Hindawi
Security and Communication Networks
Volume 2021, Article ID 3050953, 13 pages
https://doi.org/10.1155/2021/3050953

*Research Article*

# Towards Trustworthy IoT: A Blockchain-Edge Computing Hybrid System with Proof-of-Contribution Mechanism

**Huan Dai**,[1] **Pengzhan Shi**,[1] **He Huang**,[2] **Ruyu Chen**,[1] **and Jun Zhao**[1]

[1]*School of Electronic and Information Engineering, Suzhou University of Science and Technology, Suzhou 215000, China*
[2]*School of Computer Science and Engineering, Soochow University, Suzhou 215000, China*

Correspondence should be addressed to Huan Dai; daihuanjob@163.com

Received 1 June 2021; Accepted 14 August 2021; Published 29 August 2021

Academic Editor: Jie Cui

The emerging smart city is driving massive transformations of modern cities, facing the huge influx of sensor data from IoT devices. Edge computing distributes computing tasks to the near-edge end, which greatly enhances the service quality of IoT applications, that is, ultralow latency, large capacity, and high throughput. However, due to the constrained resource of IoT devices, currently, systems with a centralized model are vulnerable to attacks, such as DDoS from IoT botnet and central database failure, which can hardly provide high-confidence services. Recently, blockchain with a high security promise is considered to provide new approaches to enhancing the security of IoT systems. However, blockchain and IoT have obvious incompatibility, and low-capacity IoT devices can hardly be incorporated into blockchain with high computing requirements. In this paper, a blockchain-edge computing hybrid system (BEHS) is presented to make the adaptation of blockchain to edge computing and provide trustworthy IoT management services for a smart city. A novel extensible consensus protocol designed for proof-of-work, named proof-of-contribution (PoC), is proposed to regulate the data upload behaviors of nodes, especially the data upload frequency of IoT device nodes, so as to protect the system from attack about frequency. In order to secure the data privacy and authenticity, a data access control scheme is designed by integrating symmetric encryption with asymmetric encryption algorithm. We implemented a concrete BEHS on Ethereum, realized the function of PoC mechanism via smart contracts, and conducted a case study for smart city. The extensive evaluations and analyses show that the proposed PoC mechanism can effectively detect and automatically manage the behavior of nodes, and the time cost of data access control scheme is within an acceptable range.

## 1. Introduction

*1.1. Motivations.* The rapid advancement of the Internet-of-Things (IoT) technologies has greatly promoted the intelligent transformation of modern cities, making the realization of smart city getting closer [1, 2]. The extensive usage of IoT devices causes the storm growth of cloud traffic. Edge computing is an emerging computing paradigm, which decentralizes computing tasks to the near-edge end to improve the quality of smart city applications and services [3]. Meanwhile, the rise of IoT has also brought some security concerns to smart cities. Due to the constrained resources, IoT devices are vulnerable to attacks. In 2016, the Dyn data center was attacked by large-scale DDoS attack from IoT devices affected by botnet, resulting in a long time interruption of relevant services and a large number of enterprise losses [4]. Moreover, systems with centralized model are vulnerable to central database failure and are not conducive to secure the data authenticity. In addition, lack of promised approaches of value transmission constrains the application scenario of IoT [5].

Cloud and edge computing also have some problems to be solved. For example, cloud platform that stores heterogeneous data is still not yet deviated from the essence of centralization, which leads to users overrelying on trust in cloud platforms for data access control. Furthermore, edge computing has the ability to continuously receive and process omnipresent data, which seems to be unremarkable in the field of data privacy protection. More importantly, the operation and maintenance of such a large yet centralized

IoT system requires an immense cost. These problems are restricting the development and progress of the Internet of things. Bitcoin [6], a decentralized cryptocurrency introduced by Nakamoto in 2008, provides a trustworthy method to transfer information in the untrustworthy environment, which is the first phenomenal application of blockchain. Owing to the distributed and digital trust properties of blockchain, the integration of blockchain into IoT architecture based on edge computing is a feasible and potential scheme [7–9].

Zhang et al. [10] designed a smart contract-based framework to investigate the access control issue in IoT systems. Huang et al. [11] developed a high-throughput industrial IoT blockchain system based on the principle of directed acyclic graph in distributed ledger technology (DLT). Pan et al. [12] proposed an EdgeChain framework to link the resource of edge servers with IoT objects by a coin system based on smart contracts. However, there are challenges remaining unsolved when integrating blockchain with edge computing. Numerous and heterogeneous IoT devices make smart contract undertake higher complexity and storage cost, which leads to inefficiency of smart contract-based IoT management and can hardly meet the requirement of high throughput of IoT systems. Due to the constrained resource, IoT devices can hardly adapt to consensus algorithms with high computational complexity and large storage requirement, such as proof-of-work [13] and proof-of-stake [14], which makes IoT devices fall out system supervision and become vulnerabilities. Additionally, the transparency of blockchain makes it difficult to protect data privacy, which is contrary to the requirements of IoT systems. Even in the permissioned chain, data is not always intended to be disclosed to all permission participants. So far, there is a lack of blockchain-based scheme designed specifically for the IoT devices.

*1.2. Related Work.* The conventional IoT systems have achieved improvements in computing and storage capabilities based on cloud computing. However, IoT devices, as a large number of writers in the system, cannot verify the integrity of stored data. The questionable credibility of IoT devices and the complexity of the network also pose challenges to information security and data privacy of large-scale smart city systems. While centralized systems have strong performance, it is difficult to be applied in IoT scenarios due to the vulnerability of single point of failure and zero-tolerance of malicious writers. Instead, blockchain, a special distributed ledger technology that sacrifices performance in exchange for trust and availability, is considered a new solution. However, the performance of blockchain is difficult to meet the requirements of IoT with massive data. Because of this, a lot of works have been done to improve the scalability of consensus algorithm or blockchain on the premise of ensuring the security. Biswas et al. [15] present a novel lightweight proof of block and trade (POBT) consensus algorithm for IoT blockchain and its integration framework, allowing the validation of trades as well as blocks with reduced computation time. They proposed a new allocation mechanism to reduce the memory requirements of IoT nodes. Viriyasitavat et al. [16] analyzed the pressure and risks of the quality of service (QoS) in the IoT and integrated the blockchain technologies (BCT) with a multiagent approach to ensure the reliability of real-time data and achieve the measurement of QoS in the IoT environment. Guo et al. [17] constructed collaborative mining network (CMN) to execute mining tasks for mobile blockchain, which solves the problem that IoT mobile devices cannot afford the high computing cost of blockchain due to the limitations of communication and computing.

Wang et al. [18], especially, designed a trust consensus scheme for IIoT; it can be implemented on the state-of-the-art PoX consensus protocols. The reputation module of this scheme is equipped with an incentive mechanism; the participants will be motivated to make honest behavior and contribution for network. However, IIoT devices are defined as nodes in the blockchain network, which overestimates the storage and computing capacity of IIoT devices. Song et al. [19] proposed a proof-of-contribution consensus mechanism for intellectual property protection, which quantifies various behaviors and actions of nodes into specific contribution values. When the current state of system meets the conditions for generating a new block, nodes are sorted by their contribution values, and the node with the highest values will become the new bookkeeping node. However, although assigning contribution values to each node can effectively improve bookkeeping credibility, overreliance on contribution will aggravate the centralization.

*1.3. Contributions.* To address the aforementioned challenges, we proposed a novel blockchain-edge computing hybrid system (BEHS) to provide trustworthy IoT service for smart cities. The core idea of BEHS is to integrate edge computing with permissioned chain to enhance the security of IoT system while ensuring efficiency. Considering the system scalability, it is designed to have multilayers and multiple modules, which can run on different IoT systems, for example, smart home, smart industry, and smart transport. A novel proof-of-contribution consensus mechanism is proposed to regulate the behavior of nodes, especially IoT device nodes, securing the system from malicious attacks. A data access control scheme, which integrates the symmetric cryptography with the asymmetric cryptography, is also presented to secure the data privacy and authenticity during the communication. As a short summary, our main contributions of this paper include the following:

(1) A novel framework integrates permissioned chain with edge computing, providing a decentralized model for IoT.

(2) A proof-of-contribution (PoC) consensus mechanism is developed to provide a trustworthy management method for the nodes in IoT systems.

(3) A data access control scheme is designed to realize the directional transmission and the privacy protection of IoT data.

(4) We implemented a concrete system on Ethereum and conducted experiments to evaluate the system.

The remaining of this article is organized as follows: Section 2 presents the overview design of BEHS. Section 3 introduces a concrete BEHS on Ethereum platform. The evaluation is discussed in Section 4. The conclusions are discussed in Section 5.

## 2. Blockchain-Edge Computing Hybrid Systems for Trustworthy IoT

In this section, we introduced the proposed system with its key modules. The overview framework design of the system is presented first, and then, we introduced the PoC consensus mechanism and the data access control scheme in detail.

*2.1. Overall Framework Design.* The system framework is built on permissioned-chain and edge computing. As depicted in Figure 1, it can be divided into two layers with four essential modules. Infrastructure layer includes sensing device, blockchain, and edge server modules, which support the system environment and functionality. Application layer implements specific services for users, which commonly relies on the cloud to realize its function.

Multiple types of sensing devices, edge server groups in multiple regions, and blockchain key approaches form the infrastructure layer. Each node has a unique identity, a specific address, and a pair of public/private keys. A private peer-to-peer (P2P) network [20] is set up for communication, where all nodes can discover each other, transmitting and broadcasting transaction information.

Sensing devices collect and collate the samples data measured from physical environment. The data will be uploaded to several nearby edge servers at the same time, and edge servers will broadcast the data throughout the whole network, thus adding the data to the transaction pool, waiting for edge servers to pack. In this way, the system separates the one-to-one subordination relationship between sensing device and edge server. Every sensing device can be a stand-alone node, peer-to-peer with the edge server and constrained by the system rules.

Edge servers have powerful computing and network resources, providing the calculation power for generating new blocks and securing system consistency. The data from sensing devices will be stored into blocks, and the generation of a new block should contain the hash value of its previous block. Thus, blocks are stored in a chain structure to form the ledger. Once formed, it is hard to change any part of the ledger. Every edge server stores a real-time updated backup of the ledger locally to make the distributed storage of data in the system.

Blockchain supports vital security functions, including consensus mechanism and encryption algorithm. Consensus mechanism ensures the consistency of the ledger stored in edge servers. We considered that the regulatory function of sensing devices should be included in the consensus mechanism, which is the key to the separation of edge devices from the subordinate relationship between servers. From this, we design a novel consensus mechanism, named proof-of-contribution (PoC).

Encryption algorithm and digital signature algorithm secure the communication between nodes. Data processed by encryption algorithm is usually difficult to be cracked in blockchain system, but while ensuring the security, it also undermines the convenience of data sharing. The digital signature algorithm can effectively ensure the integrity, credibility, and nonrepudiation of data, which is one of the reliable technical means of data sharing. Therefore, in the application scenario of smart city, we designed the data access control scheme that integrates encryption algorithm and digital signature algorithm to balance the requirements of security and data sharing.

Cloud is the interface of services for users, such as visualized analysis, device management, and privacy protection. Its implementation commonly relies on website platform, applet of WeChat, apps, and so on. Moreover, the system retains the valuable token mechanism, which is designed as a value container. It has a novel function: digitally incentivize the contribution and loyalty behavior of each node for the system. This mechanism encourages devices to upload timely and authentic data and stimulates the participation of merchants and other stakeholders to promote the development of IoT. In addition, the token system gives the ability to transfer value between entities and expands the application of IoT in economy related scenarios.

*2.2. Proof-of-Contribution Mechanism.* In this subsection, we present a novel consensus mechanism, named proof-of-contribution (PoC), which can synchronize the ledger and regulate the behavior of nodes, that is, sensing device node (SDN) and edge server node (ESN). PoC is inspired by PoW mechanism, which has been upgraded to adapt to IoT systems. We considered behaviors that benefit the system's services as contributing to the system. Sensing device contributes to the system by uploading data, and edge server contributes to the system by mining block. Any contribution will be recorded in the block, forming the system's ledger, and PoC rewards the contributors for encouraging their behaviors.

*2.2.1. Edge Server.* The edge server node (ESN) is the miner. Any behavior of nodes will be spread throughout the system and be added to the local transaction pool in every ESN. ESNs pack the behaviors in the transaction pool as an incomplete block and solve a hash puzzle to generate the block and log these behaviors. By inputting the information in a block into secure hash algorithm (SHA) function, such as SHA256 [21], the hash value of the block can be obtained. When the previous block hash value, Merkle root of the behaviors, timestamp, and nonce are input to SHA function, and the output is within the target range, the hash puzzle is solved, and the block is packed. If other ESNs verify that the block is correct and first published, the block will be accepted as the latest block in the ledger of the system.

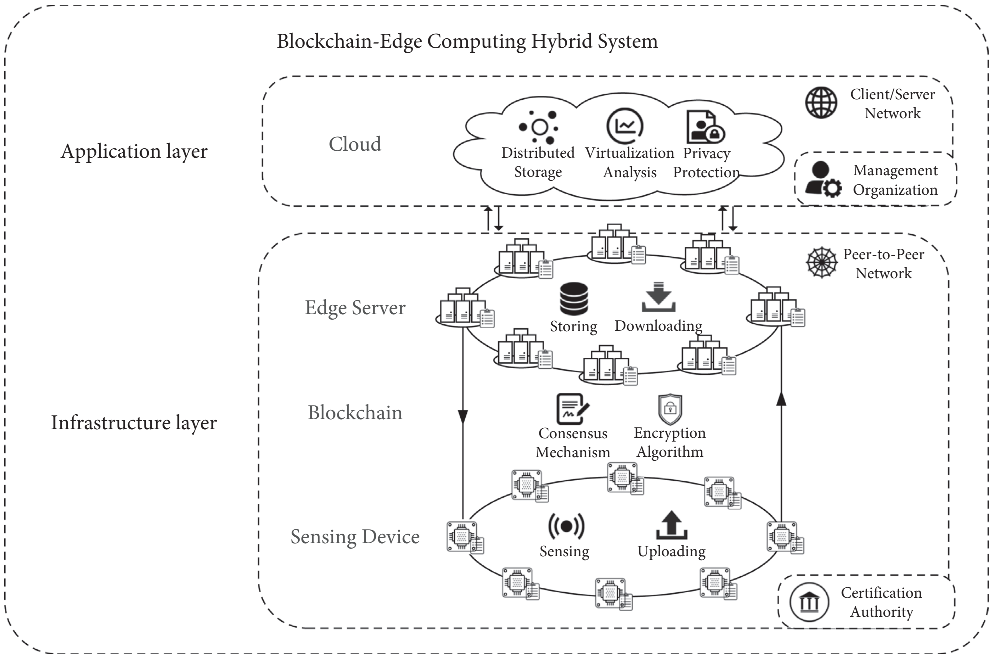

Figure 1: Framework design of the blockchain-edge computing hybrid system.

ESNs store the verified blocks locally. Since each block contains its previous block hash, the blocks are stored with chain structure in each ESN, thus making the distributed ledger. Due to the chain storage structure and large computing power for solving the hash puzzle, once the ledger is formed, it will be hard to tamper with the ledger's content. When a new block is verified and added into the distributed ledger as the latest block, PoC will reward the ESN that contributes to the block generation. Therefore, the structure of the block is designed to contain the address of contributors, as shown in Table 1.

From the aforementioned block verification scheme, the ESN with a higher computational power of hash might have a higher chance to find the correct nonce and obtain the reward. The probability $P_e$ that the ESN $e$ gets the reward of generating a new block with $N$ ESNs is

$$P_e = \frac{H_e}{\sum_{k=1}^{N} H_k} \cdot D_e \cdot \sigma, \tag{1}$$

where $H_e$ is the hash computational power of $e$, $H_k$ is the hash computational power of $k$, $D_e$ is impact factor of the density of ESNs adjacent to $e$ in the network topology, and $\sigma$ is the factor weight of the impact of time consumption. Due to the time consumption of message propagation in the system, ESNs close to the SDNs will receive behavior messages first, which means that it will start searching for nonce earlier than the ESNs far away from the SDNs. Therefore, the number of ESNs in the region, where ESNs are located, has a great impact on the opportunity to obtain rewards of mining, which conforms to the concept of edge computing. This makes ESNs with low computing power still have a relatively reasonable chance to get rewards through mining, effectively preventing ESNs with strong computing power from combining to monopolize rewards and opportunities of block generation.

Table 1: Block structure designed for BEHS.

| Block header | |
|---|---|
| Number | Block height number |
| Timestamp | Creation time of the block |
| Hash | Hash of the block |
| ParentHash | Hash of the previous block |
| Nonce | Nonce |
| ESN | Address of ESNs |
| SDN | Address of SDNs |
| Difficulty | Mining difficulty value |
| Size | Length of the block in bytes |
| Block body | |
| Transaction | List of included behaviors |

The computing resources of all the ESNs in a region can be considered a computing pool. When the computing resources overflow the pool, as the number of ESNs increases, the chance of each ESN to get a reward through mining will be reduced. Therefore, the mutual competition between ESNs in the same region will lead to the renewal and elimination of ESNs, which will allow the number of ESNs to be adapted to the number of SDNs. This also promotes the competition of interest organizations, which are the most innovative role in smart cities and provide fresh blood for the development of IoT industries.

*2.2.2. Sensing Server.* We defined that each sensing device node (SDN) $s$ has a property of credit value Cs and a property of subcategory value $r$. PoC will dynamically evaluate the credit score of the SDN based on its past behaviors. The normal behaviors, that is, the SDN obeying the prefixed rules to upload data, will increase its credit score, while the abnormal behaviors, for example, the SDN uploading data at an incorrect time interval or format will reduce its credit score. When a block is accepted by the system, the SDN contributing to the block will be rewarded. SDNs with high credit score are considered honest and have relatively high rewards. On the contrary, SDNs with low credit score will receive relatively less rewards. When the credit score of a SDN falls below the system threshold, it will be considered as a malicious attacker and be removed from the system.

Before giving the detailed mechanism of PoC for SDNs, we first introduced the possible existing abnormal behaviors of SDNs in the system.

(1) Extra gain: in order to gain more reward, a "greedy" SDN wants to compete inappropriately for rewards, for example, arbitrarily shortening the upload interval. This will lead to unfair reward competition among SDNs and result in system resource redundancy.
(2) Lazy strike: when a SDN fails to finish the work in time, it will be considered as a "lazy" device and stop contributing to the system. Although it does not threaten the security of the system, it affects the normal operation of related services in the system, which can reduce the service quality of the system.
(3) Malicious attack: a "malicious" SDN would want to monopolize the upload authority or prevent others from uploading. It sacrifices itself to disrupt the normal operation of the system, for example, uploading at an ultrahigh frequency. This will cause network congestion and waste system resources, making the system unable to handle normal transactions.

Thus, according to the past behavior of $s$, the $C_s$ can be denoted as

$$C_s = \delta_1 \cdot C_s^N + \delta_2 \cdot C_s^A, \qquad (2)$$

where $C_s^N$ represents the score of normal behaviors, $C_s^A$ represents the score of abnormal behaviors, and $\delta_1$ and $\delta_2$ represent the system sensitivity to normal behaviors and abnormal behaviors, respectively, which can be adjusted dynamically to distribute the weight of these two parts according to application requirements.

$C_s^N$ evaluates the quality of work completed by $s$, which is positively related to the upload interval $\Delta t$. When the interval is between $(2-\alpha_s)\overline{t_s}$ and $\alpha_s\overline{t_s}$ ($\alpha_s$ is the preset parameter of the reasonable range of the specified upload interval $\overline{t_s}$, which is determined by the subcategory $r$ of $s$), the behavior will be regarded as normal. Thus, $C_s^N$ can be defined as

$$C_s^N = \sum_{i=1}^{K_s^N} \frac{1}{\eta^{\left|\overline{t_s}-\Delta t\right|} \cdot (t-t_i)^{\zeta_1}} (2-\alpha_s), \quad \overline{t_s} > \Delta T > \alpha\overline{t_s}, \qquad (3)$$

where $K_s^N$ represents the total number of normal behaviors conducted by $s$, $t$ represents current time, $t_i$ represents the time point of the $i$th behavior conducted by $s$, $\Delta t$ represents the time interval of $i$th behavior and $(i-1)$ th behavior conducted by $s$, $\eta$ represents the sensitivity factor of the difference between rated interval and actual interval, and $\zeta_1$ is the impact index of time on the credit score of normal behaviors.

As described in (3), we can observe that the credit score of normal behaviors of a SDN is related to the quality of its work completion, that is, the upload interval. Besides, as time goes on, the influence of past behaviors on the credit score will decrease.

$C_s^A$ is negatively related to the upload interval, which can be defined as

$$C_s^A = -\sum_{i=1}^{K_s^A} \frac{\left|\overline{t_s}-\Delta t\right|}{(t-t_i)^{\zeta_2}+\kappa}, \quad \Delta t \le \alpha_s\overline{t_s} \cup \Delta t \ge (2-\alpha_s)\overline{t_s}, \qquad (4)$$

where $K_s^A$ represents the total number of abnormal behaviors conducted by $s$, $\zeta_2$ is the impact index of time on the credit score of abnormal behaviors, and $\kappa$ is the constraint parameter, which can adjust the range of the impact of abnormal behaviors.

As described in (4), we can observe that when a behavior is judged as an abnormal one, its situation will also be determined by its interval time. From (2), we can observe that the normal behaviors will increase the credit score, and the abnormal behaviors will reduce the credit score. Moreover, with the passage of time, the influence of the past behavior on the score will gradually decrease.

After the behavior is recorded in the ledger, PoC rewards the corresponding SDN based on $C_S$ by

$$P_s = \lambda_r \cdot \overline{P} \cdot \left(\frac{\xi^{C_s}-\xi^{-C_s}}{\xi^{C_s}+\xi^{-C_s}}+1\right), \qquad (5)$$

where $\overline{P}$ is the preset reward of the ESN for every time finishing their work, $\xi$ is the sensitivity of the reward $P_s$ to the credit score $C_s$, and $\lambda_r$ represents the weight between the reward of ESNs and the reward of the $r$ type SDNs, which can be adjusted according to the management requirements of different types of SDNs. The concrete setting of these parameters will be discussed in Section 4.1.

As described in (2) and (5), when abnormal behaviors happened, $P_s$ will decrease timely according to the decrease of $C_s$, while when a normal behavior happened, $P_s$ will increase timely according to the increase of $C_s$. Moreover, as the value of $C_s$ increases, $P_s$ will never be higher than the upper limit, that is, $2\lambda_r\overline{P}$. A SDN with continuous abnormal behaviors in a time unit will cause the sharp decline of $C_s$. When $C_s$ is lower than the preset threshold of the system, the SDN will be considered as a malicious attacker, and it will be temporarily removed from the system.

*2.3. Data Access Control Scheme.* Since every behavior message needs to be broadcasted and transmitted many times throughout the system, it is essential to ensure the data authenticity during the communication. Moreover, behavior messages come mostly from IoT devices in smart cities, which involve the privacy of IoT users. In order to protect the data authenticity and privacy in the system, we designed the data access control scheme (DACS).

From the aforementioned framework design, we know that BEHS is built on permissioned blockchain, and every node holds a pair of asymmetric keys ($Pk$, $Sk$). The message encrypted by $Sk$ can only be decrypted by the corresponding $Pk$. The digital signature algorithm [22] uses this method to realize the directional transmission and nontampering of messages, but the messages can be exposed during transmission, and the privacy can hardly be protected.

Symmetric encryption is a lightweight scheme with high efficiency, and asymmetric encryption can provide a secure distribution way for symmetric keys [11]. Thus, we considered integrating the symmetric and asymmetric keys to encrypt privacy data and control data access in IoT devices. The encryption process of DACS can be denoted as

$$\text{encryption}(M) = \begin{cases} \text{ENC}_{Sk_s}\{\text{SHA}\{\text{ENC}_K(M)\}\}, \\ \text{ENC}_K(M), \end{cases} \tag{6}$$

where $Sk_s$ is the secret key of the sender, $K$ is the symmetric key generated by sender, ENC is the abbreviation of encryption, $\text{ENC}_{Sk_s}$ represents the encryption by $Sk_s$, $\text{ENC}_K$ represents the encryption by $K$, $M$ denotes the message to be transmitted, and $SHA()$ represents generating the summary information leveraging secure hash algorithm. The receiver can decrypt the $\text{ENC}_{Sk_s}\{\text{SHA}\{\text{ENC}_K(M)\}\}$ by the public key $Pk_s$ of the sender and verify the authenticity of the message by comparing $\text{SHA}\{\text{ENC}_K(M)\}$ with $\text{ENC}_K(M)$. The sender stores the key locally and send it to its owner. If a user wants to get the data, it needs to request the corresponding key from the owner of the data to decrypt the data. The distribution of $K$ can be denoted as

$$\text{distribution}(K) = \text{ENC}_{Sk_s}\left\{\text{ENC}_{Pk_U}\{K\}\right\}, \tag{7}$$

where $Pk_U$ is the public key of the user, and $\text{ENC}_{Pk_U}$ represents the encryption by $Pk_U$. DACS utilizes user's public key to ensure that only the target user can decrypt the correct symmetric key.

According to the aforementioned scheme, the message format is designed for the system, as listed in Table 2. Every message is directed and contains the address of sender and receiver. The type field represents the type of content contained in the message, including data uploading, currency trading, and smart contract calling. If the message contains an upload behavior, the value and contract code fields can be blank, and the uploaded data is stored in payload field. For calling the smart contract, the payload field can be blank, and for the simple currency transaction, the payload and contract code fields are both blank.

Worthy of note is that three different types of timestamps exist in our system, respectively, recorded by sensing device, edge server, and blockchain. The combination of three timestamps creates a complete timeline for each behavior message uploaded by device. This timeline has the ability to trace and review behavior messages and preclude devices from uploading duplicate data to defraud credit value and rewards. The following is a detailed explanation of three timestamps:

(1) Creation timestamp of the behavior: $t_s$. When a sensing device collects a sufficient amount of data, it will send data to the edge server in the format of behavior message.

(2) Reception timestamp of behavior message: $t_e$. If the edge server monitors a behavior message sent by a sensing device, it will record the timestamp of that moment. In the event that the edge server detects $t_s$ is too close to $t_e$ or even later than $t_e$, it will refuse to store and package this behavior message.

(3) Creation timestamp of the block: $t_b$. While cloud or edge server packs the accumulated transactions to generate a new block, the system will save the timestamp of the block generation time based on specification requirements of the block structure.

Table 2: Format of behavior message.

| Header | |
|---|---|
| Type | Type of the behavior |
| From | Sender's address |
| Order | Number of behaviors from the sender |
| To | Receiver's address |
| Value | Number of currencies |
| Timestamp | Creation time of the behavior |
| *Payload* | |
| Data$_1$, data$_2$, ..., data$_n$ | |
| *Contract code* | |
| Variable$_1$, variable$_2$,..., variable$_n$ | |

The flowchart of DACS is shown in Figure 2, which can be divided into two parts. Part 1 includes encryption, uploading, and storage of data. Part 2 includes request and distribution of symmetry key and decryption of data. The nonce is a random check code. If the receiver returns the right nonce, we will consider the receiver has decrypted the message correctly. When a sensing device submits data to an edge server, it utilizes its asymmetric keys to sign the data in payload field. The consistency of the summary and content of the data ensures the authenticity of the message. In this way, all messages in the system are traceable, and edge servers cannot tamper with behavior messages of sensing devices during broadcasting. For private data that need to be protected, the sensing device will generate a random symmetric key to encrypt the data, store the key locally and then distribute it to the administrator. Thus, the data stored in the ledger are encrypted by symmetric keys, and getting the access to the data requires obtaining the corresponding symmetric key. If a user wants to get the access of the data, he needs to request the corresponding key from the administrator to decrypt the data. If agreed, the sensing device will distribute the corresponding symmetric key by utilizing user's public key. Thus, the administrator or the node itself can protect the privacy of the data through controlling the access of the data.

## 3. System Implementation

In this section, we present the detailed implementation of the proposed BEHS, following by the evaluation of its performance.

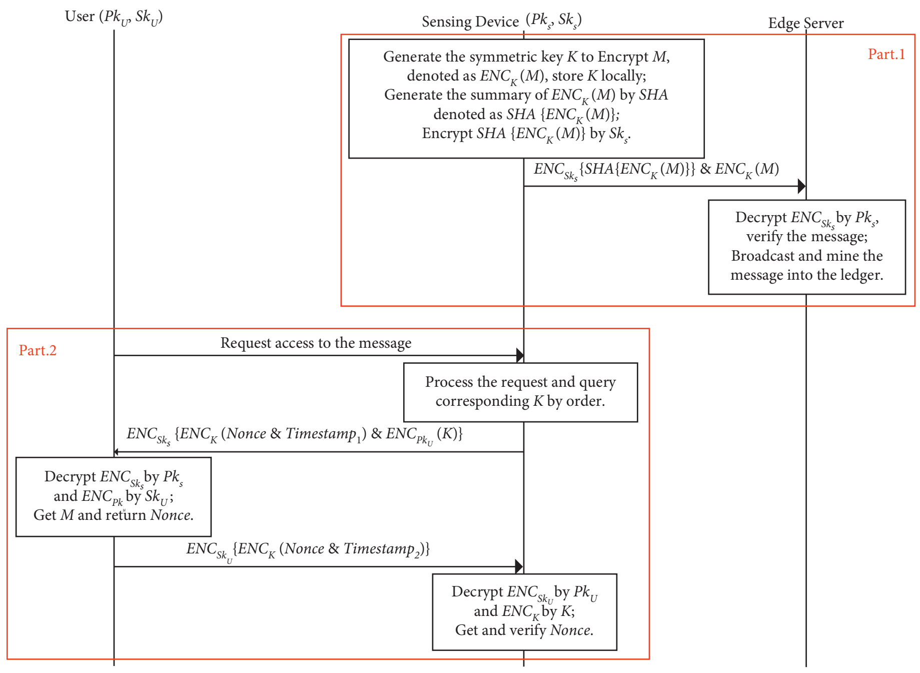
User ($Pk_U$, $Sk_U$)
Sensing Device ($Pk_s$, $Sk_s$)
Edge Server
Part.1
Generate the symmetric key $K$ to Encrypt $M$, denoted as $ENC_K(M)$, store $K$ locally; Generate the summary of $ENC_K(M)$ by $SHA$ denoted as $SHA\{ENC_K(M)\}$; Encrypt $SHA\{ENC_K(M)\}$ by $Sk_s$.
$ENC_{Sk_s}\{SHA\{ENC_K(M)\}\}$ & $ENC_K(M)$
Decrypt $ENC_{Sk_s}$ by $Pk_s$, verify the message; Broadcast and mine the message into the ledger.
Part.2
Request access to the message
Process the request and query corresponding $K$ by order.
$ENC_{Sk_s}\{ENC_K(Nonce \,\&\, Timestamp_1) \,\&\, ENC_{Pk_U}(K)\}$
Decrypt $ENC_{Sk_s}$ by $Pk_s$ and $ENC_{Pk}$ by $Sk_U$; Get $M$ and return $Nonce$.
$ENC_{Sk_U}\{ENC_K(Nonce \,\&\, Timestamp_2)\}$
Decrypt $ENC_{Sk_U}$ by $Pk_U$ and $ENC_K$ by $K$; Get and verify $Nonce$.

Figure 2: Flowchart of data access control scheme.

### 3.1. *Ethereum-Based BEHS for Smart City*

*3.1.1. Edge Server.* We installed Go-Ethereum on each of three clouds and built a consortium chain network by initializing the same configuration of the genesis.json file [23]. The configuration of the cloud is listed in Table 3, and some modules have already been embedded in Ethereum for providing interface, as summarized in Table 4. Cloud has the same status as edge server in distributed data processing, which packs transactions from the transaction pool and provides the computation support for mining new blocks. The block includes the block header and body, which will be synchronized locally to each cloud. Block generation requires a certain time interval to ensure the consistency of nodes, and the capacity of transaction payload for containing data is at most 1024 bytes. Therefore, compared with the ideal state of BEHS, the performance of Ethereum-based BEHS is significantly limited.

*3.1.2. Sensing Device.* We chose several common monitoring devices as sensing device nodes, including smoke, lampblack, and current devices. Each sensing device consists of an ARM Cortex-M3-based 32-bit processor named STM32F103VCT6 and a SIM7020 C NB-IoT module operated by China Telecom. We deploy multiple laptops as the agent to help sensing devices connect to Ethereum network. Each laptop utilizes the Geth client of Ethereum and connects the network as a light node, which only needs to store the block header and verifies blocks via Merkle Proof [6]. Each sensing device communicates with its agent through UDP protocol of NB-IoT gateway and uploads data through the RPC interface of the agent. In this case, the rewards of sensing devices cannot be directly distributed to their accounts, being held by the account of their agent. Additionally, we implemented the data access control scheme by integrating ECDSA based on the secp256k1 elliptic curve with AES symmetric encryption algorithm in each sensing device. Each piece of data uploaded from sensing devices is encrypted by a randomly generated symmetric key of 16 bytes. The key is stored locally on the device and periodically synchronized to its agent.

Table 3: Configuration of cloud.

| Attributes | Configuration |
|---|---|
| Processor | Inter(R) core(TM) i5-3470 3.20 GHz |
| Memory | 4 GB |
| OS | Windows 7 |
| Database | Oracle |
| Disk | 1 TB |

Table 4: Modules in Ethereum-based BEHS.

| Module | Technology |
|---|---|
| RPC interface | RPC APIs of Go language |
| Smart contract interface | Ethereum virtual machine |
| Application interface | Web3 protocol of JavaScript |

### 3.2. *Implementation of PoC via the Smart Contract.*

We implemented PoC on Ethereum-based BEHS via multiple smart contracts. An entry contract is the entrance for receiving data from sensing device, and a judgment contract is responsible for evaluating credit scores and rewarding contributors.

*3.2.1. Entry Contract.* A lookup table containing the identity information of sensing devices is established in entry contract, as shown in Table 5, in which each row contains the following information of each sensing device:

Table 5: Illustration of the lookup table.

| DecAddress | AgeAddress | DecType | Order | CreScore | AccAuthority |
|---|---|---|---|---|---|
| 0xs35bpjk3... | 0 x 6b6cad3c... | Smoke | 35 | 2.1 | True |
| 0xdj92j29e... | 0 x 2jk43lv4... | Lampblak | 108 | −20 | False |
| 0xd31f60gl... | 0 x 2816b60s... | Current | 0 | 0 | True |

(1) DecAddress: address of the sensing device

(2) AgeAddress: address of the agent

(3) DecType: type of the sensing device

(4) Order: number of messages from the sensing device

(5) CreScore: credit score of the sensing device

(6) AccAuthority: access authority of the sensing device

Thus, the entry contract can store the record of every behavior and the access authority of each sensing device. In addition, the entry contract provides the following application binary interfaces (ABIs) to maintain the lookup table:

deviceRegister(): this ABI receives the identity information of a new sensing device and registers its information into the lookup table.

deviceUpdate(): this ABI receives the new identity information of an existing sensing device and updates its information in the lookup table.

deviceDelete(): this ABI deletes the existing identity information of a sensing device from the lookup table.

accControl(): this ABI receives the credit score from judgment contract, controls the access authority, and updates the related information of the lookup table, especially the fields of CreScore and AccAuthority.

We noticed that only nodes with the authorized address can register, update, and delete the sensing device. The entry contract also has subData() ABI for receiving data from sensing devices; subData() ABI will call ABIs of judgment contract for judging the credit score and transact with the judgment contract to log the data.

*3.2.2. Judgment Contract.* The judgment contract implements a behavior evaluation method. When the data from a sensing device is logged in the block, judgment contract will reward the sensing device according to its credit score by transacting with its agent. A timestamp list is established for recording every behavior of each sensing device. The judgment contract provides the timUpdate() ABI for updating the list and the timQuery() ABI for querying timestamps from the list.

Based on the proposed PoC mechanism, we implemented the *creEvaluation*() ABI in judgment contract, as in Algorithm 1. It evaluates the credit score according to the inputs of the source address, type, timestamp and hash of a behavior, and returns the result. The evaluation of credit score is from lines 1 to 7 by (2), and the distribution of reward is from lines 8 to 17 by (5). The event in line 18 is used to return the results of updating the credit score and the reward distribution.

The detailed workflow of PoC mechanism is shown in Figure 3, which can be described in the following steps:

(1) Edge servers initialize the private chain based on Go-Ethereum and create account. Sensing devices and their agents create accounts by the integrated encryption scheme and connect to the blockchain network.

(2) Agents deploy entry and judgment contract on the chain, and sensing devices call entry contract to register their identity information.

(3) Then, sensing devices upload data through the RPC port of its agent and call ABIs in entry contract for recording this behavior.

(4) After that, when the behavior is packed by an edge server into a block and accepted by the system, the edge server will get the reward for mining, and judgment contract will evaluate the credit score of sensing devices and reward them.

## 4. Evaluation

In this section, we start by introducing the specific parameters setting of Ethereum-based BEHS, and then, we evaluate the performance of the system, including the effectiveness of PoC and the cost of DACS.

*4.1. Parameters Setting.* According to (2), the weight of the impact of normal and abnormal behaviors on credit score is 1 : 1, so we set $\delta_1 = 1$ and $\delta_2 = 1$. According to (3), due to the short time between the production and judgment time of each behavior message, we set $\zeta_1 = 1/2$ to constrain the impact of time on credit score. The upload period of sensing devices is set to 30 s, and $\overline{t_s}$ is thus set to 30. When $\eta$ is set to 1, the credit score of normal behavior will be fixed. In order to motivate sensing devices to provide high-quality services, we set $\eta$ to 2. The timestamps of behaviors will be stored in the list of judgment contract; thus, $\overline{t_s}$ and $K_s^N$ can be calculated. A large tolerance of abnormal behaviors is beneficial to the effectiveness evaluation of the system, so we set $\alpha_s$ to 2/3.

For a sensing device with abnormal behaviors, PoC will limit its future rewards in several cycles. The negative impact of the abnormal behavior will gradually decrease over time but cannot be erased. Therefore, the reward of node with abnormal behavior will always be less than node without abnormal behavior with the same performance. According to (4), we set the decrease rate of the influence of abnormal behavior to be the same as that of normal behavior, $\zeta_2 = \zeta_1 = 1/2$. $\kappa$ is set to 2, which adjusts the impact range of abnormal behaviors according to that of normal behaviors.

```
Input: address, type, timestamp, hash
Output: result (update, reward)
Require: result.update ← False, result.reward ← False.
Create a timestamp array timestampArray[].
timestampArray ← timQuery (address)
get Cs (address, timestampArray) using (2)
create an Entry Contract instance entry
if entry.accControl(address, Cs) is captured then
    result.update ← True
end if
while true do
    check the transaction receipt
    if the block containing the behavior is generated then
    get Ps(Cs) using (5)
    create a transaction(address, Ps)
    launch the transaction.
    result.reward ← True
    break.
        end if
end while
return result (update, reward)
```

Algorithm 1: creEvaluation() ABI.

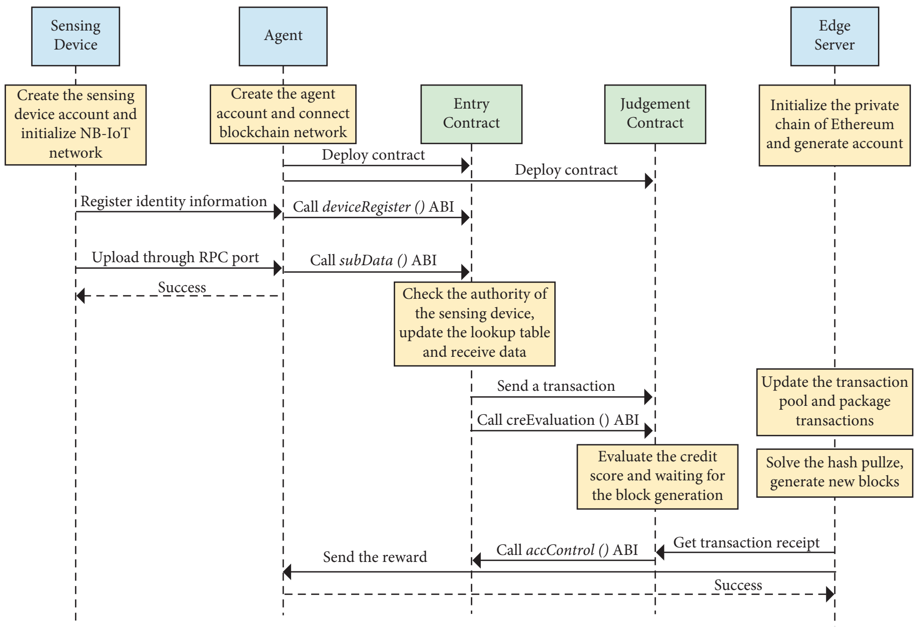
Sensing Device
Agent
Edge Server
Create the sensing device account and initialize NB-IoT network
Create the agent account and connect blockchain network
Entry Contract
Judgement Contract
Initialize the private chain of Ethereum and generate account
Deploy contract
Deploy contract
Register identity information
Call *deviceRegister ()* ABI
Upload through RPC port
Call *subData ()* ABI
Success
Check the authority of the sensing device, update the lookup table and receive data
Send a transaction
Update the transaction pool and package transactions
Call creEvaluation () ABI
Evaluate the credit score and waiting for the block generation
Solve the hash pullze, generate new blocks
Get transaction receipt
Call *accControl ()* ABI
Send the reward
Success

Figure 3: Flowchart of PoC via the smart contract.

If a larger range is desired, we can set it smaller. $K_r^A$ can also be calculated from the list in judgment contract.

Based on the previously mentioned setting, PoC can evaluate and regulate behaviors of sensing device according to its credit score. According to (5), there are three parameters $\lambda_r$, $\xi$, and $\overline{P}$. Every time a new block is mined, PoC will reward 5 Ether to the corresponding miner, so $\overline{P}$ is thus set to 5. The weight between the reward of edge server and sensing device can be adjusted according to the needs, where we set $\lambda_r = 1$ in the experiment. $\xi$ is set to 2, which is not a large sensitivity level. However, this value can be adjusted if needed.

*4.2. Effectiveness Proof-of-Contribution.* To present the effectiveness of PoC, we set sensing devices in different states for simulating different types of behaviors. Figure 4 shows the results of credit score changes based on behaviors of sensing device. The $x-$axis represents the timeline, containing multiple $\Delta t$. The period of normal behavior is set to 30 s, and the abnormal behavior is divided into the malicious behavior and lazy behavior. The period of the malicious behavior is set to 10s, and the lazy behavior will stop working for 130 s. The $y-$axis represents the value of credit score, with three curves representing normal behavior score,

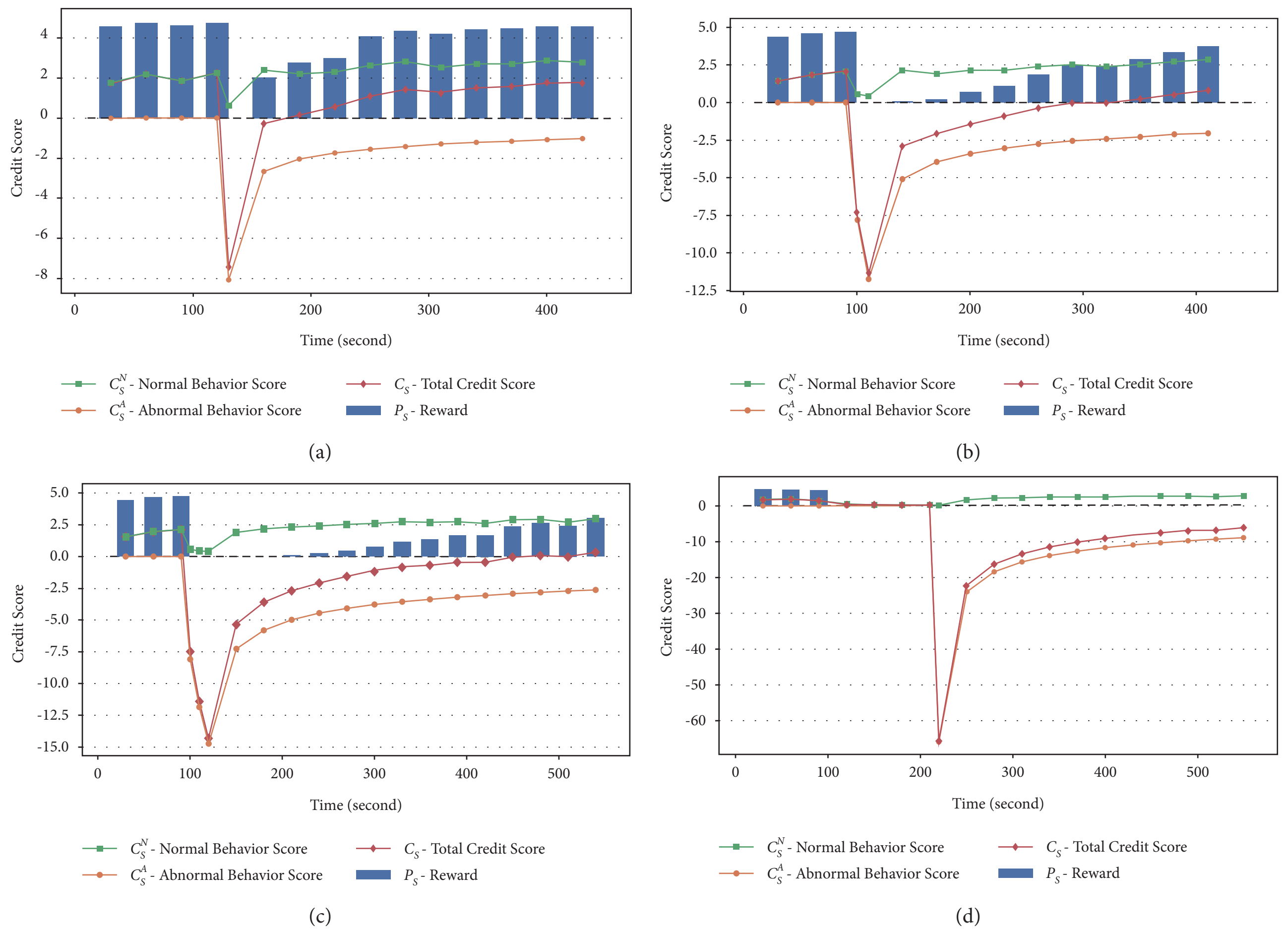

Figure 4: The credit score changes based on behaviors of sensing devices. (a) When a malicious behavior happens. (b) When malicious behaviors happen twice. (c) When malicious behaviors happen three times. (d) When a lazy behavior happens.

abnormal behavior score, and total credit score, respectively. The reward is also denoted according to the changes of the total credit score.

As can be seen from Figure 4(a), when time is at 110s, the sensing device conducts a malicious behavior. $C_s^N$ has a sharp decline according to (4), $C_s$ also has a sharp decline according to (2), and $P_s$ is decreased to 0. After the malicious behavior, the sensing device continues to behave normally. After several normal behavior periods, its total credit score gradually recovers but is lower than the average before the malicious behavior. In Figure 4(b), when the sensing device commits two consecutive malicious behaviors, it will be punished more severely. After these two malicious behaviors, the next normal behavior will lose its reward, and more normal behaviors are required to recover the total credit score. The average reward after recovery is lower than the average reward after one abnormal behavior. Certainly, the system will not endlessly tolerate a node with too much negative behavior. When a node's $C_s$ falls below the threshold at a certain moment, it will be kicked out of the network. Figure 4(c) shows the change of credit score and reward of the sensing device conducting three times consecutive malicious behaviors. In addition to the malicious behaviors not being rewarded, the next three normal behaviors will also not be rewarded. The recovery period of credit score is longer and the average reward is lower than the first two cases. As can be seen from Figure 4(d), the sensing device stops uploading data after 90 s and then starts uploading data normally at 220 s. The credit score of the sensing device continuously decreases during the shutdown. Since the system cannot capture lazy behaviors from the sensing device, no abnormal behavior score accumulates. When the sensing device resumes uploading data, the first behavior is considered as a lazy behavior, and the credit score plummets. Due to the long shutdown period, the sensing device needs to reduce the impact of this abnormal behavior through more time of normal behavior.

To further study the relationship between reward and behavior cycle, we analyzed the changes of the reward based on different uploading periods, that is, 20 s, 22 s, ..., 34 s, 36 s, and the result is shown in Figure 5. We can observe that, with the increase of the gap between the actual period and the preset period, the credit score gradually decreases. The change of reward is more sensitive to the uploading period than the change of credit score. Sensing devices cannot get more rewards by reducing their uploading period, which encourages sensing devices to complete their work more honestly.

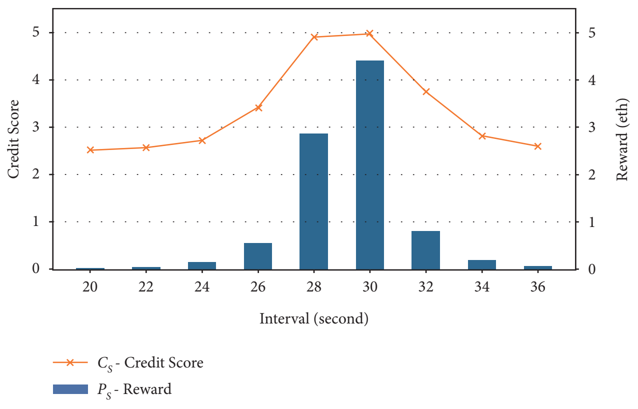

Figure 5: Reward changes based on different uploading periods.

We can conclude that the total credit score changes dynamically according to the occurrence of behaviors. When the period of upload is normal, the behavior will be considered as normal. Normal behavior score will be evaluated timely according to the accuracy of the normal behavior. The higher the accuracy is, the higher the score is. However, abnormal behaviors with a short period (e.g., 10 s) will change the abnormal behavior score, which will decrease the total credit score. The abnormal behavior of sensing device is not rewarded and affects its total credit score. Additionally, the average total credit score will also be reduced by abnormal behaviors and lead to the change of the corresponding reward. In this way, PoC can evaluate the behaviors of sensing devices, reward honest sensing devices, punish dishonest sensing devices, and clamp down malicious sensing devices. Thus, the system can effectively manage the behaviors of sensing devices, making the trustworthy IoT.

*4.3. Cost of Data Access Control Scheme.* We finally evaluated the cost of data authority management scheme running on sensing device and edge server. DACS is implemented by integrating AES and ECDSA encryption algorithm, with AES as the symmetric encryption method and ECDSA as asymmetric encryption method, and its flowchart has introduced in Section 2.3. We used secp256k1 developed by National Institute of Standards and Technology (NIST) as the elliptic curve required by ECDSA. Specifically, the prime order of the elliptic curve is set as $p = 2^{256}$-$2^{32}$-$2^{9}$-$2^{8}$-$2^{7}$-$2^{6}$-$2^{4}$-1, and hash function adopts SHA256 with ECDSA defined by ANSI X9.62 and finally realizes this algorithm in Java with Signature Class. AES algorithm adopts AES-128, which uses 10 rounds for 128-bit keys. The system clock tick of sensing device is set to 0.5 ms, and we collected the timestamp of each composition of DACS. The results are average values calculated from multiple experiments.

ECDSA is a common algorithm utilized for signature in blockchain system. DACS integrate AES with ECDSA, which increases the cost of data uploading of sensing devices to some extent. Figure 6 shows the impact of DACS on upload efficiency of sensing device. The efficiency of ECDSA is low and takes up most running time of DACS, especially

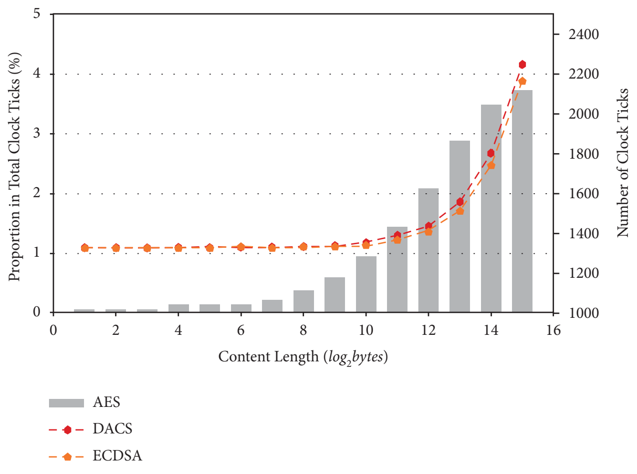

Figure 6: Impact of DACS on upload efficiency.

obvious when the content length is small. On the contrary, the efficiency of AES is very high, at a minimum of 1.2% of the total clock ticks. In smart city, IoT devices generally upload small batch data. When encrypting these data (less than $2^{13}$ bytes), AES accounts for less that 2% of total time cost. We can conclude that DACS sacrifices very little cost of time but brings high security and access control functionality.

We fixed the content length to $2^6$ *bytes* and calculate the average total time cost of DACS, as shown in Figure 7. DACS consists of four step: production of symmetric key, symmetric encryption based on AES, signature, and decryption based on ECDSA. The first three steps are completed by sensing device, and the last step is done by edge server. The efficiency of AES based symmetric key generation and encryption is very high, only taking 2 ms and 14 ms. The efficiency of ECDSA in sensing device is relatively low, taking 667 ms, and the result is not ideal, while the decryption of ECDSA in the edge server is very fast, and it only takes 9 ms. We can conclude that the time cost of DACS is within acceptable range. However, the time cost of DACS in sensing devices is high, which can be significantly improved.

DACS effectively and securely realizes the privacy protection and trusted sharing of data, but its sharing process is complicated. "One-to-one" data sharing scheme is not the best solution for high concurrent access control requirements. Therefore, in the future, we will consider introducing CP-ABE in blockchain and improving it to a specific access scheme for IoT [24]. CP-ABE relies on an access structure for encryption, and all users who satisfy the access structure can decrypt and get the plaintext message. This one-time encryption realizes the access control of multiple users, which is a feasible and promising solution in future work.

*4.4. Performance Proof-of-Contribution.* Getting the performance of PoC means we need to prove the availability of PoC mechanism towards IoT devices in reality. In addition, an appropriate consensus algorithm should be chosen as the underlying consensus protocol in this subsection, which can

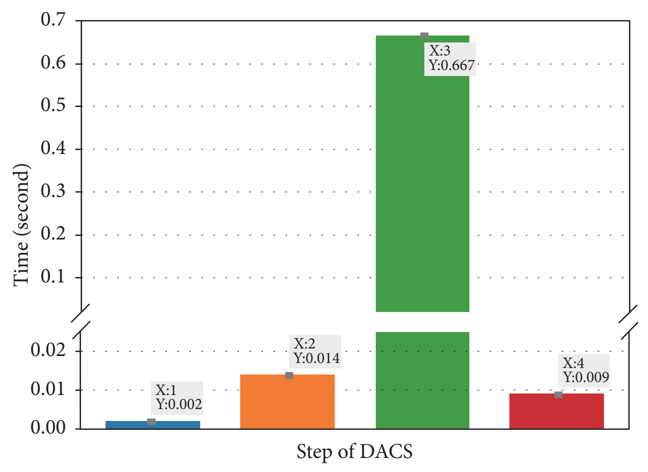

Figure 7: Total time cost of DACS. The four steps, respectively, represent production of symmetric key, symmetric encryption based on AES, signature, and decryption based on ECDSA.

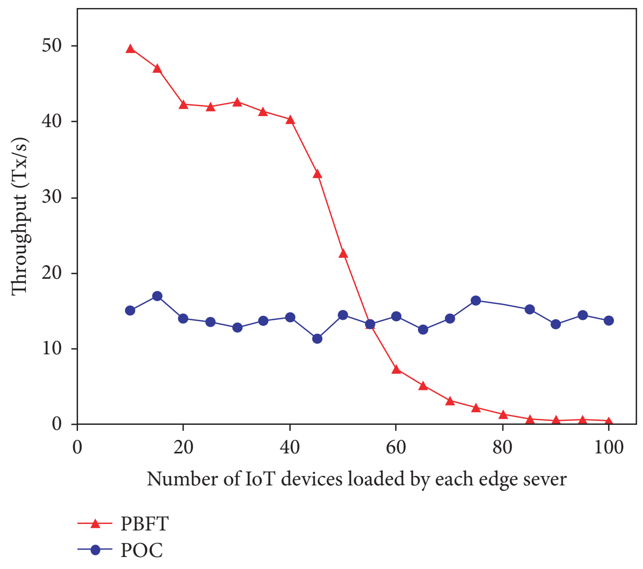

Figure 8: Comparison of throughput and load capacity between PoC based on PoW and PBFT.

be combined with PoC to serve IoT devices. Concretely, we deployed Ethereum and Hyperledger fabric on three cloud servers and configured Hyperledger Caliper [25] on one of the clouds, which is an effective performance monitoring tool for blockchain. Each cloud server will act as an edge server accessing the same number of IoT devices, and Caliper will monitor the throughput performance of PoC mechanism in blockchain systems with different underlying consensus protocols. Considering the large-scale IoT devices in practical application scenarios, we built a consortium chain in Ethereum, as in Section 3.1. This consortium chain is based on PoW consensus algorithm, and we set an adaptive mining mechanism similar to the public chain, which has the ability to adjust the difficulty of mining timely and maintain the stability of the system. Besides PoW, we think that PBFT algorithm with excellent fault tolerance is also one of the available underlying consensuses. However, it should be noted that we chose an earlier version of fabric to run the PBFT algorithm normally because of being not well implemented.

We set the frequency of IoT devices interacting with blockchain network to two requests/s and continuously increased the number of IoT devices loaded by each edge server. Figure 8 is the comparison of throughput performance between PBFT and PoC, which is plotted from the average value of 10 rounds of experiments. PBFT shows excellent performance when only a few IoT devices join the network, but it is a consensus serving the consortium chain after all. Since the communication complexity of PBFT is $O(n^2)$, a large number of IoT devices accessing the network will significantly increase the workload of message broadcasting process. Concurrently, the number of consensus nodes, server hardware, and other factors also limit the scalability of PBFT. This shortcoming is also reflected in the figure, as the number of IoT devices connected to each edge server increases continuously, the throughput of PBFT has a dramatic decrease, and eventually fails to function properly. Comparatively, although proof-of-contribution mechanism based on PoW does not have superior performance, it has a remarkable stability. PoC dynamically adjusts the difficulty of mining, and it leads to throughput performance of blockchain always maintaining at a consistent level. It also means that PoC has more practical and stable throughput when a large number of IoT devices participate in the network. The stability of PoW is what PBFT cannot do for the time being, which is the reason for choosing PoC based on PoW finally.

## 5. Conclusion

A blockchain-edge computing hybrid system is presented to provide trustworthy IoT services in smart cities. A novel proof-of-contribution consensus mechanism is proposed to regulate the behavior of nodes, especially IoT device nodes. PoC can detect and prevent abnormal behaviors realized by modifying the data upload frequency, such as greed, absenteeism, or sabotage, so as to prevent them from damaging system. A data access control scheme is proposed to secure the data authenticity, especially to protect the private data of IoT devices. We implemented the concrete system on Ethereum platform. The extensive evaluations and analyses show that the proposed PoC mechanism can effectively manage behaviors of nodes in the system, securing the system from attacks, and the cost of the designed data access control scheme is within reasonable range. However, Ethereum-based BEHS is an application prototype system, and there are still some shortcomings that need to be continuously updated and improved, such as low concurrency caused by smart contract and limited functionality of sensing device. In the future work, we will continue to explore the implementation approaches of the system, such as architecture with pluggable consensus algorithm and cross-chain communication to improve the expansibility and purity of the system.

## Data Availability

The data used to support the findings of this study are included within the article.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This work was supported by the National Science Foundation of China (61702354 and 61876121), Scientific Research Project of Suzhou University of Science and Technology (XKZ2017004), Key Laboratory of Mobile Interconnection Technology Engineering of Jiangsu Province (JSWLW2017004), Graduate Research Innovation Project (SKSJ18_012 and SJCX19_0963), and Educational Reform Project of Suzhou University of Science and Technology (SKJG18_05).

## References

[1] X. Wang, X. Chen, Z. Li, and Y. Chen, "Access delay analysis and optimization of NB-IoT based on stochastic network calculus," in *Proceedings of the 2018 IEEE International Conference on Smart Internet of Things (SmartIoT)*, pp. 23–28, IEEE, Xi'an, China, August 2018.

[2] T. Wang, H. Ke, X. Zheng, K. Wang, A. K. Sangaiah, and A. Liu, "Big data cleaning based on mobile edge computing in industrial sensor-cloud," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1321–1329, 2019.

[3] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. Ahsan Kazmi, T. N. Dang, and C. S. Hon, "Edge-computing-Enabled smart cities: a comprehensive survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10200–10232, 2020.

[4] V. A. F. Almerida, D. Doneda, and D. S. A. Jacqueline, "Cyberwarfare and digital governance," *IEEE Internet Computing*, vol. 21, no. 2, pp. 68–71, 2017.

[5] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, "On blockchain and its integration with iot. challenges and opportunities," *Future Generation Computer Systems*, vol. 88, pp. 173–190, 2018.

[6] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized Business Review*, Article ID 21260, 2008.

[7] H. Yu, Z. Yang, and R. O. Sinnott, "Decentralized big data auditing for smart city environments leveraging blockchain technology," *IEEE Access*, vol. 7, pp. 6288–6296, 2018.

[8] G. Yu, X. Zha, X. Wang et al., "Enabling attribute revocation for fine-grained access control in blockchain-iot systems," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1213–1230, 2020.

[9] P. Kumar, R. Kumar, G. P. Gupta, and R. Thirupathi, "A Distributed framework for detecting dos attacks in smart contractbased blockchainiot systems by leveraging fog computing," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 3, pp. 1–31, 2020.

[10] Y. Zhang, S. Kasahara, Y. Shen, X. Jing, and J. Wan, "Smart contract-based access control for the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1594–1605, 2018.

[11] J. Huang, L. Kong, and G. Chen, "Towards secure industrial iot: blockchain system with credit-based consensus mechanism," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3680–3689, 2019.

[12] 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 of Things Journal*, vol. 6, no. 3, pp. 4719–4732, 2018.

[13] R. Chen, I. P. Tu, K. E. Chuang, Q.-X. Lin, S.-W. Liao, and W. Liao, "Endex: Degree of mining power decentralization for proof-of-work based blockchain systems," *IEEE Network*, vol. 34, no. 6, pp. 266–271, 2020.

[14] F. Saleh, *Blockchain without Waste: Proof-Of-Stake*, Social Science Electronic Publishing, Rochester, NY, USA, 2018.

[15] S. Biswas, K. Sharif, F. Li, S. Maharjan, S. P. Mohanty, and Y. Wang, "Pobt: a light weight consensus algorithm for scalable iot business blockchain," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2343–2355, 2019.

[16] W. Viriyasitavat, L. D. Xu, and Z. Bi, "Managing qos of internet-of-things services using blockchain," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 6, pp. 1357–1368, 2019.

[17] S. Guo, Y. Dai, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing stackelberg game and double auction based task offlfloading for mobile blockchain," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5549–5561, 2020.

[18] E. K. Wang, Z. Liang, C. M. Chen, S. Kumari, and M. K. Khan, "PoR. X.: A reputation incentive scheme for blockchain consensus of IIoT," *Future Generation Computer Systems*, vol. 102, pp. 140–151, 2020.

[19] H. Song, N. Zhu, R. Xue, J. He, K. Zhang, and J. Wang, "Proof-of-Contribution consensus mechanism for blockchain and its application in intellectual property protection," *Information Processing & Management*, vol. 58, no. 3, Article ID 102507.

[20] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *Proceedings of the IEEE Thirteenth International Conference on Peer-To-Peer Computing*, Trento, Italy, April 2013.

[21] N. T. Courtois, M. Grajek, and R. Naik, *Optimizing Sha256 in Bitcoin Mining*, Springer, New York, NY, USA, 2014.

[22] A. I. Abhi and S. Y. Shin, "Bus: a blockchain-enabled data acquisition scheme with the assistance of uav swarm in internet of things," *IEEE Access*, vol. 7, no. 1, pp. 103231–103249, 2019.

[23] https://github.com/ethereum/go-ethereum. 2020.

[24] K. P. Yu, L. Tan, M. Aloqaily, H. Yang, and Y. Jararweh, "Blockchain-enhanced data sharing with traceable and direct revocation in IIoT," *IEEE transactions on industrial informatics*, vol. 17, no. 11, pp. 7669–7678, 2021.

[25] T. Wang, J. Guo, and S. Ai, "RBT: a distributed reputation system for blockchain-based peer-to-peer energy trading with fairness consideration," *Applied Energy*, vol. 295, Article ID 117056, 2021.