A Privacy-Preserving Charging Scheme for Electric Vehicles Using Blockchain and Fog Computing

Hongzhi Li, Dezhi Han, *Member, IEEE*, and Mingdong Tang, *Member, IEEE*

***Abstract*—A distributed charging system based on the Internet of Things can provide important supports to ensure the safe and sustainable operation of electric vehicles (EVs). Usually, drivers prefer to use local charging piles by querying the remote cloud server. Frequent communication with the cloud server will not only produce an unnecessary communication overhead but also increase the latency of response. More seriously, the cloud-based centralized management mode is vulnerable to cyber-attacks, which usually leads to privacy leakage. However, previous studies seldom focus on the privacy issue of the charging system for EVs. In this article, a decentralized and privacy-preserving charging scheme for EVs is proposed, which is based on blockchain and fog computing. In this scheme, fog computing is introduced to provide local computing with low latency. Specifically, a fog computing network, which is composed of fog computing nodes (FCNs), is used to provide localized services. Besides, a flexible consortium blockchain architecture is proposed. The blockchain system is deployed on the distributed FCNs, providing a decentralized and secure storage environment. By combining mutual authentication, smart contract, and blockchain-based storage, the security of privacy in the charging process can be ensured. The theoretical analysis and experiments demonstrate the advantages of the proposed scheme.**

***Index Terms*—Blockchain, fog computing, security and privacy, smart contract, vehicle charging.**

I. Introduction

ELECTRIC vehicles (EVs) have recently become an indispensable part of the modern transportation system. As there is almost no air pollution, EVs have been widely concerned as green transportation tools [1]–[3]. Due to the limitation of battery power, a large number of charging piles are needed to ensure the sustainable operation of EVs. Usually, EV charging is a location-sensitive operation, and drivers prefer to choose the charging pile close to their current locations. Generally, charging requests from EVs are analyzed and processed by cloud platforms, and several suitable charging piles will be recommended based on the geographic location of EVs. However, the terminals-cloud-based computing mode relies on the committing-waiting-return procedure, which is not so efficient for EVs charging systems. For instance, Tesla Motors, which currently offers more than 10 000 supercharging piles, provides charging services for nearly 100 000 vehicles every day [4]. Meanwhile, analyzing and processing such large-scale charging requests is a challenge for the existing centralized cloud-based scheme. Fog computing [5] is proposed to extend cloud computing's capabilities to the network edge and brings a new solution to this challenge. A fog computing network is usually composed of fog computing nodes (FCNs) distributed at the network edge. In fact, FCNs are devices with certain computing and storage capabilities that can analyze and process the received charging requests locally. Therefore, such a localization processing scheme based on fog computing can significantly reduce unnecessary network bandwidth and response time.

However, the extensive use of fog computing devices has raised concerns about data security and user privacy [6]. For instance, malicious attackers could analyze the user's geographical location to infer the user's home address, workplace, etc. More seriously, user privacy (e.g., telephone or license number, user preference, etc.) may be exposed to an untrusted party when the sensitive information is maintained on fog nodes without encryption. Furthermore, the security of the charging process based on fog computing is another key issue. Malicious attackers can sniff the communication between EVs and FCNs during charging, and obtain sensitive information through the power analysis attack [7], [8]. Therefore, FCNs cannot be trusted completely, and the security of FCNs is insufficient. For charging systems, data security and privacy-preserving are critical. However, the existing schemes [9]–[11] cannot provide reliable data and privacy protection, due to the cloud server may still crash or confronts records tampering. Nevertheless, the widespread use of blockchain brings a promising solution for privacy protection. In fact, the decentralized environment provided by the blockchain can effectively prevent the stored data from being tampered [12], [13]. From the perspective of privacy protection, a consortium blockchain is more suitable for this article. Specifically, distributed FCNs in the charging system can be organized into a reliable computing network based on the blockchain architecture. With such a blockchain-based network, the privacy and security of charing records can be ensured and the single point of failure can be avoided.

In this article, a privacy-preserving charging scheme using blockchain and fog computing is proposed. To our knowledge, this is the first work combining fog computing and the blockchain technology to improve the reliability and security of the charging system for EVs. Specifically, the main contributions of this article can be summarized as follows.

Manuscript received February 10, 2020; revised June 15, 2020 and July 2, 2020; accepted July 10, 2020. This work was supported by the China Natural Science Funding under Grant 61873160. *(Corresponding author: Dezhi Han.)*

Hongzhi Li and Dezhi Han are with the College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China (e-mail: 1071260932@qq.com; 17855009630@163.com).

Mingdong Tang is with the School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou 510420, China (e-mail: mdtang@gdufs.edu.cn).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/JSYST.2020.3009447

1) Fog computing is introduced into EVs charging systems, which can improve the efficiency and reliability of such systems compared with the traditional cloud-based mode. Furthermore, to ensure the security of the communication between EVs and FCNs, mutual authentication and encrypted communication technologies are utilized.
2) A consortium blockchain is proposed, which is composed of FCNs, can provide decentralized storage services for charging records or transactions.
3) A consensus protocol proof of online duration (PoD) is proposed, which is based on device online duration. Compared with the proof of work (PoW) consensus, PoD can significantly improve transaction verification efficiency and reduce resource consumption.
4) An experimental system based on the proposed scheme is implemented, which evaluates the performance of our scheme regarding computational overhead and communication costs.

The remainder of this article is organized as follows. In Section II, we briefly introduce the related work. In Section III, we review the preliminaries used in this article. In Section IV, we present the design of our system model. In Section V, we describe the proposed scheme in detail. In Section VI, an experimental system is designed and implemented, and the security and advantages of our scheme are analyzed. Finally, conclusions are introduced in Section VII.

II. Related Work

In this section, we survey related work in three parts. First, state-of-the-art studies on the management of EV charging are analyzed. Thereafter, some literature about fog computing are introduced. Finally, some representative applications of blockchain are surveyed.

A. Management of EVs Charging

Tian *et al.* [14] proposed a real-time charging station recommendation system based on large-scale GPS data mining. An appropriate charging station can be recommended for EVs based on historical charging events and real-time GPS data. The literature of [15] focuses on the decision-making of charging station selection and proposes a charging plan management scheme for EVs. To solve the problem that EVs cannot reach the planned charging stations on time, researchers introduced periodical updating mechanisms to adjust the planned charging plans in time. Compared with [15], a mathematical model of optimal charging strategy is presented in [16] by analyzing the charging behaviors of EVs. In addition, Tajeddini and Kebriaei [17] proposed a decentralized optimal scheduling algorithm for EVs charging. This algorithm formulates the EVs charging scheduling problem as an optimal control problem and exploits the elasticity of EV loads to fill the valleys in electric load profiles.

B. Application of Fog Computing

Basudan *et al.* [18] proposed a privacy-preserving protocol for vehicular crowdsensing-based road surface condition monitoring. Fog computing is utilized to collect and analyze the sensor data from edge devices. Furthermore, Bonomi *et al.* [19] proposed a hierarchical distributed architecture and define the role of fog computing in Internet of Things (IoT). In [20], a fog computing based framework for process monitoring and prognosis in cyber-manufacturing is proposed, which is the first framework to introduce fog computing into smart manufacturing. In [21], to address the challenges of traditional IoT-Cloud architecture brought by the growth of data analysis requirements and the proliferation of sensing devices in Industry 4.0. In this scheme, a low-complexity fog computing layer is introduced, which is between IoT devices and the cloud, and transfers computationally heavy data processing tasks from the cloud to the fog computing layer. Fog computing can be also applied in smart grids (SGs). Zahoor *et al.* [22] proposed a cloud-fog-based model for resources management in SGs. The focus of this article is to clarify the hierarchical structure of cloud and fog computing and then provide different types of computing services for SG resources management. After that, Li *et al.* [23] proposed a fog computing based vehicular carpooling scheme, which uses fog computing and the blockchain technology to improve the privacy and efficiency of carpooling applications. Also, a distributed public vehicle system is proposed in [24], fog computing is used in such a system to assist the scheduling of vehicles. However, to the best of our knowledge, few studies have applied fog computing to the charging system for EVs.

C. Blockchain-Based Privacy Protection

As a trusted distributed ledger, blockchain is proposed for secure storage and privacy protection. Some research efforts [25], [26] have devoted to demonstrating the advantages of blockchain-based medical records secure storage and sharing scheme. In [26], a privacy-preserving mechanism based on blockchain is proposed, and the blockchain is used to ensure the anonymity of user payments. In [27], to ensure the safe and efficient of road traffic, Mihelj *et al.* attempt to use smart contracts to eliminate single authority over such systems and put forward a secure and real-time traffic events detection scheme. In [28], Salah *et al.* put forward a blockchain-based traceability system for the agricultural supply chain. In [29], a blockchain-inspired IoT architecture is designed for creating a smart food supply chain. Radio frequency identification (RFID) is used to identify food and other things, the blockchain is employed to store the food-related sensitive information. Besides, Fernández-Caramés *et al.* [30] proposed a storage system, which combines the blockchain and unmanned aerial vehicles (UAVs) for inventory management. To specific, RFID tags are used as labels of inventory items and RFID readers are carried by UAVs for warehouse inventory. Finally, the collected inventory data will be stored in the blockchain for external auditing.

The proposed scheme in this article aims to improve the efficiency and security of the charging system for EVs. To our knowledge, this is the first scheme that combines fog computing and the blockchain to improve the charging of EVs.

III. Preliminaries

In order to make our scheme better understood, this section describes the necessary background and the main technologies on which the proposed scheme is based.

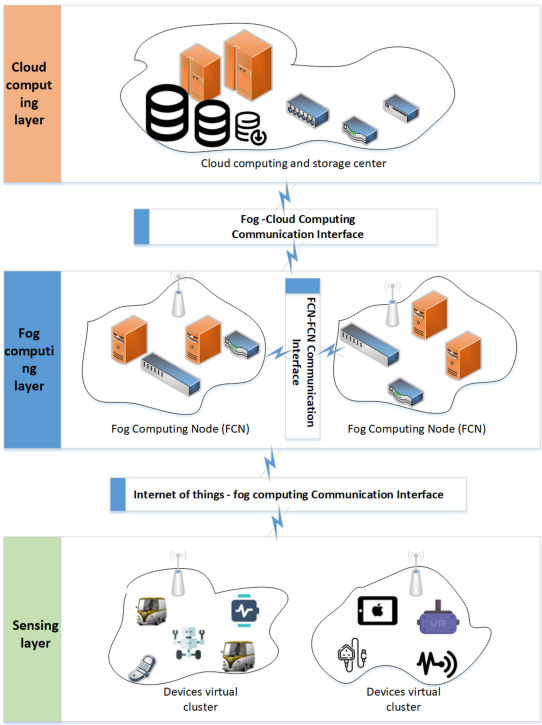

Fig. 1. Classical three-layer fog computing.

A. Bilinear Mapping

As an important method in cryptography, bilinear mapping can be constructed by Weil pairing or Tate pairing [23] in elliptic curves. Let q to be a prime number, G and G_T are two cyclic group, a bilinear mapping can be defined as follows:

$$e : G \times G \rightarrow G_T.$$

The mapping e should satisfy the following properties.

1) Bilinearity: For any $a, b \in Z_q$ and $R, S \in G$, it has $e(R^a, S^b) = e(R, S)^{ab}$.
2) Nondegeneracy: For any $R, S \in G$ exists such that $e(R, S) \neq 1_{G_T}$, where 1_{G_T} represents the unit element of G_T.
3) Computability: For any $R, S \in G$, it exists an efficient computation for $e(R, S)$.

B. Fog Computing for IoT

Fog computing has become a promising solution for IoT applications, since it can overcome the weaknesses (e.g., poor mobility, weak perception of geographical information, and high response delay) of cloud computing in the IoT environment. In fact, fog computing is based on interconnected device nodes at the edge of the network. This design makes the data and data processing closer to devices in the actual application scenario, and improves the efficiency of data processing and analysis. In general, the architecture of fog computing based applications can be divided into three layers [19]–[21]: *sensing layer*, *fog computing*, and *cloud computing* layers. Fig. 1 depicts the architecture in detail.

1) Sensing Layer: This layer is on the edge of the network and is composed of some smart devices. Specifically, these devices are usually geographically distributed with good environmental perception. All kinds of sensing data are generated from these devices in this layer.

2) Fog Computing Layer: This layer is located in the middle of the architecture and consists of geographically distributed fog servers. Fog servers are essentially virtual computing systems with information processing and storage capabilities, similar to lightweight cloud servers. Furthermore, fog servers can be connected to form a coordinated information processing network through network devices (e.g., routers, gateways, switches, access point (APs)) to handle large-scale sensing data.

3) Cloud Computing Layer: Traditional cloud servers and data centers reside on the cloud layer, ensuring adequate storage and computing resources for complex computing. Due to the limited resources of fog servers, cloud computing is needed to handle highly complex computing tasks. Such tasks should be uploaded from the fog layer to the cloud layer through high-speed wired or wireless networks.

C. Consortium Blockchain and Proof of Stake (PoS)

As a decentralized ledger, blockchain is first applied in the financial field [25]. In the blockchain, transactions will be organized into authenticated blocks distributed on different network nodes to form a chain structure. Public blockchains (e.g. Bitcoin, Ethereum) allow anyone to add new blocks using the PoW consensus.

In many application scenarios (e.g., data management and auditing within a company), the public blockchain is not an appropriate choice for protecting commercial benefit and user privacy. Generally, the consortium blockchain is more suitable in an independent enterprise or organization. Specifically, in the consortium blockchain, member nodes should be authorized and the write permission of each node is strictly restricted, and the read permission is selectively opened. The consensus algorithm in the blockchain defines the rules for member nodes to obtain the transaction packaging rights. PoW consensus is first applied in Bitcoin [31], which determines the transaction packaging rights based on the computing power of member nodes.

To solve the issue of computing resources, consumption in PoW, PoS is proposed [32], [33]. Such a consensus mechanism prevents member nodes from computing resources consumption to obtain the block packaging opportunity. It advocates obtaining the transaction packaging opportunity through the comparison of virtual assets. Therefore, PoS is suitable for consortium blockchains in specific scenarios where member nodes with measurable virtual assets. For instance, in [34], a joint PoS-based miner election method is designed, which takes the sum of trust value as stakes, and the difficulty of completing block packaging depends on the generated stakes.

IV. System Model

A. System Architecture

Our scheme is composed of a trusted authority (TA), a cloud center, FCNs, charging piles, and EVs. The architecture is depicted in Fig. 2, and the entities in our scheme are listed as follows.

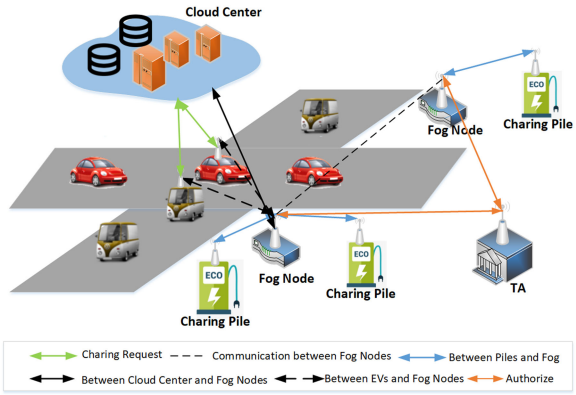

Fig. 2. System model.

TABLE I
KEY NOTATIONS

Notation	Description
k,p,G	security parameter,prime number,cyclic group
$HMAC$	keyed hash function
ENI	entity of this scheme
REQ_i	the registration request of entity i
$Cert_{ENI_i}$	certificate of the entity i
$\emptyset(.)$	filter function
EV_i	electric vehicle i
FCN_i, Pi_i	the fog computing node i, charing pile node i
$Encrypt(.)$, $Decrypt(.)$	encryption and decryption functions
SK	session key between entities
CT_x	encrypted content for x
PAu_{EV_i}	the authorization code for electric vehicle i
PoD	Proof of Online Duration
$ChargingRd_i$, Rd_i	charing records of EV_i

1) TA: As the initializer of the employed system, any entity trying to join the blockchain should register its identity information through the *TA* component. This role does not conflict with the blockchain because it only serves as a key generator and can disclose identities of malicious nodes.

2) Cloud Center: The cloud center is responsible for heavyweight and complex computing. Besides, all of the charging information could be encrypted and stored in the cloud for backup.

3) FCNs: FCNs with computing, communication and storage abilities, and can directly handle requests from EVs and charging piles. Basically, FCNs provide services such as location perception and task scheduling. In addition, the employed blockchain is deployed on FCNs.

4) Charging Piles: As the entity to complete the charging process, the charging pile is essentially an embedded computer device that provides charging service for EVs. The charging pile is located at the edge of the network and controlled by FCNs.

5) EVs: In the charging process, EVs should send encrypted charging requests to local FCNs for seeking charging piles. Therefore, EVs should be able to communicate with charging piles and FCNs.

B. Threat Model

Security threats come from both external attackers and internal malicious entities. First, the cloud center and FCNs are likely to be honest-but-curious, and thus nonfully trusted third parties, such internal entities may be able to probe into users' private and sensitive information. Besides, cyber-attacks from the external attackers should also be taken into account. For example, malicious attackers can eavesdrop on communications between entities, such as those between EVs and FCNs, and between FCNs and the cloud. Last but not least, the stored charging records in the cloud may be at risk of being tampering, lost, and attacked by sniffers.

C. Design Goals

Following the latest research efforts [15], [18], and [23], our scheme in this article is expected to achieve the following design goals.

1) *Records security and integrity:* Any sensitive information including charging records, user identities, payment records, etc., should be protected from the cloud server, FCNs, and other external entities.
2) *Privacy protection:* The private information about EVs or users (e.g., ID, name, location, and payments) should be protected from charging piles, FCNs, and cloud servers whether it is stored or transmitted.
3) *Entity authentication:* All entities in our scheme should be authenticated and no adversary can impersonate an authorized entity.
4) *Efficiency:* The communication overhead and computational costs of our scheme should be low. Also, the processing efficiency of our scheme should be sufficient to meet the needs of EVs.

V. PROPOSED SCHEME

In this section, we analyze the advantages of our scheme from two aspects in detail: secure charging process based on fog computing and blockchain-based charging records protection. In addition, the key notations used in this article are listed in Table I.

A. Secure Charging Process Based on Fog Computing

In this section, we describe the proposed secure charging process from four parts: system initialization, entity registration, mutual authorization, and EVs charging scheduling given as follows.

1) System Initialization: *TA* is responsible for system initialization. It generates an elliptic curve $E_p(a,b)$ with the prime order p, G is on the elliptic curve and selected as the base point. TA generates its public and private keys (K_{TApub}, K_{TApri}) by the elliptic curve cryptography (ECC) algorithm. A secure hash function $HMAC(key,m)$ is chosen to compute the hash of message m. Considering the importance of geographic location for fog computing, a function $\varnothing$ is used to extract location information. Finally, *TA* publishes the system parameters: $\{p, E_p, G, K_{TApub}, HMAC, \varnothing\}$.

2) Entity Registration: As entities in the system, EVs and FCNs should be verified and authorized by the TA. Besides the identity information will be encrypted and synchronized to the

cloud and blockchain. The registration process can be described as follows.

When an entity ENI_i attempts to register its information in the system, a random number k_i is selected as its *private key*, then the *public key* Q_i should be calculated as follows:

$$Q_i = k_i G. \tag{1}$$

Eid_i represents the identity information of ENI_i, and the hash of Eid_i can be calculated as $HMAC(Q_i, Eid_i) \rightarrow H_{Eid_i}$. In fact, the Eid_i should be mapped to a point $m(x, y)$ on $E_p(a, b)$ and a random number r will be selected as a parameter, then the encrypted content CT_{Eid_i} can be calculated as $\{m + rQ_i, rG\} \rightarrow CT_{Eid_i}$. Besides, the location information of ENI_i should be extracted from the environmental signals £$_i$: $l_i = \varnothing(£_i)$. Hence, we can define registration request REQ_{reg} as follows:

$$REQ_{reg} = \{Q_i, CT_{Eid_i}, H_{Eid_i}, l_i, Type, Timestamp\} \tag{2}$$

where $Type$ denotes the entity type such as "EV" and "FCN." TA should check and verify the received REQ_{reg}, and then put the registration information into storage mappings as (3) and (4)

$$SMAP_{Eid_i} = Map < Q_i \Rightarrow obj(CT_{Eid_i}, H_{Eid_i}, l_i) > \tag{3}$$

$$SMAP_{Eid_i} = Map < l_i \Rightarrow obj(CT_{Eid_i}, H_{Eid_i}, Q_i) > . \tag{4}$$

Finally, the obtained information will be synchronized to the blockchain and the cloud center. Moreover, TA should compute the hash of $SMAP_{Eid_i}$ and sign the content as $HMAC(K_{TApri}, H(SMAP_{Eid_i})) \rightarrow Sig_{TA}$. Then, the ENI_i could obtain its certificate from TA as follows:

$$Cert_{ENI_i} = \{Q_i, Issuer, Algorithm, Sig_{TA}, Date_i\} \tag{5}$$

where $Issuer$ denotes the issuer of the certificate, which commonly is the encrypted information about *TA* (e.g., IP address, Hostname, etc.). $Algorithm$ denotes the signature algorithm performed by *TA*. Furthermore, all certificates are backed up by *TA*.

3) Mutual Authorization: During the charging process, the communication between EVs and FCNs will be involved inevitably. To ensure the security of communication, a session key should be generated through mutual authorization. Suppose the EV_i attempts to establish a charging session with the FCN_i, the mutual authorization process can be described as follows.

Step 1: The EV_i sends its relevant information CT_{EV_i} to the FCN_i through a secure channel

$Encrypt(K_{TApub}, Q_{EV_i}||Cert_{EV_i}) \rightarrow CT_{EV_i}$

$CT_{EV_i} \overset{send}{\rightarrow} FCN_i$.

The FCN_i decrypts the received CT_{EV_i} and verifies the identity of EV_i with $Cert_{EV_i}$.

Step 2: The FCN_i generates the session key (this session key is used to encrypt the communication from FCN_i to EV_i) as follows:

$$SK_{FE} = k_{FCN_i} Q_{EV_i} \tag{6}$$

where k_{FCN_i} is the private key of FCN_i. Then, the FCN_i sends its responses to the EV_i as

$Encrypt(Q_{EV_i}, Q_{FCN_i}) \rightarrow CT_{Q_{FCN_i}}$

$Encrypt(Q_{EV_i}, SK_{FE}) \rightarrow CT_{SK_{FE}}$

$Encrypt(SK_{FE}, Cert_{FCN_i}) \rightarrow CT_{Cert_{FCN_i}}$.

Step 3: The EV_i obtains the received Q_{FCN_i}, SK_{FE}, and $Cert_{FCN_i}$ through the following decryption process:

$Decrypt(k_{EV_i}, CT_{Q_{FCN_i}}) \rightarrow Q_{FCN_i}$

$Decrypt(k_{EV_i}, CT_{SK_{FE}}) \rightarrow SK_{FE}$

$Decrypt(SK_{FE}, CT_{Cert_{FCN_i}}) \rightarrow Cert_{FCN_i}$.

First, the EV_i verifies whether the received $Cert_{FCN_i}$ is valid with the help of *TA*, and if the $Cert_{FCN_i}$ is a legitimate certificate, it indicates that the encryption by the SK_{FE} is valid and the SK_{FE} can be used in the session. Then, the EV_i generates its session key SK_{EF} as (7), which is used to encrypt the session from the EV_i to the FCN_i

$$SK_{EF} = k_{EV_i} Q_{FCN_i} \tag{7}$$

where k_{EV_i} is the private key of EV_i, and the EV_i uses the same way as above to verify the generated SK_{EF} by sending the encrypted $Cert_{EV_i}$ to the FCN_i. Then, the FCN_i decrypts and verifies the received $CT_{Cert_{EV_i}}$. If the $Cert_{EV_i}$ is valid, it indicates that SK_{EF} is reliable for session encryption.

In fact, SK_{FE} and SK_{EF} are equal. It is easy to see that $SK_{FE} = k_{FCN_i} Q_{EV_i} = k_{FCN_i} k_{EV_i} G = k_{EV_i} k_{FCN_i} G$. As shown in (7), we can see that $SK_{EF} = k_{EV_i} Q_{FCN_i} = k_{EV_i} k_{FCN_i} G$. In this way, we can see that SK_{EF} is equal to SK_{FE}, and then the session key SK can be defined as follows:

$$SK = SK_{FE} = SK_{EF}. \tag{8}$$

After the above mutual authorization, the generated session key SK can be used to encrypt the session communication.

4) Charging Process of EVs: When the EV_i has become a legitimate member of charging systems, a defined charging request $REQ_{charging_i} = \{Cert_i, l_i, Data_i, Hash_{req}\}$ can be sent to the cloud. After verifying the received $REQ_{charging_i}$, the cloud will recommend several appropriate FCNs for the EV_i based on the location information l_i. To be specific, we can describe the selection process of FCN_i as follows. Also, Algorithm 1 shows the selection process of FCN_i in detail.

1) The EV_i sends a defined charging request $REQ_{charging_i}$ to the cloud server $Cloud_i$ through a secure channel.
2) The $Cloud_i$ completes the verification of the received $REQ_{charging_i}$ and extracts the location of EV_i from the $REQ_{charging_i}$ as $l_{EV_i} \leftarrow REQ_{charging_i}.(l_i)$ (lines 2–6).
3) The $Cloud_i$ calculates the distance between EV_i and FCN_i as $dis_i \leftarrow calculateDis(l_{EV_i}, obj_{FCN_i}.(l_i))$ (lines 7 and 8). Considering that the communication mode between EV_i and $FCN_i \in FCNs$ is mainly wireless, and the communication is affected by the distance. Hence, we set λ as the threshold parameter of distance and the FCN_i whose distance exceeds λ will not be considered (lines 9 and 10).
4) The $Cloud_i$ obtains the load of each $FCN_i \in FCNs$ and calculates the recommended coefficient W_i based on dis_i and $load_i$ as $W_i = \omega_1 * dis_i + \omega_2 * FCN_i.(load)$ (lines 12 and 13).
5) Finally, the FCN_i with the minimum W_i will be selected as the FCN for the EV_i (lines 16 and 17).

When the FCN_i is chosen by the EV_i as its FCN, a session key SK will be generated based on the mutual authorization

Algorithm 1: Selection of FCN_i.

Input: $REQ_{charging_i}$ denotes the charging request from EV_i,
ω_1 represents the weight of the distance to EV_i,
ω_2 represents the weight of FCN load,
λ is the threshold for the distance to EV_i.
Output: Object.

$REQ_{charging_i} \overset{send}{\rightarrow} Cloud_i$;
if verify($REQ_{charging_i}$) == false **then**
  return Null;
end if
$FCNs \leftarrow \{FCN_1, FCN_2, \ldots, FCN_n\}$;
$l_{EV_i} \leftarrow REQ_{charging_i}.(l_i)$;
for FCN_i in $FCNs$ **do**
  $dis_i = calculateDis(l_{EV_i}, FCN_i.(l_i))$;
  if $dis_i > \lambda$ **then**
    continue;
  else
  % Calculate the comprehensive score of FCN_i
    $W[i] = \omega_1 * dis_i + \omega_2 * FCN_i.(load)$;
  end if
end for
$Index \leftarrow getIndexOfMinW(W[1, 2, 3, .., n])$;
$FCN_i \leftarrow FCNs[Index]$;
return FCN_i;

mentioned above. The selected FCN_i is responsible for all interactions with the EV_i during the charging process. In fact, the FCN_i should recommend suitable charging piles for the EV_i based on the received $REQ_{charging_i}$. Here, there are about three kinds of scheduling strategies that can be adopted in charging systems.

1) *Nearest distance-based scheduling*: The distance of the EV_i to each charging pile is computed. Then, the nearest charging pile Pi_i to the EV_i should be chosen. However, this approach just takes into account the driving cost to the Pi_i, but the load of the target charging pile Pi_i is not considered. In fact, the load of the Pi_i determines the charging waiting time of the EV_i.
2) *Waiting time-based scheduling*: The waiting time of each charging pile is calculated, based on which the charging pile Pi_i with the minimum waiting time will be selected, which means that the Pi_i with fewer EVs queued as an option. Suppose the average charging time of EVs is t_c, the number of vehicles waiting to be charged is n_w. In theory, the minimum waiting time is $(n_w - 1)t_c$.
3) *Comprehensive time-based scheduling*: The comprehensive cost time t_{com} consists of the driving time and the waiting time. It is easy to see that the t_{com} is determined by the distance to the charging pile Pi_i and the load of Pi_i. To specific, the comprehensive cost time of each charging pile should be calculated, and the charging pile with the minimum comprehensive cost time should be selected.

Obviously, it is more reasonable to choose the charging pile according to the comprehensive time-based scheduling, especially the system load is not heavy. But note that this approach is not suitable for all situations. For instance, if the remaining power of EV_i is insufficient, then the nearest distance-based scheduling may be more reasonable.

Finally, the Pi_i will be chosen for the EV_i as an appropriate charging pile, and the EV_i will receive an authorization code $PAu_{EV_i} = \{AuCode, expiration, timestamp, Sig_{FCN_i}\}$ from the FCN_i. Meanwhile, the Pi_i will receive the same authorization code packet PAu_{Pi_i}, and $PAu_{EV_i} = PAu_{Pi_i}$. Then, the Pi_i should verify the EV_i in the charging process as follows.

1) The EV_i sends the received packet PAu_{EV_i} to the Pi_i.
2) Then, the Pi_i will check $HMAC(PAu_{EV_i}, Q_{FCN_i}) \overset{?}{=} PAu_{EV_i}.(Sig_{FCN_i})$.
3) If passing the above verification, the Pi_i should check whether $PAu_{EV_i}.(timestamp) + PAu_{EV_i}.(expiration) \leq timestamp(now)$.
4) Finally, the Pi_i will check $PAu_{EV_i}.(AuCode) \overset{?}{=} PAu_{Pi_i}.(AuCode)$.

Once the EV_i completes its charging process with the Pi_i, then all charging records (e.g., payments, consumption of electricity, location information, etc.) will be aggregated and uploaded to the FCN_i.

TABLE II
DEFINITION OF SENSITIVE DATA

Type	Descriptions
Payment data	User's payment information, including account, payment amount, payment time, etc.
Charging records	The record of EV charging, including the duration of charging, the amount of electricity consumed, the number of charging piles, etc.
Location information	Location information, including the location of the chosen charging pile, the location at which the EV requests charging.

B. Blockchain-Based Charging Records Protection

In our scheme, the proposed blockchain is deployed on FCNs. In fact, the computing and storage resources required for the blockchain are provided by FCNs. In the following, we will describe the working flow of the blockchain in detail, including the definition of sensitive data, the storage process, and the PoD consensus.

1) Definition of Sensitive Data: Unlike cryptocurrency systems, a charging system may generate more than hundreds of GB records every day. The blockchain may be not able to store all of the charging records due to the limited storage and computing resources of FCNs. In fact, the storage and computing resources are typically limited for an embedded computer. Therefore, it is unacceptable to store all of the charging records in the blockchain. To address this issue, we specify that only sensitive data can be stored in the blockchain. We use ξ to indicate the sensitive data type, and the sensitive data are defined in Table II.

Meanwhile, only the hash results of insensitive information are stored in the blockchain, and the original data should be encrypted and stored in the cloud.

2) Blockchain Storage: In our scheme, Hyperledger Fabric 1.3 is employed as the implementation platform, and PoD is adopted as the basic consensus algorithm. As shown in Fig. 3, the

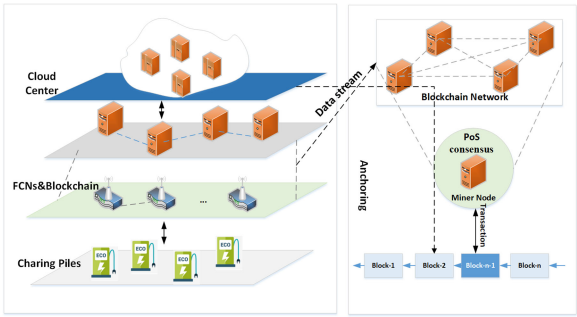

Fig. 3. Workflow of storage process.

charging records should be transmitted to the fog computing network through a secure channel. Then, the consortium blockchain deployed on FCNs is responsible for the secure storage of sensitive information, whereas the remote cloud center provides computing and storage assistance for FCNs. Here, we make full use of the consortium blockchain and the cloud to achieve the secure storage of charging records. When the EV_i completes its charging process, the FCN_i will obtain the charging record $ChargingRd_i$, and the $ChargingRd_i$ will be processed as follows.

1) Analyze whether $ChargingRd_i$ is sensitive data

$$Z_{ChargingRd_i} = \begin{cases} 1, ChargingRd_i \in \xi \\ 0, ChargingRd_i \notin \xi. \end{cases}$$

2) Compute the hash: $HMAC(k_{FCN_i}, ChargingRd_i) \rightarrow Hash_{Charging_i}$.
3) Encrypt the $ChargingRd_i$ by the public key of FCN_i: $Encrypt(Q_{FCN_i}, ChargingRd_i) \rightarrow CT_{ChargingRd_i}$.
4) If $Z_{ChargingRd_i} = 1$, then the $Hash_{Charging_i}$ and $CT_{ChargingRd_i}$ will be uploaded to the blockchain. If $Z_{ChargingRd_i} = 0$, then the $Hash_{Charging_i}$ will be uploaded to the blockchain, $CT_{ChargingRd_i}$ will be uploaded to the cloud.
5) It is important to note that if we just need to upload the $Hash_{Charging_i}$ to the blockchain as a transaction (i.e., $Z_{ChargingRd_i} = 0$), then we will get the transaction id as $TransId_{Charing_i}$, therefore, we should upload the defined $< TransId_{Charing_i} \Rightarrow CT_{ChargingRd_i} >$ to the cloud as a complete record.

In this article, to achieve scalable data storage, a smart contract named *TransStorage* is defined and deployed in the blockchain. Such a smart contract is usually called by FCNs. In detail, the mapping relationship between EVs and their charging records *ChargingRd* should be defined in *TransStorage*, then a mapping type variable $EVChargingMap = Map < Q_{EV_i} \Rightarrow ChargingRd[] >$ is used to store the relationship. Here, we can define the structure of *ChargingRd* used in *TransStorage* as follows:

typedefine Struct *ChargingRd* {
Cert : String,
RdHash: String,
RdContent: JavaScript object notation (JSON),
Timestamp: Long
}.

Algorithm 2: TransStorage($Cert_{EV_i}, Rd_i, Addr_{FCN_i}$).

Input: $Cert_{EV_i}$ denotes the identity of the EV_i,
Rd_i denotes one of charging records,
$Addr_{FCN_i}$ denotes the blockaddress of FCN_i.
Output: boolean.
1: **if** $msg.sender$!= $Addr_{FCN_i}$ **then**
2: **return** false;
3: **end if**
4: **if** $Cert_{EV_i}$ is NULL **OR** Rd_i is NULL **then**
5: **return** false;
6: **end if**
7: $Hash(Cert_{EV_i}) \rightarrow Index$;
% The hash of $Cert_{EV_i}$ as the retrieval index.
8: **if** empty($EVChargingMap[Index]$) **then**
9: $ArrayofChargingRd$ is a empty array of *ChargingRd*;
10: $EVChargingMap[Index] = ArrayofChargingRd$;
11: **end if**
12: CRd_i=$Encrypt(Rd_i, Addr_{FCN_i}||Cert_{EV_i})$;
13: $EVChargingMap[Index].add(CRd_i)$;
14: **return** true;

In the defined *ChargingRd*, the *RdContent* field is with the unstructured JSON type, which is widely used for storing unstructured information. The encrypted charging records are stored in this filed. The *RdHash* field is the hash value of the *RdContent*, which can be used to verify the integrity of the stored charging record. Algorithm 2 presents the functionality of *TransStorage*.

3) Proof of Online Duration: A consensus protocol is the core of blockchain. As we know, the famous PoW consensus is widely used in public blockchains (e.g., Bitcoin, Ethereum, etc.). However, the PoW consensus algorithm usually leads to unnecessary cost of computing resources, therefore, the existing public consensus protocols are not suitable for IoT systems, since an embedded computer may be unable to perform well enough with its limited computing and storage resources. In our scheme, we propose a consensus protocol based on the device's online duration, which utilizes the PoS consensus mechanism.

Based on the above considerations, it is reasonable to suppose that the device with longer online duration, higher reliability, and better-operating condition is more suitable to become a block packaging node. Thereby, we can see that the "online duration" of the device is equivalent to "Stake" in the PoS consensus.

To make our consensus algorithm better understood. We assume $FCNs = \{FCN_1, FCN_2, \ldots, FCN_n\}$ represents all nodes that compete for the bookkeeping rights at time t. Moreover, $D = \{d_1, d_2, \ldots, d_n\}$ denotes their respective online duration. Thus, the node FCN_i with the maximum online duration d_i will obtain the bookkeeping rights of the generated block b. To avoid the bookkeeping opportunity being monopolized by a specific node, if the FCN_i has completed the packaging of block b, then the online duration d_i will be subtracted by a random number $c(0 < c < d_i)$ according to probability $p(0 < p < 1)$. After that, the online duration of FCN_i is reaccumulated based

Algorithm 3: Proof of Online Duration.

Input: $nodes$ denotes the node information,
$p \in (0, 1)$ denotes the probability,
tx_i is the transaction that needs to be packaged.
Output: boolean.
$map(fcn_1 \Rightarrow d_1, fcn_2 \Rightarrow d_2, \ldots, fcn_n \Rightarrow d_n) \rightarrow nodes$;
$max \leftarrow 0, c \leftarrow 0, tmpIndex \leftarrow Array, selectedIndex$;
for $j = 1$; $j \leq$ nodes.length; j++ **do**
  if $(nodes[j].d \geq max)$:
    $max \leftarrow nodes[j].d$;
end for
for $j = 1$; $j \leq$ nodes.length; j++ **do**
  if $(nodes[j].d == max)$:
    $tmpIndex.append(j)$;
end for
if (Size($tmpIndex$) > 1) :
  for $j = 0$; $j <$ tmpIndex.length; j++ **do**
    if ($Hash(nodes[tmpIndex[j]])$ is maxStr) :
      $selectedIndex \leftarrow tmpIndex[j]$;
  end for
else: $selectedIndex \leftarrow tmpIndex[0]$;
Math.Random(0, max) $\rightarrow randomNum$;
$nodes[selectedIndex].d \xleftarrow{p} (nodes[selectedIndex].d - randomNum)$;
$nodes[selectedIndex] \rightarrow minerNode$;
$minerNode.generateBlock(tx_i) \rightarrow block_i$;
for $j = 1$; $j \leq$ nodes.length; j++ **do**
  if $(nodes[j].check(block_i))$:
    $c++$;
end for
if $(c/nodes.length > 0.51)$:
  $minerNode.addChain(block_i)$;
return true;

on this value. The election process of the bookkeeping node can be described as follows.

1) Each participating node FCN_i sends its online duration d_i to other competing nodes.
2) After Δt, all competing nodes complete information exchange and achieve a unique view about online duration as $(FCN_1 \Rightarrow d_1, FCN_2 \Rightarrow d_2, \ldots, FCN_n \Rightarrow d_n)$.
3) Each node selects the node with the maximum online duration as FCN_{target}. If there is more than one node with the maximum online duration, then a hash algorithm can be used to select FCN_{target} among these nodes with the same maximum online duration.
4) Each node sends a confirmation message with FCN_{target} to other nodes, and also receives confirmation messages from other nodes.
5) Each node FCN_i will check whether it is the FCN_{target} based on the received confirmation messages. If it is determined that it is the bookkeeping node of the election process, it will perform the responsibility of the bookkeeping node.

The simplified process of PoD is shown in Algorithm 3.

TABLE III
SPECIFICATIONS OF DEVICES

Devices	CPU	OS	RAM	Disk
Lenovo M720	Intel Core i7-7500 2.7GHz	Windows 10 (64 bit)	32GB	256GB
Raspberry Pi4 Model	A72 1.5GHz quad-core 64-bit	Ubuntu MATE 16.04	2GB	32GB
Meizu V8	MT6739	Android 7.1.1	3GB	32GB

VI. IMPLEMENTATION AND PERFORMANCE EVALUATION

In this section, we first introduce the experimental environment and the trial system based on the proposed scheme, then present the security analysis of our scheme. Finally, we analyze the performance of our scheme through experimental results.

A. Experimental Environment and Trial System

Based on the proposed scheme, a trial system is developed and implemented. The system consists of a cloud center, distributed FCNs, and several EV terminals simulated by Android clients. All fog nodes form a consortium blockchain based on Hyperledger Fabric 1.3. To implement the trial system, Alibaba Cloud platform is chosen to deploy our cloud applications, a desktop computer (Lenovo ThinkCentre M720) is used to run the *TA* procedure, and four single-board computers (Raspberry Pi 4 Model) are used as FCNs. Besides, we use five Android devices to simulate the terminals of EVs and charging piles. The specifications of these devices are described in Table III.

B. Security Analysis

In this section, we evaluate the security of our scheme based on the design goals defined in Section IV.

1) Security and Integrity of Records: First of all, all records in our scheme are encrypted, and only the record owner ENI_i has the decryption key k_i, so that other entities cannot sniff any sensitive information without k_i. Furthermore, all records or hash results are stored on the consortium blockchain. The decentralized environment provided by the blockchain ensures that the stored records cannot be tampered by any malicious entity.

2) Privacy Protection: The identity information of registered entities (e.g., ID, name, location) is encrypted by *TA* and stored in the cloud and blockchain. For the behavior records (i.e., payment and charging records) of entities, the contained sensitive information is encrypted and stored in the blockchain. Furthermore, these records are completely backed up to the cloud. Thus, if the private key of FCN_i is stored securely, then the privacy is secure enough.

3) Entity Authentication: We assume that an adversary ENI_a attempts to impersonate a legitimate entity EV_i. Then, ENI_a should participate in the session between EV_i and a fog computing node FCN_i, but the session is encrypted by the key SK. As mentioned above, the generation of SK requires the private keys both of k_{EV_i} and k_{FCN_i}. We assume that the private keys k_{EV_i} and k_{FCN_i} are stored securely, and the computing power of ENI_a is limited. In this case, ENI_a cannot obtain

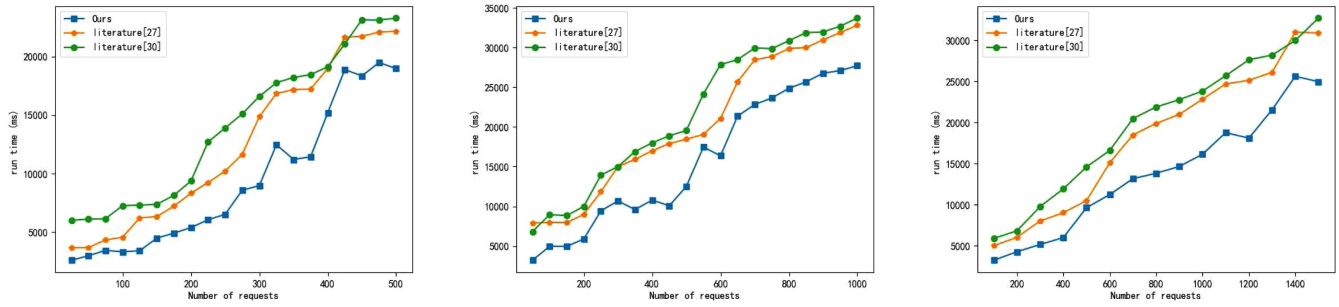

Fig. 4. Comparison of computational costs. (a) Execution time with maximum 50 threads. (b) Execution time with maximum 100 threads. (c) Execution time with maximum 150 threads.

any sensitive information from the session between EV_i and FCN_i.

4) Session Key Disclosure: The session key SK is used to encrypt and decrypt the session between EV_i and FCN_i. Therefore, the security of SK determines the security of the session. An adversary ENI_a cannot generate the valid session key $SK = k_{EV_i} k_{FCN_i} G$ because ENI_a cannot obtain the private keys k_{EV_i} and k_{FCN_i}. Moreover, the SK is not transmitted through the network, and there is no possibility of the SK being leaked.

5) Security Analysis of TA: TA component in our scheme is composed of several TA servers, which are backup to each other. Generally, only one server is online at the same time. These servers provide services to external requests in turn according to their respective time slices. Therefore, the design of TA component can effectively avoid the single point of failure bottleneck. As we know, the identity information m can be encrypted as $\{m + rQ_i, rG\} \rightarrow CT_{Eid_i}$, and the decryption process can be described as $m + rQ_i - k_i rG = m$, where k_i is the private key of ENI_i, and Q_i is the public key that can be calculated as $Q_i = k_i G$. Hence, even if the TA is compromised, the privacy of ENI_i cannot be obtained without the private key k_i.

C. Performance Evaluation

In this section, we evaluate the performance of the proposed scheme in terms of computational overhead and communication costs. In addition, the performance of our PoD consensus is analyzed.

1) Computational Overhead: For the evaluation of computational overhead, we mainly consider the performance of FCNs, because these nodes undertake a large amount of computing work in the whole system. The proposed evaluation test is based on the Apache JMeter 5.2. We simulated 500, 1000, 1500 requests to one FCN node by setting the number of test threads as 50, 100, 150, and for each thread, we set the loop count as 10. We compare the computational costs of the proposed scheme with that of [27] and [30]. The results are listed in Fig. 4(a)–(c).

In this experiment, to compare the performance of all involved schemes fairly, the cost of the scheduling process is not considered, since the literature [27] and [30] mainly focus on authentication and security issues in data storage. From the results in Fig. 4, we can see that our scheme performs better than the other two schemes. In general, the computational cost of our scheme is at a low level and remains stable. Specifically, when requests reach 300 (the total number of threads is 50), the average run time of each request is 21.88 ms. While the total number of threads is 150 and requests reach 1200, the average run time of each request is 15.23 ms. Therefore, the performance of our scheme is relatively stable.

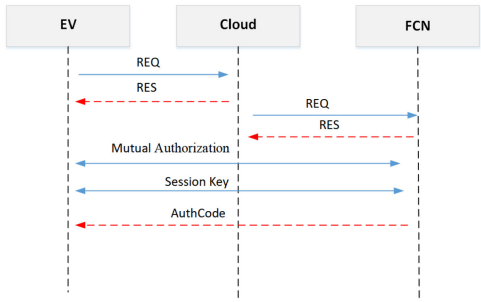

Fig. 5. Simplified flowchart.

2) Communication Overhead: For the evaluation of communication overhead, we consider the main entities involved in the charging process, and the simplified flowchart of communication between entities is shown in Fig. 5. Specifically, the communication process can be divided into charging request submission, session key generation, and charging verification stages.

Charging request submission: EV_i sends a charging request $REQ_{charging}$ to the cloud, and the received $REQ_{charging}$ will be forwarded to an appropriate FCN_i. After that, the FCN_i will verify and analyze the received $REQ_{charging}$ and sends a response $RES_{charging}$ to the EV_i. Hence, the communication cost of this stage is $C_{sub\_req} = 2\{|REQ_{charging}| + |RES_{charging}|\}$.

Session key generation: In this stage, the communication participants need to transmit public keys to each other. Here, we can briefly describe the process as follows:

$EV_i \rightarrow FCN_i : Q_{EV_i}; FCN_i : SK_{FE} = k_{FCN_i} Q_{EV_i}$

$FCN_i \rightarrow EV_i : Q_{FCN_i}; EV_i : SK_{EF} = k_{EV_i} Q_{FCN_i}$.

Charging verification: In this stage, EV_i will receive an authorization code packet, and the generated PAu_{EV_i} will be also sent to a charging pile Pi_i for charging verification. Therefore, the communication cost is $C_{\text{verify\_code}} = 2\{|PAu_{EV_i}| + |RES_{Au_i}|\}$. Overall, the total

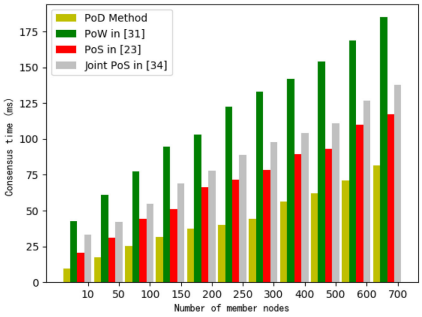

Fig. 6. Comparisons with other consensuses.

communication overhead of a charging request can be defined as $C_{\text{total}} = 2\{|REQ_{\text{charging}}| + |RES_{\text{charging}}|\} + \{|Q_{FCN_i}| + |Q_{EV_i}|\} + 2\{|PAu_{EV_i}| + |\text{RES}_{\text{Au}_i}|\}$, and the communication overhead is proportional to the number of charging requests. Therefore, the communication cost corresponding to N charging requests is $N * C_{total}$, and then the communication complexity can be defined as $O(N * C_{total})$.

3) Performance of the PoD Consensus: To evaluate the performance of PoD in detail, we have designed and implemented a comparison test with other consensuses (i.e., PoW in [31], PoS in [23], and PoS in [34]). In detail, for the PoW in [31], the SHA-256 algorithm is used, and we specify that the node that first obtains the target hash value is the packaging node. Here, we define the hash string starting with "00" as the target string (the difficult setting is 2). For our PoD consensus, random numbers from 0 to 1000 are used to simulate the online duration. Both the PoS in [23] and the PoS in [34] are used in IoT scenarios; the PoS in [23] uses the proportion of records held by devices to calculate stakes, whereas the PoS in [34] takes the sum of trust value as stakes and the difficulty of completing the block packaging depends on these stakes. From the results in Fig. 6 (here, the *Y*-axis is the average run time to achieve a consensus in the corresponding network), we can see that our PoD consensus performs better than other consensuses. Specifically, we have simulated a network of nodes from 10 to 700, the consensus time of PoD is still about 90 ms. Meanwhile, the PoW in [31] is significantly more time-consuming than our PoD. This is because our PoD consensus does not have a comparison of computing power. Since the PoS in [34] is actually a combination of PoW and traditional PoS, which still retains some hash operations. Therefore, compared with the PoS [23], the PoS [34] has better decentralization characteristics, but the consensus efficiency is slightly lower. Furthermore, to evaluate the resources consumption of PoD, we have designed and implemented a comparison experiment with the PoW in [31] and the PoS in [23]. The experiment is built on a Raspberry Pi4 computer with quad-core A72 1.5 GHz processor and 4 GB memory.

In the experiment, we evaluate the resource consumption of algorithms in terms of memory consumption and CPU usage. Besides, VisualVM 1.4.4 is used as a measurement tool to monitor memory and CPU usage. As shown in Table IV, we can see that PoD performs better than the other two consensuses; the memory consumption of PoD is less than 0.5 MB, whereas the PoW [31], the maximum memory usage is more than 15 MB.

TABLE IV
COMPARISON OF RESOURCES CONSUMPTION

Consensus	Virtual nodes	MEM(MB)	CPU(%)
PoD	100	0.15	1.4
	200	0.32	2.1
	300	0.43	2.4
PoW in Literature [31]	100	5.51	4.7
	200	7.46	11.4
	300	16.2	17.5
PoS in Literature [23]	100	0.54	1.6
	200	1.21	2.4
	300	1.87	3.6

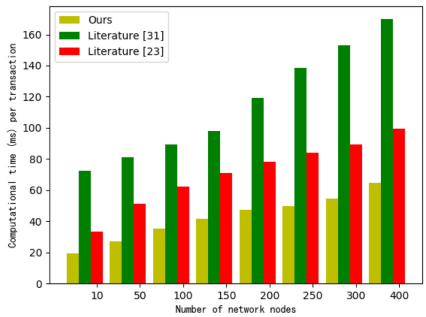

Fig. 7. Runtime comparisons with [23] and [31].

In terms of CPU usage, PoD is significantly lower than PoW. Moreover, the highest CPU usage of PoD is 2.4% (the number of virtual nodes is 300), but lower than the lowest value of PoW (when the number of virtual nodes is 100, the CPU usage is 4.7%). Since the PoD is essentially based on the PoS mechanism, which is similar to the consensus adopted in [23]. Then, these two methods are relatively close in resources consumption. However, the PoD utilizes the online duration of devices as stakes, which reduces the virtual asset calculation compared with the PoS in [23]. Therefore, as shown in Table IV, the PoD performs better than the PoS in [23]. To further compare computational costs of different consensus-based blockchain schemes. We design an experiment to evaluate computational costs by simulating multinode networks. Note that we use the PoW-based Ethereum instead of the Bitcoin blockchain in [31].

As shown in Fig. 7, with the same network scale, our scheme has obvious advantages in computational costs compared with [23] and [31]. To specific, we simulate the network with the node size ranging from 10 to 400 (docker and docker-composer are used) and evaluate the average processing time of each transaction. Since our scheme is based on the PoD consensus, there is no large number of hash operations to consume computing resources compared with the PoW-based Ethereum. Meanwhile, compared with the scheme in [23] based on the traditional PoS (i.e., the probability of being a miner node is determined by the proportion of stakes held by nodes), the PoD directly uses the *online duration* provided by operating systems without a tedious calculation process of stake proportion.

VII. CONCLUSION

In this article, we proposed a privacy-preserving charging scheme using blockchain and fog computing. The introduction of fog computing can effectively solve the problem of server-side

overload in the cloud model. Also, the mutual authentication mechanism ensures the security of communication between EVs and FCNs. In terms of privacy protection, a consortium blockchain based on Hyperledger Fabric is proposed, which is deployed on FCNs and make full use of resources provided by FCNs. Furthermore, a consensus mechanism for our blockchain is proposed, which is based on the online duration. Analysis and evaluation show that the proposed scheme is safe and effective.

APPENDIX

A. Resistance of Consensus Attacks

1) Sybil Attack: Sybil attack is commonly used to attack a P2P network, which allows a malicious participant subverts to a peer-to-peer network by creating a large number of pseudonymous identities in order to appear and function as multiple distinct nodes. In a public/permissionless blockchain, this kind of attack can trivially break the voting-based protocol. However, in our scheme, if a node attempts to join the blockchain network, it should provide its certificate. Such a certificate is issued by the TA after the node completes its registration, which contains its unique official authorization. That is to say, a blockchain node needs to provide a unique certificate issued by the TA to participate in the consensus process. Different from one-CPU-one-vote where the majority could be subverted by anyone able to allocate many CPUs, in our PoD, entity nodes maintain a one-host-one-certificate relationship, and in the consensus process, a valid certificate can only be held by one node. Therefore, the proposed PoD can prevent the Sybil attack from malicious participants.

2) Selfish Mining: Selfish mining is a strategy for a miner to keep its discovered blocks private, thereby intentionally forking the chain, to obtain more revenue than its ratio of the total mining power. However, this kind of attack strategy does not benefit a lot in PoD. Unlike Bitcoin or Ethereum, which solves fork via the longest chain, our PoD immediately solves the blockchain fork problem by restricting that only one miner node can complete block packaging, even if there are other nodes with the conditions (i.e., more than one node has the longest online duration). Theoretically, the PoD algorithm will not generate any blockchain fork, so the longest chain principle is not required. Besides, a selfish miner will get nothing from its withholding private blocks, only wasting computing resources.

3) Double Spend Attack: The double-spend attack refers to rolling back the transactions that have been confirmed so that tokens paid in previous transactions can be used again. When it comes to the token-less blockchain, the double-spend attack refers to deleting previously stored records. To achieve the double-spend attack, an attacker tries to generate an alternate chain, which does not contain several records that the attacker wants to erase. According to the longest chain principle, the blockchain tampering can be achieved when the alternate chain becomes the longest chain. In fact, the basis of double-spend attack lies in the blockchain fork mechanism and the longest chain principle. However, in our PoD, it is impossible to generate legal blockchain forks, and the longest chain principle is not required. Therefore, it is impossible to realize a double-spend attack in our scheme.

B. Time and Space Complexity Analysis

As shown in Algorithm 3, the core of the PoD algorithm is mainly composed of the maximum *online duration* searching (lines 3–6), the generation of alternative miner nodes (lines 7–9), and the comparison of *hash values* (lines 11–15). Therefore, the total computational time of our PoD is calculated through the combination of these three parts, which can be defined as follows:

$$T_{\text{total}} = T_{max\_search} + T_{gen\_nodeset} + T_{compare\_hash} \quad (9)$$

where T_{max_search} denotes the time cost of the maximum online duration searching, $T_{gen\_nodeset}$ is the generation cost of alternative miner nodes, and $T_{compare\_hash}$ is the cost of hash comparison for consensus messages. To specific, the for-loop (lines 3–6) takes $nodes.length$ steps and T_{max_search} can be calculated as follows:

$$T_{max\_search} = \sum_{i=1}^{\text{length}} T_i^{search} \quad (10)$$

where T_i^{search} is the time cost of the ith comparison operation (line 4), and the comparison operation should be performed $nodes.length$ times. Based on the same principle as (10), the $T_{gen\_nodeset}$ can be calculated as $T_{gen\_nodeset} = \sum_{i=1}^{\text{length}} T_i^{gen}$ and the $T_{compare\_hash}$ can be calculated as $T_{\text{compare\_hash}} = \sum_{i=1}^{nodeset.len} T_i^{hash}$. Therefore, we can also define the T_{total} as $T_{\text{total}} = \sum_{i=1}^{\text{length}} T_i^{\text{search}} + \sum_{i=1}^{\text{length}} T_i^{gen} + \sum_{i=1}^{\text{nodeset.len}} T_i^{\text{hash}}$. Here, we can see that the T_{total} is in direct proportion to the $number$ of consensus messages (i.e., $nodes.length$), whereas the single operation time (i.e., $T_i^{tags}, tags \in \{search, gen, hash\}$) remains relatively stable. In this article, we assume that the number of messages to be processed is n (i.e., $n = nodes.length$), then the execution frequency of lines 3–6 is $T_1(n) = 3n + 1$. Considering an extreme situation of algorithm execution, that is, all consensus nodes have the same online duration. Then, the execution frequency of lines 7–9 is $T_2(n) = 4n + 1$, and the execution frequency of hash comparison (lines 12–15) is $T_3(n) = 3n + 2$. Besides, the execution frequency of block checking(lines 21–24) is $T_4(n) = 4n + 1$. Hence, we can define the total execution frequency of PoD as $T(n) = \sum_{i=1}^{4} T_i(n) = 14n + 5$. Thus, we can see that the *time complexity* of Algorithm 3 is $O(n)$.

When it comes to the space complexity, the storage space required by the PoD is proportional to the scale of participating consensus nodes. Specifically, the space needed in Algorithm 3 is mainly composed of the space of the defined array $nodes$ (line 1), and the array $tmpIndex$ (line 2), then we can define the required space as $S(n) = S_{nodes}(n) + S_{tmplndex}(n) + 3$. Here, $S_{nodes}(n) = S_{tmpIndex}(n) = n$ (e.g., all consensus nodes have the same online duration), then the overall space $S(n)$ can be defined as $S(n) = 2n + 3$. Therefore, the space complexity of Algorithm 3 is $O(n)$.

REFERENCES

[1] "How Tesla is driving electric car innovation," 2013. [Online]. Available: https://www.technologyreview.com/s/516961/how-tesla-is-driving-electric-car-innovation/

[2] "In Sep. 2019, NIO EV sales in China increased by 14%," 2019. [Online]. Available: https://insideevs.com/news/375132/september-2019-nio-sales-china/

[3] A. Tintelecan, A. C. Dobra, and C. Marţiş, "LCA indicators in electric vehicles environmental impact assessment," in *Proc. Elect. Veh. Int. Conf.*, 2019, pp. 1–5.

[4] "New milestone: Tesla has more than 10000 global super charging piles," 2018. [Online]. Available: https://finance.yahoo.com/news/tesla-now-more-10-000-082553337.html

[5] D. A. Chekired, L. Khoukhi, and H. T. Mouftah, "Fog computing based energy storage in smart grid: A cut-off priority queuing model for plug-in electrified vehicles charging," *IEEE Trans. Ind. Inform.*, vol. 16, no. 5, pp. 3470–3482, May 2020.

[6] J. Ni, K. Zhang, X. Lin, and X. S. Shen, "Securing fog computing for internet of things applications: Challenges and solutions," *IEEE Commun. Surv. Tut.*, vol. 20, no. 1, pp. 601–628, Jan.–Mar. 2018.

[7] J. Xiao, G. Zu, X. Gong, and F. Li, "Observation of security region boundary for smart distribution grid," *IEEE Trans. Smart Grid*, vol. 8, no. 4, pp. 1731–1738, Jul. 2015.

[8] Y. Zhang, P. You, and L. Cai, "Optimal charging scheduling by pricing for EV charging station with dual charging modes," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 9, pp. 3386–3396, Sep. 2019.

[9] F. Knirsch, A. Unterweger, and D. Engel, "Privacy-preserving blockchain-based electric vehicle charging with dynamic tariff decisions," *Comput. Sci.-Res. Develop.*, vol. 33, no. 1/2, pp. 71–79, 2018.

[10] L. Gan, U. Topcu, and S. H. Low, "Optimal decentralized protocol for electric vehicle charging," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 940–951, May 2013.

[11] G. Liang, S. R. Weller, F. Luo, J. Zhao, and Z. Y. Dong, "Distributed blockchain-based data protection framework for modern power systems against cyber attacks," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 3162–3173, May 2019.

[12] J. Xu *et al.*, "Healthchain: A blockchain-based privacy preserving scheme for large-scale health data," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8770–8781, Oct. 2019.

[13] 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. Inform.*, vol. 15, no. 6, pp. 3602–3609, Jun. 2019.

[14] Z. Tian *et al.*, "Real-time charging station recommendation system for electric-vehicle taxis," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3098–3109, Nov. 2016.

[15] Y. Cao, T. Wang, O. Kaiwartya, G. Min, N. Ahmad, and A. H. Abdullah, "An EV charging management system concerning drivers' trip duration and mobility uncertainty," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 4, pp. 596–607, Apr. 2018.

[16] X. Xu, D. Ke, L. Li, and B. Xu, "Optimal charging strategy for heterogeneous EVs for cyber-physical-social systems," in *Proc. 2nd IEEE Conf. Energy Internet Energy Syst. Integr.*, 2018, pp. 1–5.

[17] M. A. Tajeddini and H. Kebriaei, "A mean-field game method for decentralized charging coordination of a large population of plug-in electric vehicles," *IEEE Syst. J.*, vol. 13, no. 1, pp. 854–863, Mar. 2019.

[18] S. Basudan, X. Lin, and K. Sankaranarayanan, "A privacy-preserving vehicular crowdsensing-based road surface condition monitoring system using fog computing," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 772–782, Jun. 2017.

[19] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," in *Big Data and Internet of Things: A Roadmap for Smart Environments*. New York, NY, USA: Springer, 2014, pp. 169–186.

[20] D. Wu *et al.*, "A fog computing-based framework for process monitoring and prognosis in cyber-manufacturing," *J. Manuf. Syst.*, vol. 43, pp. 25–34, 2017.

[21] G. Peralta, M. Iglesias-Urkia, M. Barcelo, R. Gomez, A. Moran, and J. Bilbao, "Fog computing based efficient IoT scheme for the industry 4.0," in *Proc. IEEE Int. Workshop Electron., Control, Meas., Signals Appl. Mechatronics*, 2017, pp. 1–6.

[22] S. Zahoor, S. Javaid, N. Javaid, M. Ashraf, F. Ishmanov, and M. Afzal, "Cloud–fog–based smart grid model for efficient resource management," *Sustainability*, vol. 10, no. 6, pp. 32–39, 2018.

[23] M. Li, L. Zhu, and X. Lin, "Efficient and privacy-preserving carpooling using blockchain-assisted vehicular fog computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4573–4584 Jun. 2019.

[24] Y. Lai, F. Yang, L. Zhang, and Z. Lin, "Distributed public vehicle system based on fog nodes and vehicular sensing," *IEEE Access*, vol. 6, pp. 22011–22024, 2018.

[25] Y. Chen, S. Ding, Z. Xu, H. Zheng, and S. Yang, "Blockchain-based medical records secure storage and medical service framework," *J. Med. Syst.*, vol. 43, no. 1, pp. 26–37, 2019.

[26] F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, and K. Ren, "A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks," *IEEE Netw.*, vol. 32, no. 6, pp. 184–192, Nov./Dec. 2018.

[27] J. Mihelj, Y. Zhang, A. Kos, and U. Sedlar, "Crowdsourced traffic event detection and source reputation assessment using smart contracts," *Sensors*, vol. 19, no. 15, pp. 78–92, 2019.

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

[29] S. Mondal, K. Wijewardena, S. Karuppuswami, N. Kriti, D. Kumar, and P. Chahal, "Blockchain inspired RFID based information architecture for food supply chain," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5803–5813, Jun. 2019.

[30] T. M. Fernández-Caramés, O. Blanco-Novoa, I. Froiz-Míguez, and P. Fraga-Lamas, "Towards an autonomous industry 4.0 warehouse: A UAV and blockchain-based system for inventory and traceability applications in big data-driven supply chain management," *Sensors*, vol. 19, no. 10, pp. 52–74, 2019.

[31] "Bitcoin: A peer-to-peer electronic cash system." 2008. [Online]. Available: https://bitcoin.org/en/bitcoin-paper

[32] S. King and S. Nadal, ''PPCoin: Peer-to-peer crypto-currency with proof-of-stake," Self-Published Paper, vol. 19, Aug. 2012.

[33] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld, "Proof of activity: Extending Bitcoin's proof of work via proof of stake," IACR Cryptology ePrint Archive, vol. 2014, p. 452, 2014.

[34] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1495–1505, Apr. 2019.

Hongzhi Li received the M.S. degree from Wuhan University of Technology, School of Computer Science and Engineering, Wuhan, China, in 2014. He is currently working toward the Ph.D. degree with Shanghai Maritime University, Shanghai, China.

His research interests include blockchain technology and information security.

Dezhi Han (Member, IEEE) received the B.S. degree from Hefei University of Technology, Hefei, China, in 1990, and the M.S. and Ph.D. degrees from Huazhong University of Science and Technology, Wuhan, China, in 2001 and 2005, respectively.

He is currently a Professor of computer science and engineering with Shanghai Maritime University, Shanghai, China.

Mingdong Tang (Member, IEEE) received the B.S. degree in electrical engineering from Tianjin University, Tianjin, China, in 2000, the M.S. degree in control engineering from Shanghai University, Shanghai, China, in 2003, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2010.