pricing in service platforms based on wireless sensor networks," *IEEE Wireless Commun. Lett.*, vol. 5, no. 1, pp. 8–11, Feb. 2016.

[22] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Blockchain for secure EHRs sharing of mobile cloud based e-Health systems," *IEEE Access*, vol. 7, pp. 66792–66806, 2019.

[23] S. Wang, Y. Zhang, and Y. Zhang, "A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems," *IEEE Access*, vol. 6, pp. 38437–38450, Jun. 2018.

[24] A. Suliman, Z. Husain, M. Abououf, M. Alblooshi, and K. Salah, "Monetization of IoT data using smart contracts," *IET Netw.*, vol. 8, no. 1, pp. 32–37, Jan. 2019.

[25] J. Kang, R. Yu, X. Huang, M. Wu, S. Maharjan, S. Xie, and Y. Zhang, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4660–4670, Jun. 2019.
[26] H. Hasan and K. Salah, "Proof of delivery of digital assets using blockchain and smart contracts," *IEEE Access*, vol. 6, pp. 65439–65448, 2018.
[27] J. Zhou, F. Tang, H. Zhu, N. Nan, and Z. Zhou, "Distributed data vending on blockchain," 2018, *arXiv:1803.05871*. [Online]. Available: https://arxiv.org/abs/1803.05871
[28] *AAAChain White Paper*. Accessed: Dec. 16, 2019. [Online]. Available: https://aaachain.net/#whitepaper
[29] *GXChain White Paper*. Accessed: Dec. 16, 2019. [Online]. Available: https://static.gxb.io/files/GXChain_WhitePaper_v3.0_EN.pdf
[30] *DXChain White Paper*. Accessed: Dec. 16, 2019. [Online]. Available: https://www.dxchain.com/static/assets/docs/DxChain-Whitepaper.pdf
[31] *XChain White Paper*. Accessed: Dec. 16, 2019. [Online]. Available: http://nxct.io/
[32] S. Wu, Y. Chen, Q. Wang, M. Li, C. Wang, and X. Luo, "CReam: A smart contract enabled collusion-resistant E-auction," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 7, pp. 1687–1701, Jul. 2019.
[33] J. Pan, J. Wang, A. Hester, I. Alqerm, Y. Liu, and Y. Zhao, "EdgeChain: An edge-IoT framework and prototype based on blockchain and smart contracts," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4719–4732, Jun. 2019.
[34] R. Xu, Y. Chen, E. Blasch, and G. Chen, "BlendCAC: A blockchain-enabled decentralized capability-based access control for IoTs," in *Proc. IEEE Int. Conf. Internet Things (iThings), IEEE Green Comput. Commun. (GreenCom), IEEE Cyber, Phys. Social Comput. (CPSCom), IEEE Smart Data (SmartData)*, Halifax, NS, Canada, Jul./Aug. 2018, pp. 1027–1034.

YUNA JIANG received the B.E. degree in communications engineering from the China University of Mining and Technology, Xuzhou, China, in 2017. She is currently pursuing the Ph.D. degree with the School of Electronic Information and Communications, Huazhong University of Science and Technology. Her research interests include the Internet of Things and blockchain technology.

YI ZHONG (S'12–M'15) received the B.S. and Ph.D. degrees in electronic engineering from the University of Science and Technology of China (USTC), in 2010 and 2015, respectively. From August 2012 to December 2012, he was a Visiting Student with the Prof. M. Haenggi's Group, University of Notre Dame. From July 2013 to October 2013, he was a Research Intern with Qualcomm Incorporated, Corporate Research and Development, Beijing. From July 2015 to December 2016, he was a Postdoctoral Research Fellow with the Wireless Networks and Decision Systems (WNDS) Group, led by Prof. T. Q. S. Quek, Singapore University of Technology and Design (SUTD). He is currently an Assistant Professor with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, China. His main research interests include heterogeneous and femtocell-overlaid cellular networks, wireless ad hoc networks, stochastic geometry, and point process theory.

XIAOHU GE (M'09–SM'11) received the Ph.D. degree in communication and information engineering from the Huazhong University of Science and Technology (HUST), in 2003. Since November 2005, he has been with HUST, China, where he is currently a Full Professor with the School of Electronic Information and Communications. He is also an Adjunct Professor with the Faculty of Engineering and Information Technology, University of Technology Sydney (UTS), Australia. Prior to HUST, he worked as a Researcher at Ajou University, South Korea, and the Politecnico Di Torino, Italy, from January 2004 to October 2005. His research interests are in the area of mobile communications, traffic modeling in wireless networks, green communications, and interference modeling in wireless communications. He has published about 200 articles in refereed journals and conference proceedings and has been granted about 25 patents in China. He services as an IEEE Distinguished Lecturer and an Associate Editor for IEEE Access, the IEEE Wireless Communications, and the IEEE Transactions on Vehicular Technology.