Secure Ride-Sharing Services Based on a Consortium Blockchain

Di Wang and Xiaohong Zhang*

***Abstract*—Due to poor traffic conditions and the high costs of travelling by private cars, ride sharing has become a popular means to trip. In view of the security threats and centralization existing in the current ride-sharing service, we propose a secure ride-sharing scheme based on a consortium blockchain, which can guarantee the security, confidentiality and privacy of data interaction via attribute-based proxy re-encryption algorithm. First, the passenger presets the access structure and encrypts the data using attribute-based encryption. The ciphertext is then sent to the roadside unit (*RSU*), which broadcasts the carpooling request to the driver. After receiving the request, the driver sends the itinerary attribute to *RSU* , which performs carpool matching according to received ciphertext and itinerary attributes, then the ciphertext is re-encrypted and sent to the matched driver. Second, the master node uses an improved DPoS (Delegated Proof of Stake) consensus to verify the carpool record, which is stored on the blockchain after the verification is successful. In case of disputes, block data can be utilized for traceability. Third, drivers and passengers use the credibility mechanism to score each other after ride sharing. In addition, trusted authority can reveal the real identity of malicious users. Finally, we conduct security analysis and performance evaluation for our proposed scheme. The results manifest that our scheme not only meets the security and privacy requirements of ride-sharing services, but also effectively resists potential security risks. Therefore, our scheme is feasible, efficient and suitable for ride-sharing services.**

***Index Terms*—Consortium blockchain, data security, proxy re-encryption, ride-sharing services, trusted mechanism.**

I. INTRODUCTION

With the development of the sharing economy and vehicular ad hoc networks (VANETs), ride sharing (carpooling) and ride hailing are increasingly becoming a significant mode of travel [1]. On the one hand, ride sharing alleviates traffic congestion during rush hour, solves the shortage of taxi in bad weather. On the other hand, although the number of private cars is increasing year by year, the actual utilization rate is not very high so that private cars are often idle. At the same time, regular maintenance and car insurance are needed for these vehicles, which will increase people's economic burden. However, ride sharing is very convenient, low cost, and can reduce energy consumption as well as pollution (e.g. exhaust emissions).

In view of above benefits, some ride-sharing services provider such as Uber [2] and Lyft [3] have recently provided services to millions of users in hundreds of cities. Didi, the largest ride-sharing service provider in China, has 450 million users and provides 20 million ride-sharing services for users every day [4]. In order to carpool [5-6], passengers and drivers publish their itineraries (e.g. starting point, departure time, and destination) on the application (app). The ride-sharing service provider matches supply and demand based on the information provided by passengers and drivers. Relevant information about the driver is sent to the mobile device of the successfully matched passenger and the passenger's itineraries are sent to the successfully matched driver, thereby facilitating the driver providing the passenger with ride-sharing services.

Along with the convenience of ride-sharing services, numerous security threats have also emerged. Consequently, many academics attach great attentions to researching and improving the ride-sharing service [7]. Ni *et al.* [8] proposed an anonymous two-way authentication scheme based on BBS+ signature, which is used for registration of the identity of passengers and drivers and then authentication of their identities when they use the system. The ride-sharing service system matches the authenticated passengers and drivers to achieve the shared ride mutually. The ride-sharing system provides plaintext information, such as the identity and location of passengers, potentially compromising the privacy and safety of the people in the carpool. Sherif *et al.* [9] used group signatures to protect user's privacy. The server used similarity measurement techniques to detect the similarity of travel data and searched for the driver that best matched the passengers. A ride-sharing platform based on trust level [10] was used to improve the security of the system and the trust between users. The platform relies on the cloud server to provide users with a reliable measurement of whether to trust another user or not. Hallgren *et al.* [11] achieved ride-sharing matching based on the similarity of departure and destination and used the Shamir secret sharing scheme to protect the privacy of the user's route.

The above schemes have achieved rideshare, they meanwhile bring about some risks and challenges, one of which is that security and privacy are threatened. It is easy for an attacker to eavesdrop, tamper with, or forge trip information sent by users (passengers and drivers) on an open

Manuscript received January 27, 2020; revised March 25, 2020, June 19, 2020 and July 11, 2020; accepted August 28, 2020. This work was jointly supported by the National Natural Science Foundation of China (Grant 61763017 and Grant 51665019), the Scientific Research Plan Projects of Jiangxi Education Department (Grant GJJ150621), the Natural Science Foundation of Jiangxi Province (Grant 20161BAB202053 and Grant 20161BAB206145), and the Innovation Fund for Graduate Students of Jiangxi Province (Grant YC2017-S302). (*Corresponding author*: Xiaohong Zhang.)

Di Wang and Xiaohong Zhang are with school of Information Engineering, Jiangxi University of Science and Technology, Ganzhou, China (*e-mail*: 1065889068@qq.com, xiaohongzh@263.net).

wireless communication network. Unsurprisingly, Hackers can infer users' privacy information, such as work unit and home address from the obtained information. The issue is deteriorated by the case that attackers misuse users' privacy information. For example, an attacker burgles users' homes once they have left. Besides, systems used by most existing ride-sharing services have a centralized structure, which depends on a trusted third party to store and process travel information sent by users. Once the central node is hacked, it can no longer be considered to contain trusted data as the basis for dispute arbitration. What's more, the single point attack can be used to easily threaten the security of the entire system. Worse still, the centralized structure is prone to information monopoly or information islands. As a result, the ride-sharing service providers may abuse data or sell users' data to other organizations. For instance, Uber used the cloud platform to leak users' data to other organizations [12]. Last but not least, existing ride-sharing schemes have large computational costs and communication overheads, which give rise to the poor performance and affect the matching efficiency in ride-sharing systems.

Inspired by the above challenges, it is of vital importance to propose a secure, decentralized, data traceable and efficient ride-sharing service. As an emerging technology with the features of secure, credibility, tamper resistance as well as traceability, blockchain has set off a research boom all over the world. Without authorization, any node in the public blockchain can join or leave the network freely, resulting in all nodes having read and write permissions for data. It is an arduous task for the public blockchain to reach a consensus quickly among nodes. Moreover, an attacker can easily forge a large number of fake nodes. Therefore, the public blockchain is not suitable for ride-sharing services. On the contrary, a private blockchain is only open to individual organizations or institutions. Furthermore, data read as well as write permissions are strictly controlled by a few trusted nodes, which is no distinguished from the centralized database. Thereby, the consortium blockchain is preferable for ride-sharing services.

In this paper, we propose a secure ride-sharing service based on a consortium blockchain. The passenger encrypts the trip data and sends the ciphertext to the roadside unit nearest to the starting point. The roadside unit then sends a carpool request to the drivers in the area. The driver who is able to provide the ride-sharing service responds by sending his trip attributes to the roadside unit. As a fog node, the roadside unit detects whether the driver's travel attributes meet passenger's data access structure and matches the passenger with a suitable ride-sharing driver. The passenger's ciphertext is re-encrypted and the re-encrypted ciphertext is sent to the driver. The driver decrypts the re-encrypted ciphertext to obtain the passenger's travel information. The driver then goes to the passenger's starting point and picks up the passenger. The roadside unit node packs the carpooling data to generate the block and uses the improved DPoS (Delegated Proof of Stake) consensus to verify the effectiveness of the block. After successful verification, the block is connected to the blockchain. The main contributions of this article are as follows:

- For the sake of the centralization existing in ride-sharing systems, a novel secure ride-sharing framework is proposed, which utilizes the distributed blockchain structure to effectively prevent single point attacks. Furthermore, the credibility mechanism is designed to encourage roadside units to match the appropriate drivers for passengers, and increase the credibility of drivers and passengers. For the established framework, an improved DPoS consensus mechanism is used to generate blocks and verify data, so that the blockchain network can still operate normally even if one third of the malicious roadside nodes exist. Hence, the data is effectively prevented from being tampered. In addition, drivers and passengers are provided with conditional privacy, that is, to protect the identity privacy of passengers and drivers, but in case of disputes, trusted authority can trace and disclose the real identity of malicious users.
- An attribute-based proxy re-encryption algorithm is proposed to tackle the issues such as the current algorithm only meets a single feature, the user has a large computational overhead, and the proxy uses the re-encryption key to encrypt the ciphertext without permission. Our algorithm not only satisfies unidirectionality, verifiability and confidentiality, but also resists replacement attacks as well as collusion attacks between roadside units and drivers who satisfy the access structure. As a fog node, the roadside unit performs the ride-sharing matching based on the data access structure set by the passenger and the attributes of the driver's itinerary. Then the ciphertext is converted into re-encrypted ciphertext, which reduces the computational burden of users, decreases the communication overhead caused by frequent encryption as well as decryption, and improves the efficiency and effectiveness of ride-sharing services.
- We analyze the security of our scheme and evaluate its performance. The results prove that our scheme not only meets the security and privacy requirements of ride-sharing services, but also has more advantages in computational costs and communication overheads as the number of attributes increases. With 50 attributes, the computational overhead of generating re-encryption key as well as decryption is decreased by 68.02% and 86.71% respectively, while our scheme ensures the controllability and verifiability of ciphertext. Moreover, the communication overhead of generating private keys as well as re-encrypted ciphertext is reduced by 26.2% and 12.51% respectively. In consequence, our scheme is suitable for ride-sharing services.

The reminder of this paper is organized as follows. Related work is described in Section II. Section III mainly introduces the technical background, including fog computing, bilinear mapping, linear integer secret sharing, and attribute-based proxy re-encryption algorithm. Details of the system framework of ride-sharing services are provided in Section IV. In section V, we describe our specific implementation process of secure ride-sharing services. Security analysis and the results of performance evaluation are provided in Sections VI. Finally, Section VII concludes the paper.

II. Related Work

A. Ride-sharing Services

In order to ensure the safety and privacy of ride sharing, both the embedded road network technology and homomorphic encryption are integrated to estimate the distance between passengers and drivers. The information of the drivers nearest to the passenger is then obtain as an output using the merged intermittent circuit [13]. This scheme ensures the privacy and accuracy of ride-sharing matching, but it is unable to prevent collusion attacks. To provide taxi services to passengers, Yu *et al.* [14] proposed a region-based estimation of minimum travel time, using security protocols to schedule taxis with minimum additional travel time. A lightweight dynamic spatial query scheme for online ride-hailing services [15] not only protects the uses' location privacy, but also partitions the area according to the user's location so that the server can search for the matching driver nearest to the passenger's boarding point. Yu *et al.* [16] proposed a lightweight privacy protection ride-matching scheme using ciphertext-blinding to calculate the distance between passengers and drivers. Approximate distance of road is applied to achieve secure matching of online ride sharing. Pham *et al.* [17] proposed a system for a privacy-enhanced ride-hailing service, which not only protects the privacy of passengers, but also improves the accuracy of vehicle matching and fare calculation. Security of the system will be threatened if the service provider colludes with the driver for greater benefits. A spatial region-based selection mechanism [18] chooses the correct ride-sharing partner according to the feasibility of saving on travel time.

The above scheme only partially improves the security and privacy of a ride-sharing service, because it continues to rely on a central processor to process and control the data. In order to prevent the central node from being attacked, Li *et al.* [19] proposed a blockchain-based ride-sharing scheme, which uses both proximity testing to achieve one-to-many matching and range query technology to attain destination matching. In addition, the blockchain is used to store carpool records for data review. Passengers and drivers are making use of the blockchain-based secure accounting protocol [20] to negotiate boarding locations, routes, and fares, which in turn will prevent malicious drivers from intentionally detouring or extending service time. A distributed ride-sharing service based on social group structure [21] solves the problem of matching geographical information in traditional ride-sharing services, but does not take into account the volume of traffic on urban roads. To improve the flexibility, robustness, and scalability of the ride-sharing system, Nabil *et al.* [22] introduced non-transferable and transferable ride-sharing services and provided passengers with suitable ride-sharing services according to their preferences. A location privacy-preserving scheme based on the MinHash algorithm [23] selects landmarks near to the user, as its location features, and applies the MinHash algorithm to optimize the similarity between their characteristic vectors. The scheme improves the speed of calculating the distance between passengers and drivers, but it still requires a huge computational cost.

B. Blockchain

Internet of Things (IoT) [24], Cloud Computing (CC) [25] and blockchain [26], as new technologies with blooming development, have attracted great attentions from scholars. Stergiou *et al.* [27] elaborated the main characteristics of IoT as well as CC and investigated the potential security risks brought by the integration of these two technologies. Then the contribution of IoT and CC to promoting big data application was explored [28]. Additionally, a new technology based on IoT and CC was used for efficient digital media transmission [29]. Blockchain came out in 2008, when Satoshi Nakamoto proposed a digital currency called Bitcoin without trust endorsement [26], and then people extracted the blockchain technology from the underlying Bitcoin. Blockchain technology [30-31] uses cryptography to encrypt information to prevent data from being forged and tampered with. Each block stores the hash value of the previous block to form a chain structure. Timestamps, distributed consensus, and incentive mechanisms promote blockchain technology to achieve credible value transmission even when network nodes do not trust each other. Smart contract [32], as the code automatically executed on the blockchain, is created and called in the distributed network. Even if some nodes fail or do evil, it will not affect the operation of the smart contract.

The characteristics of blockchain technology, such as decentralization, security, credibility, tamper proof and traceability, have attracted extensive attentions from researchers [33]. As a result, blockchain technology is widely used, for example, in the fields of vehicular ad hoc networks (VANETs), smart grids, and medical treatments. Zhang *et al.* [34] proposed a regulation mechanism for traffic lights, in which the traffic department processes and analyzes road condition information stored on the blockchain. The smart contract is utilized to control the duration of green lights and increase traffic flow. Wang *et al.* [35] proposed a secure data sharing scheme based on a consortium blockchain to prevent data from being accessed by the unauthorized. What's more, the service department uses smart contracts to provide customized services for drivers. To solve problems posed by hidden dangers of energy security and power data leakage in a smart grid, a privacy-protected permissioned blockchain [36] combining edge computing technology was used. The security awareness strategy is used to implement intelligent control and secure scheduling of power resources. Wang *et al.* [37] established a consortium blockchain for sharing medical data based on ACP (artificial systems + computational experiments + parallel execution), which is convenient for doctors to diagnose, treat, and predict the likely development of diseases in patients.

III. Prerequisite Knowledge of Related Theories

A. Fog Computing

Fog computing [36] is a new technology that extends cloud computing. It results in a small data center by introducing computing storage devices (i.e. fog nodes) locally, to solve the problems of delays in long-distance data transmission and vulnerability to distributed denial of service attacks. Fog computing has been applied to VANETs to provide vehicles

with real-time services such as safety warnings and autonomous driving. The roadside unit node in the present paper acts as a fog node, which has strong computing power, large storage space and communication functions. In order to facilitate ride-sharing services, the roadside unit node applies fog computing to process the request and response information sent by passengers and drivers.

B. Bilinear Pairing

Suppose G_1 and G_2 are multiplicative cyclic groups with prime orders p. g is the generator of G_1. Bilinear pairing [39] $e: G_1 \times G_1 \to G_2$ satisfies the following properties.

(1) Bilinearity: for all $V, W \in G_1$ and $v, t \in Z_p^*$, $e(V^v, W^t) = e(V^t, W^v) = e(V, W)^{vt}$.

(2) Computability: for all $V, W \in G_1$, bilinear pairing $e(V, W)$ can be effectively calculated.

(3) Non-degeneracy: $e(V, W) \neq 1$ for all $V, W \in G_1$.

C. Linear Integer Secret Sharing

We used the Linear Integer Secret Sharing (LISS) scheme proposed by Damgard [40]. Suppose n secret sharers are represented as $S = \{1, 2, \ldots, n\}$, (M, ρ) is the access structure on S, and L is an integer constant. If the secret distributor wants to share the secret parameter s in the interval $\left[-2^L, 2^L\right]$ with the secret sharer, each set satisfying (M, ρ) can reconstruct s called an authorized set. Each secret share in LISS consists of an integer set $(s_i)_{i \in I}$. Any s_i belongs to a secret sharer. The given authorization subset $(s_i)_{i \in I}$ can reconstruct the secret parameters by linear combination $s = \sum_{i \in I'} \lambda_i s_i$, where $\{\lambda_i\}_{i \in I'}$ is the integer coefficient.

D. Attribute-based Proxy Re-encryption

According to the different embedding positions of different access policies, attribute-based encryption can be divided into either key-policy attribute-based encryption (KP-ABE) or ciphertext-policy attribute-based encryption (CP-ABE) [41]. When using KP-ABE [42], the authorized organization defines the access policy of the data and embeds the access policy into the user's key. Only when the access policy in the user's key satisfies the access policy set by the data owner, can the user decrypt the data. Alternatively, when using CP-ABE, the data owner defines the access policy of the data and embeds it in the ciphertext [43]. Only when the attributes of the data visitor satisfy the data access policy set by the data owner, can the visitor decrypt the data.

Proxy re-encryption was proposed by Blaze [44]. As an example, the data owner Amy uses her public key to encrypt the plaintext to generate the ciphertext C_A and then sends C_A to the agent Bill who re-encrypts C_A to generate the re-encrypted ciphertext C_A' that Chris can decrypt. In the whole process, the data owner Amy only needs to perform one encryption operation. The agent Bill only owns the ciphertext and cannot obtain the plaintext, which not only guarantees the confidentiality of the data, but also reduces the computational overheads of the data owner. The paper combines attribute-based encryption algorithm and proxy re-encryption algorithm. We propose a ciphertext-policy attribute-based proxy re-encryption algorithm, which primarily includes the following seven algorithms:

(1) $(SPK, MSK) \leftarrow Setup(k, X, U)$. The initialization algorithm *Setup* is executed by the trusted authority. Given the security parameters k, the attribute space X, and the attributes shared by the user U, the output is the system public key SPK and the system master key MSK. The system public key SPK is exposed and the system master key MSK is kept secret.

(2) $SK \leftarrow KeyGen(SPK, MSK, A)$. The algorithm generates the private key. *KeyGen* is implemented by the trusted authority. The system public key SPK, the system master key MSK, and the user's attribute set A are input. The user's private key SK associated with the attribute set A is output.

(3) $C \leftarrow Enc((M, \rho), SPK, m)$. The encryption algorithm *Enc* is executed by the users. Ciphertext C is generated by the access structure (M, ρ), a system public key SPK, and the plaintext m.

(4) $RK \leftarrow \mathrm{Re}KeyGen(SPK, SK, (M', \rho'))$. The algorithm generates the re-encryption key. $\mathrm{Re}KeyGen$ is executed by the user. Given the system public key SPK, the user's private key SK, and the new access structure (M', ρ'). The re-encryption key RK is output. If the user's set of attributes A satisfies the access structure (M, ρ), the proxy can use the re-encryption key RK to convert the access structure (M, ρ) of the ciphertext to a new access structure (M', ρ').

(5) $C' \leftarrow \mathrm{Re}Enc(SPK, C, RK)$. The re-encryption algorithm $\mathrm{Re}Enc$ is executed by the proxy. The system public key SPK, the ciphertext C corresponding to the access structure (M, ρ), and the re-encryption key RK are input. When the user's set of attributes A satisfies the access structure of the ciphertext, the re-encrypted ciphertext C' associated with the new access structure (M', ρ') is generated. Otherwise, $\perp$ is output.

(6) $1/\perp \leftarrow Verify(C')$. The algorithm to verify the re-encrypted ciphertext. $Verify(C')$ is executed by the user to verify whether the re-encrypted ciphertext C' is correct. If C' is verified then 1 is output, otherwise $\perp$ is output.

(7) $m \leftarrow Dec(C', SK)$. After the algorithm *Verify* verifies the re-encrypted ciphertext and outputs 1, the user executes the decryption algorithm *Dec*. The re-encrypted ciphertext C' and the user's private key SK are input, then the plaintext m is output.

IV. The Proposed System Framework

In our scheme, passengers and drivers send ciphertext and trip attributes to the roadside unit respectively. The roadside unit matches the driver who meets the access structure for the passenger via fog computing. Next, the ride-sharing report is recorded in the block and the preselected accounting node participates in the consensus. The successfully verified block is connected to the blockchain. The passenger and driver will

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

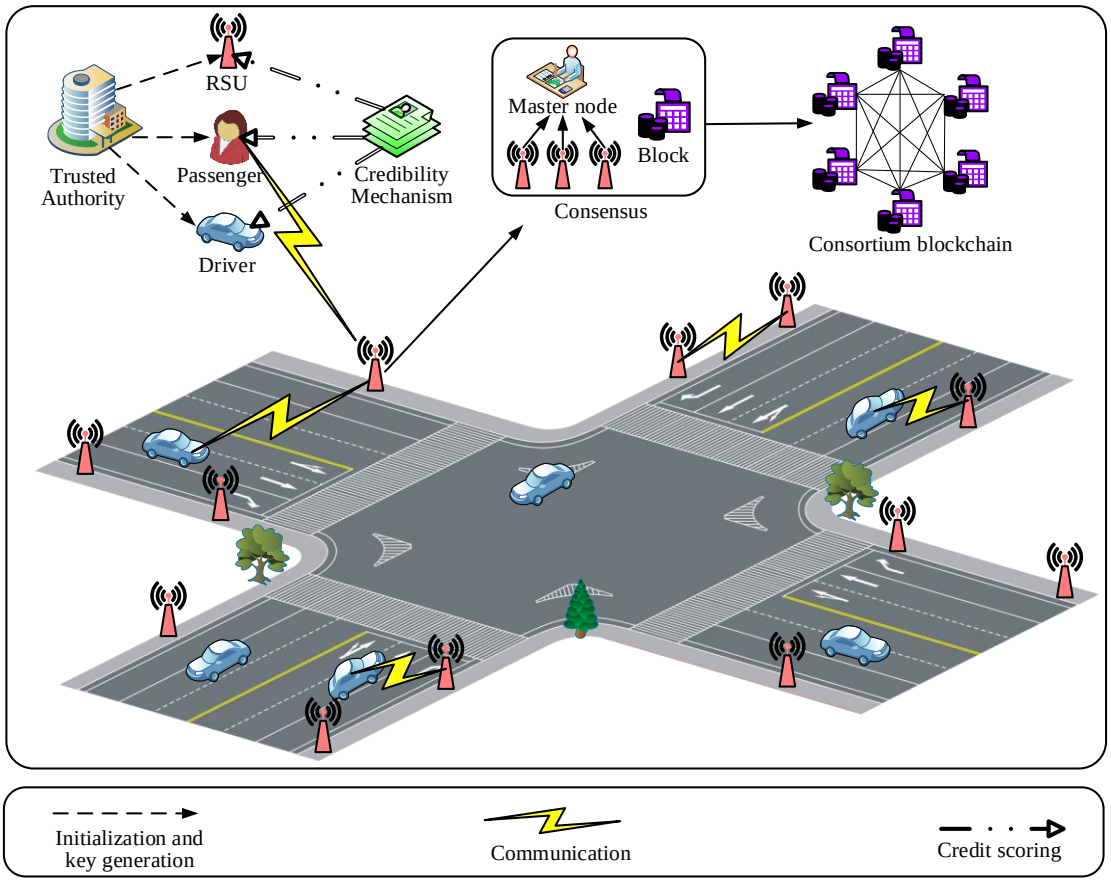

Fig. 1. The ride-sharing system framework based on a consortium blockchain

score each other's credibility as soon as ride-sharing service is completed. In case of disputes, the trusted authority uses the information stored on the blockchain for arbitration. The system framework is shown in Fig. 1, mainly containing the trusted authority, roadside units, users, the consortium blockchain, the consensus mechanism, and the credibility mechanism. The trusted authority generates keys for roadside units as well as users. The roadside unit matches the appropriate driver for the passenger and re-encrypts the ciphertext. Users are composed of drivers and passengers. To realize information interaction, wired communication technology is applied among roadside units, whereas dedicated short distance communication is adopted between roadside units and users. For example, suppose that the passenger's current location belongs to the roadside unit RSU_i and the source for the journey is affiliated with the roadside unit RSU_j. Passengers who feel like enjoying ride-sharing service need to send ride-sharing request and ciphertext to RSU_i via dedicated short distance communication. Next, RSU_i forwards the received information to RSU_{i+1} by wired communication. Then, RSU_{i+1} transmits the received information to RSU_{i+2} ,…, and finally RSU_{j-1} sends the received information to RSU_j. The above information transferring process among roadside units is not require, but only if the passenger's current location as well as the source for the journey are subordinate to the same roadside unit. The ride-sharing report stored on the consortium blockchain can be utilized for digital forensics. Typically, the consensus mechanism guarantees data consistency. Additionally, the roadside units that complete ride-sharing matching and block verification are rewarded with credit scores, otherwise they are deducted credit scores as punishment. The detailed definition of each entity is as follows.

(1) Trusted authority

The trusted authority (TA) is primarily responsible for the initialization of the system, generating keys for roadside units and users, arbitrating disputes and exposing the identity of malicious users. It is assumed that the trusted authority is not easily captured and is completely trustworthy.

(2) Roadside unit

As a fog node with strong computing power, strong network communication capability, and significant storage space, the roadside unit (RSU) can provide ride-sharing services based on the information sent by passengers and drivers. The credit score of the roadside unit increases after each ride sharing. The roadside unit re-encrypts the travel ciphertext to generate the re-encrypted ciphertext that can be decrypted by the driver. The accounting node records the ride-sharing report in the block and verifies the data in the block. After reaching a consensus, the current block is connected to the blockchain. Carpooling report stored on the blockchain can be used as the basis for the arbitration of disputes by the trusted authority.

(3) Users

1) Passengers

To enjoy a comfortable and fast ride-sharing service, passengers use a mobile application to enter their travel data, which includes not only their place of departure, their earliest and latest departure time, and destination, but also the latest time by which they must have arrived at their destination. The passenger defines the access structure of the travel data. The travel data and access structure are encrypted to generate the ciphertext that is sent to the roadside unit. The roadside unit will provide the ride-sharing matching result to the passenger. After receiving the result, the passenger need only go to their place of departure and wait for the driver to arrive.

2) Drivers

The commuter who provides ride-sharing services is referred to as the driver. The driver uses the mobile application to generate their travel attributes, which include their place of departure, departure time, destination and their latest time of arrival at the destination. The driver encrypts the service information to generate the ciphertext and sends it to the roadside unit. The roadside unit realizes ride-sharing matching by detecting whether the driver's travel attributes meet the access structure of the passenger's travel data. The roadside unit re-encrypts the passenger's ciphertext to generate the re-encrypted ciphertext, which is sent to the matching driver. After receiving the re-encrypted ciphertext, the driver decrypts it to obtain the passenger's itinerary. The driver departs from his/her current position and picks up the passenger at point of departure, and then delivers the passenger to the destination of passenger, after which the driver goes to his/her destination.

(4) Consortium blockchain

The consortium blockchain with only preselected nodes participating in block validation is more preferable for our proposed ride-sharing services scheme. Not requiring all nodes to participate in the consensus can effectively reduce the network burden and computational costs, which accelerate the speed of block generation.

(5) Consensus mechanism

The Delegated Proof of Stake (DPoS) [45] is known as an efficient, decentralized and flexible consensus mechanism. It gives nodes in the network the right to vote and elects 101 representatives through a fair and democratic voting method. In the follow-up process, new delegates can be re-voted for based on their performance. The consensus mechanism can reasonably control the number of nodes participating in the consensus process, which reduces calculation costs and speeds up the generation of blocks as well as data verification. The block generated by DPoS consensus does not consume much computing power, but the power of representatives is large. Once malicious nodes become representatives, they can generate many blocks in a very short time leading to network bifurcations. Therefore, we improve DPoS consensus to achieve block generation and data verification. The basic principle of the improved DPoS consensus mechanism is as follows. The top 101 roadside unit nodes with the highest credit score are selected as accounting nodes and are added to the accounting node list ANL. Accounting nodes alternately acts as the master node to package the data and generate blocks. After the block is generated, it will not be linked to the blockchain immediately. Practical Byzantine Fault Tolerance (PBFT) is used to verify the data in the blockchain. The current block is connected to the blockchain only if the block verification is successful. In the event of generating the wrong block, the accounting node will be removed from the accounting node list ANL and its credit score will be reduced. Afterwards, the next accounting node on the ANL will continue to generate the block. Each roadside unit node has a performance indicator that records the behavior of each accounting node. Each accounting node is selected based on this indicator.

(6) Credibility mechanism

The ride-sharing service is completed when the driver delivers the passenger to destination. The passenger will score the driver's service according to the following formula

$$Credit_D(i+1) = Credit_D(i) + I \tag{1}$$

where $Credit_D(i)$ represents the credibility score of the driver D after completion of i times ride-sharing service, the driver's initial credibility score was 0. I denotes the real feelings of passengers during the ride-sharing process. The specific values are as follows

$$I = \begin{cases} -1, terrible \\ 1, just\ so\ so \\ 2, comfortable \end{cases} \tag{2}$$

The manner in which the driver's rate the passenger's credibility is similar, and is not described here. The higher the credibility score of the driver, the earlier the RSU matches the right passenger. Similarly, the higher the credibility score of the passenger, the earlier the RSU matches the right driver.

As long as RSU node completes ride-sharing matching and block verification, it obtains credit score as a reward. Credit score of RSU cannot be given to other roadside units. In case the roadside unit node fails to provide ride-sharing matching or the block generation failure, its credit score is deducted as a penalty. The initial credit score of RSU is 0, and credit score is calculated by the following formula

$$Credit_{RSU}(i+1) = Credit_{RSU}(i) \pm 1 \tag{3}$$

where $Credit_{RSU}(i)$ is the credibility score of RSU after completing i times carpool matching. The results of the equation are added if RSU is successfully matched, otherwise the results of the equation are subtracted. RSU will be removed from the accounting nodes list when its credibility score is lower than the threshold. RSU with higher credibility scores will be selected to fill the accounting node list. The introduction of a credibility mechanism fully mobilizes the desirability of accounting nodes to participate in the consensus.

V. The Proposed Scheme

This section elaborates on the implementation process about our proposed secure ride-sharing services based on a consortium blockchain, mainly including secure ride-sharing services, consensus process as well as credit score and malicious user disclosure, as shown in Fig. 2. Key symbols

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

are shown in Table Ⅰ.

In Fig. 2, secure ride-sharing services consist of seven parts: system initialization, private key generation, data encryption, ride-sharing matching, re-encryption key generation, ciphertext re-encryption, and data decryption. During system initialization, *TA* generates the system public key *SPK* and the system master key *MSK*. In addition, the user assigns an identifier to his/her own data and calculates his/her own pseudonym *FID*. Then, user forwards (FID, ID, v) to *TA*. In private key generation, *TA* generates private key and public key for users. Passengers

TABLE Ⅰ
KEY SYMBOLS

Symbol	Definition
TA	the trusted authority
RSU	the roadside unit
P_j	passenger P_j
D_j	driver D_j
G_1, G_2; g	multiplicative cyclic group; group generator
k; p	system security parameter; prime number
X; A_{P_j}	attribute space; attribute set of passengers P_j
f_{encode}; H_1, H_2	encoding transformation function; hash function
MSK; *SPK*	system master key; system public key
ID; *FID*	real identity; pseudonym
SK, *PK*; *RK*	private key, public key; re-encryption key
Tag	data label
(M,ρ); (M',ρ')	access structure; new access structure
ACK	confirmation message
C; C'	ciphertext; re-encrypted ciphertext
m	plaintext
NL; *ANL*	node list; accounting node list

predefine the access structure for the plaintext m and encrypt m to generate ciphertext C prior to ride-sharing matching. Afterwards, passengers send ride-sharing request $\mathrm{Re}q$ and ciphertext C to the affiliated roadside unit *RSU*, which matches the right driver for the passenger after receiving $\mathrm{Re}q$. *RSU* subsequently broadcasts $\mathrm{Re}q$ within jurisdiction. The driver D_j sends travel attributes A_{D_j} to *RSU*, which uses fog computing to determine whether A_{D_j} meet the data's access structure (M,ρ). *RSU* then transmits *Reply* to the passenger for the sake of informing the passenger of submit the re-encryption key. After that, *RSU* re-encrypts the ciphertext C to generate the re-encrypted ciphertext C' and forwards C' to the successfully matched driver, which then goes to the departure point for picking up the passengers within the specified time. The accounting nodes on the list take turns at acting as the master node that packages C as well as C' to generate blocks. Other nodes verify the data in blocks. After the verification is successful, the current block is connected to the blockchain for permanent preservation. The master node obtains credit scores as a reward for generating blocks. The block body of the consortium blockchain stores the carpooling report, which includes passengers' travel data and drivers' itinerary attributes. The receipt root in the block head stores ride-sharing service results, such as the credit score of the passenger and driver. The transaction root records the Merkle tree of the ride-sharing report. However, the status root stores the status of the passenger as well as driver, such as the driver in service. After arriving at the destination, passengers and drivers score each other's credit. In the event of disputes between drivers and passengers, a forensic investigation is performed based on the data stored on the blockchain.

A. Secure Ride-sharing Services

(1) System initialization

TA generates the system public key *SPK* and the system master key *MSK*. In addition, the user assigns an identifier to his/her own data and calculates his/her own pseudonym *FID*. Then, the user forwards (FID, ID, v) to *TA*. The detailed steps are as follows.

Step 1: *TA* selects the multiplicative cyclic groups G_1 and G_2 with a large prime number p, g is the generator of G_1, and there is bilinear mapping $e: G_1 \times G_1 \to G_2$. Suppose k is the system security parameter, X is the attribute space. *TA* defines the encoding transformation function $f_{encode}: G_1 \to \{0,1\}^k$ and hash function $H_1: \{0,1\}^* \to Z_p^*$ as well as $H_2: \{0,1\}^* \to \{0,1\}^k$, where $\{0,1\}^*$ is a string of any length.

Step 2: *TA* randomly selects $g_1 \in G_1$ and $x, y, z, \alpha \in Z_p^*$. U denotes the user's common attributes. Then *TA* randomly selects $G_3, T_i \in G_1$ for all $i \in X$ to calculate $B = e(g,g)^z$, $h_1 = g^\alpha$, $h_2 = g^x$ and $Y = g^y$. The system public key is $SPK = \{p, g, g_1, G_1, G_2, e, B, G_3, \forall i \in X: T_i, h_1, h_2, Y, H_1, H_2, f_{encode}\}$, the system master key is $MSK = \{x, \alpha, g^z\}$, where *SPK* is open to the public and *MSK* is kept by *TA* in secret.

Step 3: The user assigns a unique identifier $\beta = H_1(Tag)$ to their own data and β is embeded to the re-encryption key and ciphertext. Only when the identifier of the re-encryption key and the ciphertext are equal, can the ciphertext be re-encrypted, effectively preventing data from being accessed illegally.

Step 4: The passenger P_j randomly selects $v_{P_j} \in Z_p^*$ to calculate the pseudonym $FID_{P_j} = ID_{P_j} \oplus H\left(PK_{P_j} + v_{P_j}\right)$ of the passenger P_j, where ID_{P_j} is the real identity of P_j. The passenger sends $\left(FID_{P_j}, ID_{P_j}, v_{P_j}\right)$ to *TA* and *TA* stores $\left(FID_{P_j}, ID_{P_j}, v_{P_j}\right)$ in the identity list. As process of calculating the pseudonym by the driver D_j is similar to what has been described for the passenger, so it will not be described here. In the event of a dispute, the information stored in the identity list is used to reveal the real identity of the malicious user.

(2) Private key generation

TA first verifies the identity of P_j .prior to generating the private key for the passenger P_j. Then, *TA* randomly selects $a \in Z_p^*$ and $a_i \in Z_p^*$ for each attribute i of the passenger P_j if and only if the identity of P_j is valid. Next, *TA* calculates $N = g^{(z+a)/\alpha}$, $N_1 = Y^{a/x}$, $N_2 = G_3^{a/x}$ and $N_3 = g^{a/x}$. The private key of P_j is $SK_{P_j} = \left\{A_{P_j}, N, N_1, N_2, N_3, \forall i \in A_{P_j}: Q_i = g^a T_i^{a_i}, Q_i' = g^{a_i}\right\}$, where A_{P_j} represents the passenger's attributes and $PK_{P_j} = gSK_{P_j}$ is calculated as the passenger's public key. The driver's private key and public key are generated in a similar way.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

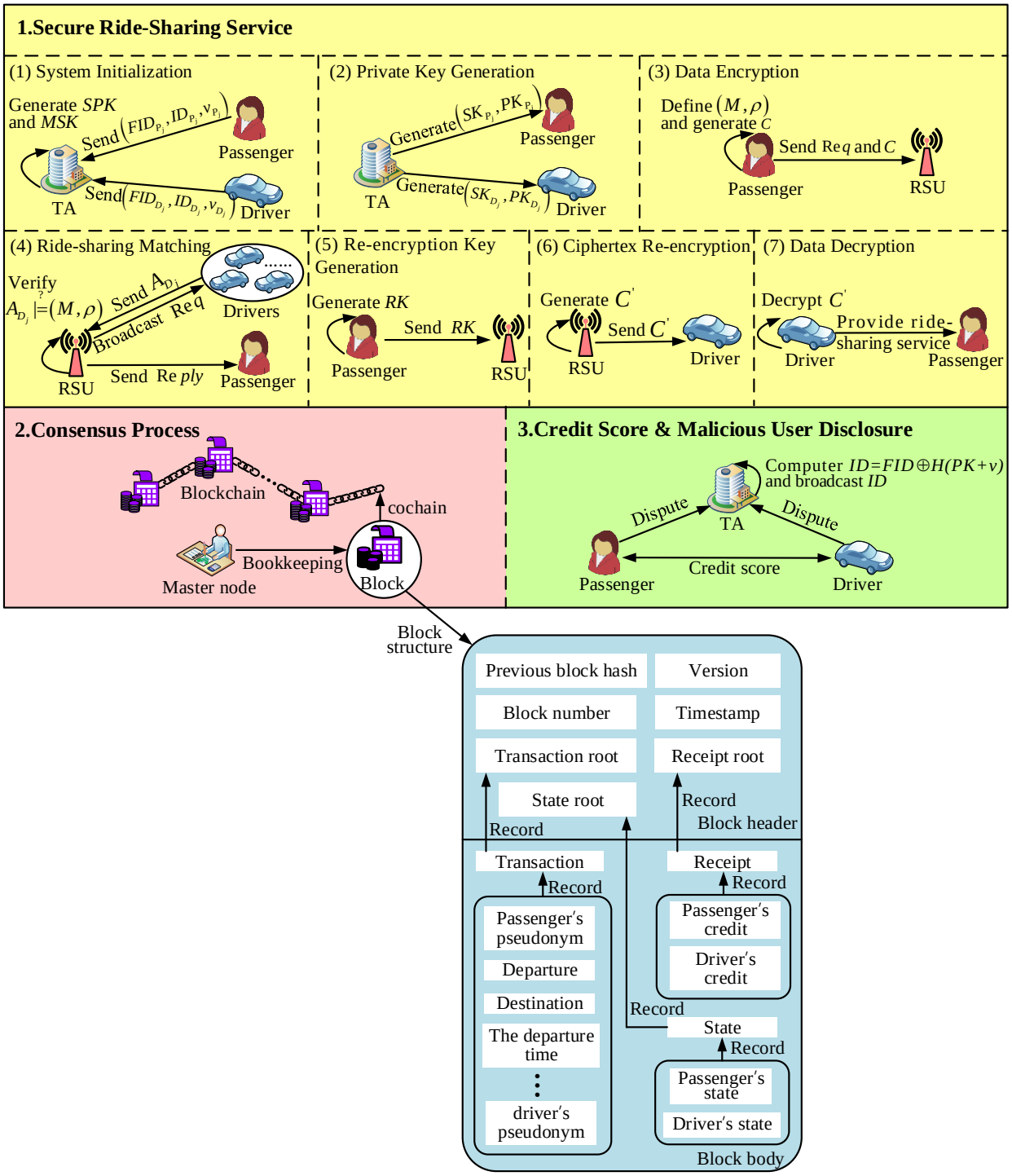

Fig. 2. The implementation process of our proposed scheme

(3) Data encryption

The passenger predefines the access structure for the plaintext m and encrypts m to generate ciphertext C. Afterwards, the passenger sends ride-sharing request $\mathrm{Re}q$ and ciphertext C to the affiliated roadside unit RSU. The specific steps are as follows.

Step 1: The Passenger P_j packages the point of departure, departure time, destination and the latest time at which they want to arrive at their destination to generate plaintext data $m \in \{0,1\}^k$. Then P_j defines the data access structure as (M,ρ), where M is a matrix of $l \times n$ and ρ maps each row vector of matrix M into attributes. P_j randomly selects $b \in Z_p^*$ to calculate $R = g^b$ and $\beta = H_1(Tag)$, where Tag is the label of data.

Step 2: The passenger P_j randomly selects $s', r_2, \ldots, r_q \in Z_p^*$ to form the random vector $r = (s', r_2, \ldots, r_q)^T$. Next, P_j calculates $\lambda_i = M_i r$, where M_i is the *i-th* row vector of matrix M. P_j randomly chooses O from the multiplicativecycle group G_2 to calculate $s = H_1(O, m)$ and $q = H_2(O)$, and then calculates

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

$$\begin{cases} R = OB^{s}, R^{'} = m \oplus q \\ R_0 = h_1^{s}, R_0^{'} = g_1^{s^{'}} \\ R_i = g^{\lambda_i}, R_i^{'} = T_{\rho(i)}^{\lambda_i}, i \in [1,l] \\ R^{''} = g^{H_1\left(e(g,g)^{zs}\right)} \end{cases} \tag{4}$$

Let $s^{''} = s - s^{'}$, P_j randomly selects w_u from Z_p^* to calculate $C_U^{'} = h_2^{s^{''}} Y^{w_u} G_3^{\beta w_u}$ and $C_U = g^{w_u}$. The ciphertext is $C = \left\{(M,\rho), R, R^{'}, R_0, R_0^{'}, R^{''}, \forall i \in [1,l]\left(R_i, R_i^{'}\right), C_U, C_U^{'}\right\}$. P_j sends ride-sharing request $\mathrm{Re}q$ and the ciphertext C to the RSU_j of the place of departure. P_j then pays a certain fees to the smart contract as a collateral, which prevents P_j from sending false requests and ensures P_j has the ability to pay for ride-sharing services.

(4) Ride-sharing matching

Once receiving ride-sharing request $\mathrm{Re}q$, RSU subsequently broadcasts $\mathrm{Re}q$ to the drivers within jurisdiction. The drivers who can provide carpooling services send travel attributes to RSU, which uses fog computing to determine whether travel attributes meet the access structure (M,ρ). If travel attributes of the driver D_j meet the access structure (M,ρ), an appropriate driver D_j is found. Subsequently, RSU_j sends ACK to the driver D_j so that D_j knows he/she is matched to an appropriate passenger P_j. D_j then pays a deposit to the smart contract address after receiving ACK, preventing D_j from sending malicious response requests that would exacerbate network congestion.

(5) Re-encryption key generation

The passenger P_j selects a random number $b \in Z_p^*$ to calculate g^b. Afterwards, P_j encodes $f_{encode}\left(g^b\right)$ and calculates $s_{rk} = H_1\left(O, f_{encode}\left(g^b\right)\right)$. Next, P_j randomly selects $s_{rk}^{'}, w_{rk} \in Z_p^*$ to calculate

$$\begin{cases} s_{rk}^{''} = s_{rk} - s_{rk}^{'}, R_{rk} = OB^{s_{rk}}, R_{rk}^{'} = E\left(g^b\right) \oplus q \\ R_{rk0} = h_1^{s_{rk}}, R_{rk0}^{'} = g_1^{s_{rk}^{'}}, R_{rk_i}^{'} = T_{\rho(i)}^{\beta_i}, i \in [1,l] \\ W_{rk}^{'} = h_2^{s_{rk}^{''}} Y^{w_{rk}} G_3^{\beta w_{rk}}, R_{rk}^{''} = g^{H_1\left(e(g,g)^{zs_{rk}}\right)} \end{cases} \tag{5}$$

thereby $C_{rk}^{'} = \left\{\left(M^{'}, \rho^{'}\right), R_{rk}, R_{rk}^{'}, R_{rk0}, R_{rk0}^{'}, R_{rk}^{''}, i \in [1,l] : R_{rk_i}^{'}, W_{rk}^{'}\right\}$. For all $i \in A$, $rk_{2_i} = Q_i g_1^b = g^a T_i^{a_i} g_1^b$ and $rk_1 = N_1 N_2^{\beta} = Y^{a/x} G_3^{a\beta/x}$ are calculated. The re-encryption key is $RK = \left\{A, \left(M^{'}, \rho^{'}\right), N, rk_1, N_3, \forall i \in A : rk_{2_i}^{'}, Q_i^{'}, C_{rk}^{'}\right\}$. P_j sends the re-encryption key RK to RSU_j and then goes to the place of departure waiting for the driver to arrive. RSU_j encrypts the ciphertext with RK and generates the re-encrypted ciphertext that can be decrypted by the driver D_j.

(6) Ciphertext re-encryption

After receiving the re-encryption key RK sent by the P_j, RSU_j first selects a random number $s_{rk}^{'}, r_2^{'}, r_3^{'}, \ldots, r_q^{'} \in Z_p^*$, then constructs a random vector $r^{'} = \left(s_{rk}^{'}, r_2^{'}, r_3^{'}, \ldots, r_q^{'}\right)^T$ to calculate $\lambda_i^{'} = M_i r^{'}$. RSU_j calculates $R_{rk_i} = g^{\lambda_i^{'}}$ for $\forall i \in [1,l]$ and selects a random number $w_{rk} \in Z_p^*$ to calculate $C_{rku} = g^{w_{rk}}$, thereby $C_{rk} = \left\{C_{rk}^{'}, C_{rku}, \forall i \in [1,l] : R_{rk_i}\right\}$. RSU_j chooses a constant η_i that satisfies $\sum_{\rho(i) \in S} \eta_i M_i = (1,0,\ldots,0)$ and calculates

$$\begin{aligned} L^{'} &= \frac{e\left(C_U^{'}, N_3\right)}{e\left(C_U, rk_1\right)} = \frac{e\left(h_2^{s^{''}} Y^{w_u} G_3^{\beta w_u}, g^{\frac{a}{x}}\right)}{e\left(g^{w_u}, Y^{\frac{a}{x}} G_3^{\frac{\beta a}{x}}\right)} \\ &= \frac{e\left(h_2^{s^{''}}, g^{\frac{a}{x}}\right) e\left(Y^{w_u} G_3^{\beta w_u}, g^{\frac{a}{x}}\right)}{e\left(g^{w_u}, Y^{\frac{a}{x}} G_3^{\frac{\beta a}{x}}\right)} \\ &= e\left(g^{s^{''}}, g^a\right) \end{aligned} \tag{6}$$

RSU_j calculates shared access policies of ciphertext

$$\begin{aligned} L^{''} &= \prod_{i \in A}\left(\frac{e\left(rk_2, R_i\right)}{e\left(Q_i^{'}, R_i^{'}\right)}\right)^{\eta_i} = \prod_{i \in A}\left(\frac{e\left(g^a T_i^{a_i} g_1^b, g^{\lambda_i}\right)}{e\left(g^{a_i}, T_i^{\lambda_i}\right)}\right)^{\eta_i} \\ &= \prod_{i \in A}\left(e\left(g^a g_1^b, g^{\lambda_i}\right)\right)^{\eta_i} = \prod_{i \in A} e\left(g^a, g\right)^{\lambda_i \eta_i} e\left(g_1^b, g\right)^{\lambda_i \eta_i} \\ &= e\left(g^a, g^{s^{'}}\right) e\left(g_1^b, g^{s^{'}}\right) \end{aligned} \tag{7}$$

RSU_j uses $L^{'}$, $L^{''}$, R_0 and N to calculate

$$\begin{aligned} L &= \frac{e\left(R_0, N\right)}{L^{'} L^{''}} = \frac{e\left(h_1^{s}, g^{\frac{z+a}{\alpha}}\right)}{e\left(g^{s^{''}}, g^a\right) e\left(g^a, g^{s^{'}}\right) e\left(g_1^b, g^{s^{'}}\right)} \\ &= \frac{e\left(g^s, g^z\right)}{e\left(g_1^b, g^{s^{'}}\right)} \end{aligned} \tag{8}$$

The re-encrypted ciphertext $C^{'} = \left\{\left(M^{'}, \rho^{'}\right), R, R^{'}, R_0^{'}, R^{''}, L, C_{rk}\right\}$ generated by the RSU_j is sent to the driver D_j.

(7) Data decryption

The driver D_j will decrypt the re-encrypted ciphertext $C^{'}$ only if $C^{'}$ is validated. The steps are as follows.

Step 1: D_j receives the re-encrypted ciphertext $C^{'}$ and verifies the correctness of $C^{'}$. First, D_j calculates $F = Le\left(R_0^{'}, g^b\right) = e\left(g^s, g^z\right)$. If equation $R^{''} = g^{H_1(F)}$ holds, then the re-encrypted ciphertext $C^{'}$ is correct and 1 is output, otherwise $\perp$ is output.

Step 2: After the re-encrypted ciphertext is verified, the driver D_j decrypts $C^{'}$ to obtain the passenger's travel information. The D_j decrypts $C^{'}$ to get $f_{encode}\left(g^b\right)$, decodes to get g^b, and then uses R, L, $R_0^{'}$ and g^b to calculate O according to formula (9).

$$\frac{R}{Le\left(R_0^{'}, g^b\right)} = \frac{Oe(g,g)^{zs}}{\frac{e\left(g^s, g^z\right)}{e\left(g_1^b, g^{s^{'}}\right)} e\left(g_1^{s^{'}}, g^b\right)} = O \tag{9}$$

Step 3: The driver D_j calculates $m = R^{'} \oplus H_2(O)$ and $s = H_1(O,m)$, if both equation $R = Oe(g,g)^{zs}$ and equation $L = e(g,g)^{zs} e\left(R_0^{'}, g^b\right)^{-1}$ hold, then the plaintext is output, otherwise $\perp$ is output. After successful decryption, the driver D_j obtains the passenger's travel information and travels to the place of departure picking up P_j within the specified time-frame. In case no a driver meets the access structure

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

within the jurisdiction area of RSU_j, RSU_j will transmit ride-sharing request to roadside units RSU_{j-1} and RSU_{j+1} in the adjacent area. Roadside units then repeat the aforementioned secure ride-sharing services until an appropriate driver is found to match the passenger.

B. Consensus Process

Compared with the DPoS consensus mechanism, the improved DPoS consensus mechanism uses the whole node voting method to complete the block verification. If and only if the number of votes is greater than the threshold, the block is connected to the consortium blockchain for permanent preservation. A flowchart outlining the improved DPoS consensus mechanism is shown in Fig. 3. The improved DPoS consensus mechanism includes three main components: node selection, block generation, and block verification. Three components are described in detail.

(1) Node selection

The main purpose of node selection is to select 101 accounting nodes that will package data and generate blocks. Roadside unit nodes vote for preselected accounting nodes according to the percentage of their credit value compared to the total credit value. Preselected accounting nodes are equal. The specific node selection process is as follows.

Step 1: Roadside unit nodes vote for preselected accounting nodes according to the proportion of its credit score compared to the total credit value.

Step 2: A check is made of whether the credit proportion σ of roadside unit nodes voted for exceeds the threshold θ, if $\sigma < \theta$, vote again, otherwise go to *step 3*.

Step 3: The number of votes obtained by each roadside unit node is ranked from high to low, roadside unit nodes with more than μ votes are filtered out and added to the node list NL, where μ represents the minimum number of votes needed to become part of accounting nodes. μ avoids nodes with low credit values from being selected to become part of the accounting node list.

Step 4: A check is made of whether the size of the NL is greater than 101. If $NL > 101$, select 101 nodes with the highest number of votes from NL as accounting nodes and add 101 nodes to the accounting node list ANL. Otherwise, return to *step 1* and continue voting until 101 accounting nodes are selected.

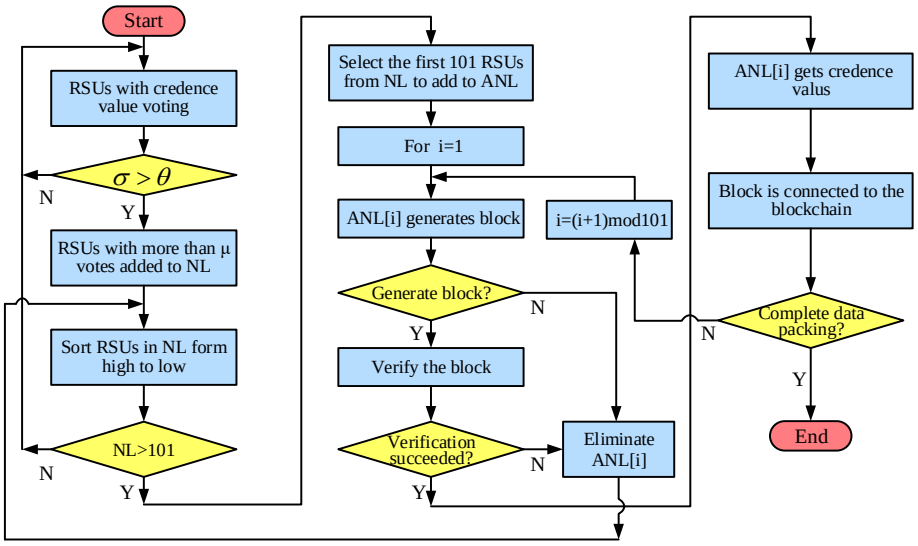

Fig. 3. The improved DPoS consensus flowchart

(2) Block generation

Passengers send their journey's ciphertext and re-encryption key to the roadside unit. Whereas, drivers send the ciphertext of their travel attributes to the roadside unit. 101 preselected accounting nodes can monitor the transaction information in the network and verify the received information. As long as the received information is verified successfully, they will be stored in the buffer pool waiting to be packaged by the master node that is alternately acted by accounting nodes. The master node first verifies whether the equation $i = height \mod 101$ is true, where $height$ is the height of the block to be packaged and i is the order of accounting nodes in ANL. If the equation is true, the master node packs the data stored in the local buffer pool and generates a block, then calculates the block's hash, and finally broadcasts the block to other nodes for block verification. After the block is successfully verified, the master node obtains a credit value as a reward. Each node receives a block, they will automatically start a timer in the meanwhile. Once the timing is over, the block verification is still not completed, which is deemed as block generation failure. If the master node fails to generate a block, it will be removed from ANL. A new accounting node will be elected and added to ANL.

(3) Block verification

The block verification process is shown in Algorithm 1. The master node broadcasts the packed block in the network and the other nodes verify the information in the block after receiving it, such as whether transactions in the block are valid, whether the block's hash is correct, etc. *True* is broadcasted after verification is successful. When a node receives *True* from more than o different nodes, it will broadcast *acknowledgement*. If each node receives *acknowledgement* from more than $2o+1$ different nodes, the

block verification is successful. Then, the current block is linked to the blockchain for permanent storage. After the block is successfully linked to the blockchain, the node status is queried and controlled, mainly to check whether the accounting node needs to be reelected and whether the current round of block verification is successful, so as to prevent a node from constantly being in the state of packing data to generate blocks.

Algorithm 1 Block Verification

```
for RSU_i in the ANL do
  if RSU_i is the master node do
    Generate and broadcast the block;
    if receive more than o True do
      Broadcast acknowledgement ;
        if receive more than 2o+1 acknowledgement do
          Return Verification_Succeeded;
        else
          Return Verification_Failed;
        end if
      else
        Return Verification_Failed;
      end if
  else
    Verify the data in the block;
    if the data is correct do
      Broadcast True ;
    else
        Return Verification_Failed;
    end if
  end if
end for
```

C. Credit Score and Malicious User Disclosure

When the driver delivers the passenger to destination, the driver and the passenger will give each other a score. The higher the passenger's credibility score, the faster roadside units respond to the passenger's requests. The higher the credibility score of the driver, the quicker roadside units verify whether the driver's travel attributes meet the access structure of the passenger's travel data. The smart contract will return the deposit and pay the driver for ride sharing and the remaining amount will be returned to the passenger. In case of a dispute between passengers and drivers, *TA* uses the ride-sharing report stored on the blockchain to trace the source and arbitrate the dispute. *TA* calculates the real identity of the malicious user by $ID = FID \oplus H(PK + v)$ and broadcasts the real identity of the malicious user throughout the network.

VI. Security Analysis and Performance Evaluation

We comprehensively analyze the security of the proposed ride-sharing services scheme in terms of the information interaction process and consortium blockchain network. The results of the security analysis show that our scheme meets the security requirements of ride-sharing services.

A. Security Analysis

(1) Information interaction security

1) Unidirectionality

The re-encryption key includes the passenger's private key that the random number is embedded in and the ciphertext $C_{rk}^{'}$ of the random number under the access structure $(M^{'},\rho^{'})$. Only if the re-encryption key satisfies the access structure (M,ρ), will the roadside unit use the re-encryption key to convert the access structure (M,ρ) of the ciphertext into the access structure $(M^{'},\rho^{'})$ of the re-encrypted ciphertext. Due to the lack of $C_{rk}^{'}$, the roadside unit cannot transform the access structure $(M^{'},\rho^{'})$ of the re-encrypted ciphertext into the access structure (M,ρ) of the ciphertext. Therefore, the roadside unit can use the re-encryption key to transform the ciphertext C under one shared permission into the re-encrypted ciphertext $C^{'}$ of the same plaintext under another shared permission. Whereas, the roadside unit cannot transform the re-encrypted ciphertext $C^{'}$ into the ciphertext C, which meets unidirectionality and irreversibility requirements of the system.

2) Verifiability

The driver obtains the re-encrypted ciphertext $C^{'}$ to calculate $F = Le(R_0^{'}, g^b) = e(g^s, g^z)$ and verifies whether the equation $R^{''} = g^{H_1(F)}$ is true. If the equation is true, $C^{'}$ is decrypted to obtain the plaintext. Otherwise, the driver discards $C^{'}$ and stops decrypting the wrong re-encrypted ciphertext, which reduces calculation overheads and the wastage of network resources.

3) Confidentiality

The process of generating the re-encryption key for a passenger includes two main parts: embedding a random number into the passenger's private key and encrypting a random number under the access policy. The above two parts can be completed by passengers independently without a trusted third party. In addition, $R_0^{''}$ is used in the decryption of the re-encrypted ciphertext, but $R_0^{''}$ is not needed in the decryption of the ciphertext. Therefore, the passenger can use $R_0^{''}$ to decide whether the ciphertext can be re-encrypted, preventing the ciphertext from being re-encrypted into the re-encrypted ciphertext that can be decrypted by unauthorized entities, and avoiding privacy leakage caused by unauthorized data access. Moreover, during the whole information interaction process, the roadside unit can only obtain the ciphertext and cannot obtain the plaintext, ensuring the confidentiality.

4) Anti-collusion attack

The roadside unit as an agent can use the re-encryption key to generate the re-encrypted ciphertext, and then sends it to the driver who meets the passenger's itinerary requirements. However, the roadside unit may also collude with other drivers $D_j^{'}$ who do not meet the passenger's itinerary requirements and use the re-encryption key to encrypt the ciphertext into the re-encrypted ciphertext that $D_j^{'}$ can decrypt. To prevent collusion, we introduced an identifier for the data. The re-encryption algorithm can only be executed if the data identifier embedded in the re-encryption key is equal to the data identifier of the ciphertext. In the process of generating the re-encryption key, the passenger embeds an

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

identifier to control whether the ciphertext can be re-encrypted, which effectively prevents collusion.

5) Anti-substitution attack

We use the randomization method to prevent the roadside unit from replacing the data identifier embedded in the re-encryption key and the ciphertext with another identifier. β is randomized by a/x in the re-encryption key and then $G_3^{a\beta/x}$ is randomize by $Y^{a/x}$. β is randomized by w_u in the ciphertext and then $G_3^{\beta w_u}$ is randomized by $h_2^{s''} Y^{w_u}$.

(2) Consortium Blockchain Network Security

In the process of ride-sharing matching, passengers, roadside units and drivers interact with each other using pseudonyms in the blockchain network. Thus, they do not know each other's real identities. Only when there is a dispute between the passenger and the driver, will the trusted authority use the information stored on the blockchain for digital forensics and arbitration of the dispute. The trusted authority uses the identity list to reveal the real identity of the malicious node. In the worst-case scenario, even if the attacker knows the true identity of the network node, it will not be able to expose the user's private data, such as travel time and route. When passengers send carpooling requests and the ciphertext to roadside units, they also need to send credibility score to the smart contract address as a collateral, preventing passengers from sending false requests and ensuring they can pay for ride-sharing services. The driver also needs to send his credibility score to the smart contract address as a collateral preventing the driver from sending false responses. After the driver delivers the passenger to the destination, the passenger and the driver score each other's credit. The smart contract pays the credibility score to the driver as ride-sharing fees, and returns the deposit credibility score to the driver, as well as returning the remaining credibility score to the passenger. The credibility mechanism improves the credibility of passengers and drivers and provides users with secure and reliable information during data interaction.

Our scheme adopts the improved DPoS consensus mechanism. The DPoS consensus is used to select the top 101 roadside units with the highest credit values as accounting nodes, which in turn act as the master node to package data and generate blocks. Other nodes use PBFT to verify the data in blocks. The current block is connected to the blockchain after verification is successful. The improved DPoS consensus mechanism ensures that one third of the malicious nodes in the blockchain network can still work normally. Assume that there are φ verification nodes in the consortium blockchain and there is a 50% probability that the verification node will become a malicious node. Then the data on the blockchain can be modified in case of at least $\delta = (\varphi - 1)/3$ malicious nodes in the consortium blockchain network. Therefore, the probability of successfully modifying the blockchain is $1/2^{\delta}$. If there are 199 verification nodes in the network, the probability of successfully tampering with the block data is $1/2^{66} = 1.3553 \times 10^{-20}$. So it is basically impossible to tamper with the blockchain, effectively protecting the security and integrity of the data.

B. *Performance Evaluation*

In the proposed secure ride-sharing services based on a consortium blockchain, passengers encrypt the travel data and access structure such as departure place, destination, and departure time to generate the ciphertext. Then passengers send carpooling request and ciphertext to the roadside unit node, which broadcasts a request. Drivers who can ride sharing in the area send their travel attributes to the roadside unit. The roadside unit determines whether the driver's itinerary attributes meet the passenger's access structure of travel data and matches the appropriate carpool driver for the passenger. The passenger's ciphertext is re-encrypted to generate the re-encrypted ciphertext that the driver can decrypt. Passengers go to the departure place waiting for the driver. The driver decrypts the re-encrypted ciphertext to obtain the trip information and picks up the passengers at the departure place. After the driver delivers the passenger to the destination, the passenger and the driver score each other's credit. Once there are conflicts and disputes between the two parties, the trusted authority conducts arbitrate. Table Ⅱ compares the performance of our scheme with existing schemes.

Our proposed ciphertext-policy attribute-based proxy re-encryption algorithm uses R_0'' to control whether the ciphertext can be re-encrypted, preventing mismatched drivers from decrypting the re-encrypted ciphertext. As an agent, the roadside unit only obtains the ciphertext, which prevents private data from being disclosed and protects confidentiality of the data. In addition, the driver first verifies the re-encrypted ciphertext, which avoids the wrong data from being obtained and ensures the authenticity of the re-encrypted ciphertext. Our scheme reduces the network burden and calculation overheads. In order to prevent collusion between drivers and roadside units, we introduced identifiers into the re-encryption key and ciphertext.

TABLE Ⅱ
COMPARISON OF THE PERFORMANCE BETWEEN OUR SCHEME AND OTHERS

Performance	Ref. [17]	Ref. [18]	Ref. [19]	Our scheme
Confidentiality	✓	✓	✓	✓
Distributed structure	×	×	✓	✓
Traceability	✓	×	✓	✓
Verifiability	×	✓	×	✓
Tamper-proof	×	×	✓	✓
Credit mechanism	✓	×	×	✓
Anti-collusion	×	✓	×	✓

We use consortium blockchain technology to verify and store the data in a decentralized structure, which prevents single-point attacks and changes the traditional ride-sharing model of centralized control. The improved DPoS consensus mechanism we propose uses PBFT to verify the validity of block data and reduce the possibility of blocks being tampered with. When a dispute arises, the trusted authority uses the data stored on the blockchain for digital forensics, then arbitrates the dispute and exposes the real identity of the malicious node. The credibility mechanism not only improves the driver's reliability and provides passengers with safe and reliable communication conditions, but also motives roadside units to generate blocks and verify blocks. Table Ⅱ shows that our

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

ride-sharing scheme has advantages over other schemes and is suitable for secure ride-sharing services.

(1) Computational overhead

The main computational overhead incurred during the provision of ride-sharing services is the encryption algorithm, re-encryption key generation, re-encryption algorithm, and decryption. Table III compares the computational overheads of our scheme with the references [46-48]. The T_{e1} represents the exponential operation on the multiplicative cyclic group G_1, the T_{e2} is the exponential operation on the multiplicative cyclic group G_2, and T_b is a bilinear operation. The computational cost of the multiplication operation is far less than the above three operations and can be ignored. J denotes the number of attributes in the shared access policy. The experiment was executed on a computer with an Intel i5 processor with 8G memory and a frequency of 3.0 GHz. The three operations of T_{e1}, T_{e2} and T_b took 1.57 ms, 0.311 ms and 0.1657 ms, respectively.

TABLE III
COMPARISON OF THE COMPUTATIONAL OVERHEADS IN OUR SCHEME AND SIMILAR ALGORITHMS

Scheme	Encryption	Re-encryption key generation	Re-encryption	Decryption
Ref [46]	$(J+2)T_{e1}+T_{e2}+T_b$	$(3J+3)T_{e1}+T_{e2}+T_b$	$(J+2)T_{e1}+T_{e2}+(J+2)T_b$	$T_{e1}+T_{e2}+(J+1)T_b$
Ref [47]	$(3J+3)T_{e1}+2T_{e2}+2T_b$	$(5J+5)T_{e1}+2T_{e2}+2T_b$	$T_{e2}+(3J+4)T_b$	$T_{e2}+(3J+4)T_b$
Ref [48]	$(4J+6)T_{e1}+T_{e2}$	$(4J+8)T_{e1}+T_{e2}$	$(3J+7)T_b$	$T_{e2}+6T_b$
Our scheme	$(2J+7)T_{e1}+T_{e2}+T_b$	$(J+9)T_{e1}+T_{e2}+T_b$	$(J+1)T_{e1}+JT_{e2}+(2J+3)T_b$	$T_{e2}+2T_b$

In the data encryption algorithm, calculating R_0, $R_0^{'}$, C_U and $C_U^{'}$ requires the use of T_{e1} for 7 times, calculating R needs to use T_b and T_{e2} only once each, and then the computational overhead of C_i and $C_i^{'}$ is $2JT_{e1}$. So the computational overhead of the encryption algorithm is $(2J+7)T_{e1}+T_{e2}+T_b$. In the process of generating the re-encryption key, the overhead of calculating rk_1, g_1^b and g^b is $3T_{e1}$, the calculation of $C_{rk}^{'}$ requires the use of T_b and T_{e2} once each, and T_{e1} for $(6+J)$ times. So the computational overhead of generating the re-encryption key is $(J+9)T_{e1}+T_{e2}+T_b$. In the ciphertext re-encryption algorithm, calculating $L^{'}$ and $L^{''}$ requires a total of $(2J+2)$ times T_b and J times T_{e2}. L and C_{rk} need T_b and $(J+1)T_{e1}$ respectively, so the computational overhead is $(J+1)T_{e1}+JT_{e2}+(2J+3)T_b$. After the driver obtains the re-encrypted ciphertext, calculating $e(g,g)^{zs}$ needs to run a bilinear operation and an exponential operation on the multiplication cycle group G_2. $e(R_0^{'},g^b)$ needs to perform a bilinear operation. So the computational overheads are one exponential operation and two bilinear operations, namely, $T_{e2}+2T_b$.

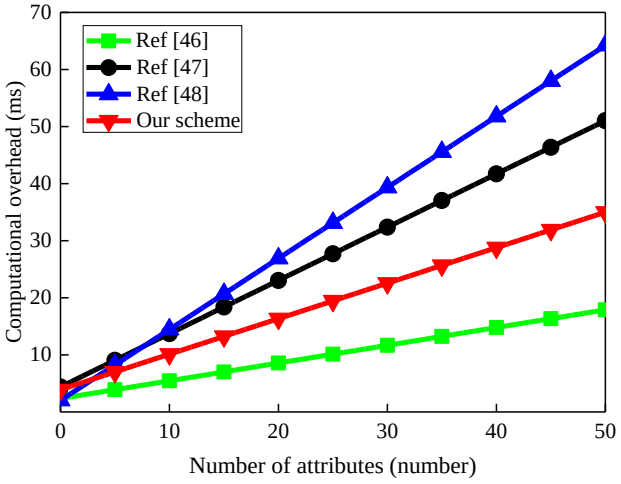

Fig. 4. Computational overhead comparison of various encryption algorithms

Fig. 4 is a comparison of the computational overhead of data encryption process. The results show that computational overheads increase linearly with the number of attributes. Compared with references [47] and [48], our encryption algorithm with shorter encryption time has obvious advantages, which computational overheads increase more slowly with an increase in the number of attributes. Compared with the reference [46], although our encryption algorithm contains parameters with higher computational costs, these parameters prevent the ciphertext from being converted into the re-encrypted ciphertext that can be decrypted by unauthorized users. Our scheme protects the confidentiality as well as security of the data and prevents unauthorized access, which solves the problem of uncontrollable ciphertexts in the reference [46].

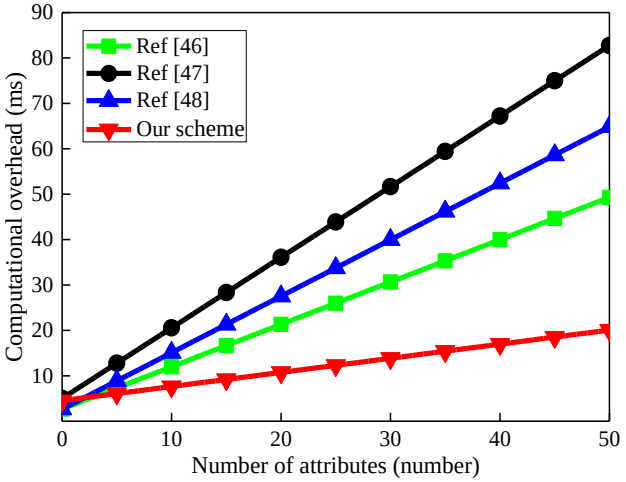

Fig. 5. Computational overhead comparison of the re-encryption key

Fig. 5 is a comparison of the computational cost of calculating the re-encryption key between our scheme and other ride-sharing schemes. There is a linear relationship between the calculation cost of generating the re-encryption key and the number of attributes. With an increase in the number of attributes, the calculation cost of our scheme is the lowest. What's more, the gap between other schemes and our scheme increases gradually. Our scheme only takes 20.076 ms to generate a re-encryption key with 50 attributes. Compared with other three schemes, the computational cost is reduced by 68.02% on average. In addition, we also use the randomization method to prevent the identifier in the re-encryption key from being replaced by the roadside unit. In the reference [47], it is necessary to calculate special sub-key

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

before generating the re-encryption key, resulting in large computation costs.

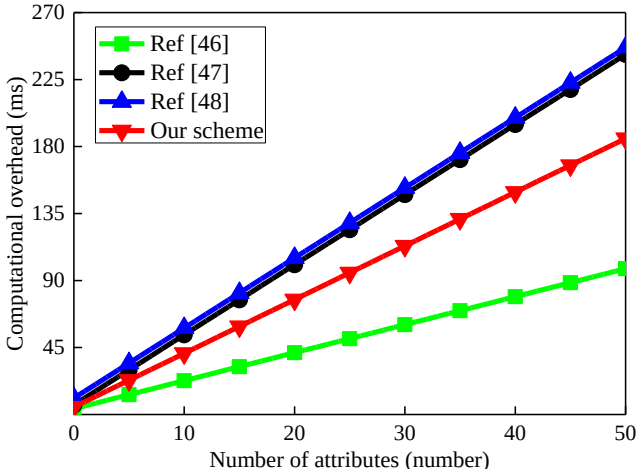

Fig. 6. Computational overhead comparison of re-encryption

A comparison of the computational cost of re-encryption algorithms is shown in Fig. 6. It illustrates that as the number of attributes increases, so does the computational cost. The computational overhead of re-encryption in the present study is less than that in references [47] and [48]. Compared with the reference [46], our scheme has large computational costs, but we embed complex parameters in the re-encrypted ciphertext to achieve verification of the re-encrypted ciphertext. Our scheme prevents the re-encrypted ciphertext from being tampered with or forged during information forwarding and transmission, and solves the problem of unverifiable re-encrypted ciphertext existed in the reference [46].

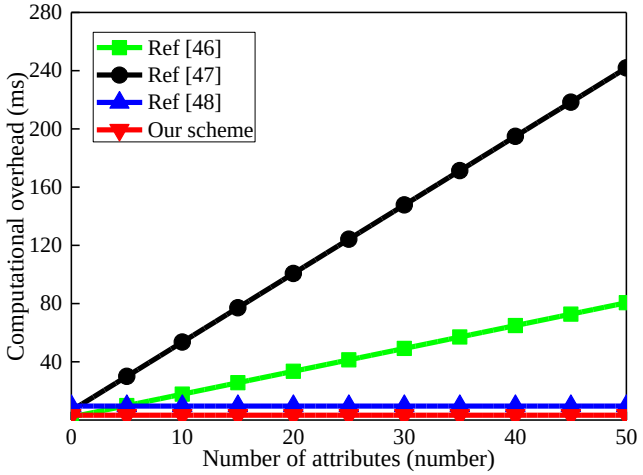

Fig. 7. Computational overhead comparison of decryption

Fig. 7 is a comparison of the computation cost of the decryption algorithm. Fig. 7 illustrates the linear increase in the relationship between the time to decryption and the number of attributes in references [46] and [47]. Decryption time in the reference [48] and our scheme is independent of the number of attributes, that is, the computational cost remains constant. In addition, the computational overhead in the present study is less than that reported in the literature [48]. In the present study, it only takes 3.297 ms to decrypt the re-encrypted ciphertext with 50 attributes. Compared with other three schemes, the computation cost of the present study is reduced by 86.71% on average.

(2) Communication overhead

The communication overhead generated by ride-sharing services mainly includes the private key, ciphertext and re-encrypted ciphertext. A comparison of the communication overhead between our scheme and other three scheme is shown in Table Ⅳ. $|G_1|$ and $|G_2|$ represent the storage space occupied by one element in the multiplication cycle group G_1 and one element in the multiplication cycle group G_2 respectively, $|G_1|$ and $|G_2|$ are 60 bit and 40 bit respectively. The storage space occupied by the element in Z_p^* is very small and is ignored here. K represents the number of attributes contained in the private key and J represents the number of attributes in the shared access policy.

TABLE Ⅳ
COMPARISON OF THE COMMUNICATION OVERHEADS

Scheme	The private key	Ciphertext	Re-encrypted ciphertext
Ref [46]	$(2K+1)\|G_1\|$	$(J+1)\|G_1\|+\|G_2\|$	$(J+3)G_1+(J+2)G_2$
Ref [47]	$(4K+4)\|G_1\|$	$(3J+3)\|G_1\|+2\|G_2\|$	$(3J+4)\|G_1\|+4\|G_2\|$
Ref [48]	$(3K+7)\|G_1\|$	$(3J+6)\|G_1\|+\|G_2\|$	$(3J+7)\|G_1\|+3\|G_2\|$
Our scheme	$(2K+4)\|G_1\|$	$(2J+5)\|G_1\|+\|G_2\|$	$(2J+7)\|G_1\|+3\|G_2\|$

In our scheme, in the process of generating the private key, $N, N_1, N_2, N_3 \in G_1$ and each attribute in the private key contains $Q_i, Q_i^{'} \in G_1$. So the communication cost of the private key is $(2K+4)|G_1|$. In the ciphertext, $R_0, R_0^{'}, R^{''}, C_U, C_U^{'} \in G_1$ and $R \in G_2$. Each attribute in the shared access policy contains $R_i, R_i^{'} \in G_1$, so the communication overhead of the ciphertext is $(2J+5)|G_1|+|G_2|$. C_{rk} in the re-encrypted ciphertext represents the ciphertext of g^b, so the communication cost of C_{rk} is $(2J+5)|G_1|+|G_2|$. However $R_0, R_0^{'} \in G_1$ and $R, L \in G_2$. Therefore, the communication cost of the re-encrypted ciphertext is $(2J+7)|G_1|+3|G_2|$.

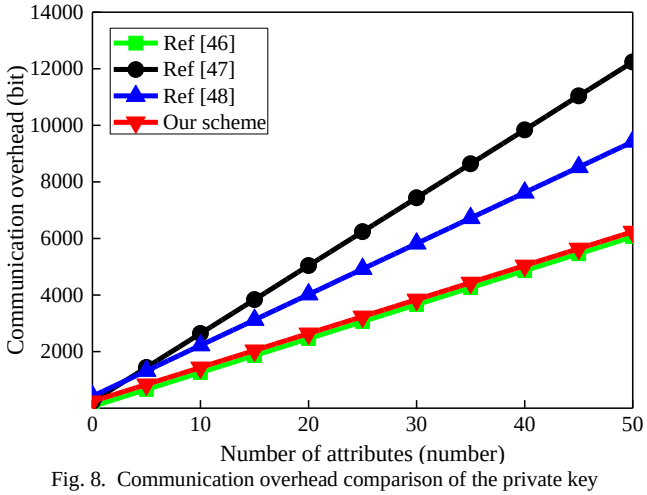

Fig. 8. Communication overhead comparison of the private key

Fig. 8 illustrates a comparison of the communication cost of the private keys. The results show that the communication cost of the private keys increases with an increase in the number of attributes. However, in our scheme, the increase is less than in references [46] and [47]. Compared with the reference [48], the communication cost of the private key in the present paper is $3|G_1|$ more than that in the reference [46], but our scheme can prevent collusion between roadside units and drivers who conform to the access structure of the re-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3023920, IEEE Internet of Things Journal

encrypted ciphertext. Additionally, the communication overhead of generating the private key with 50 attributes is reduced by 26.2%.

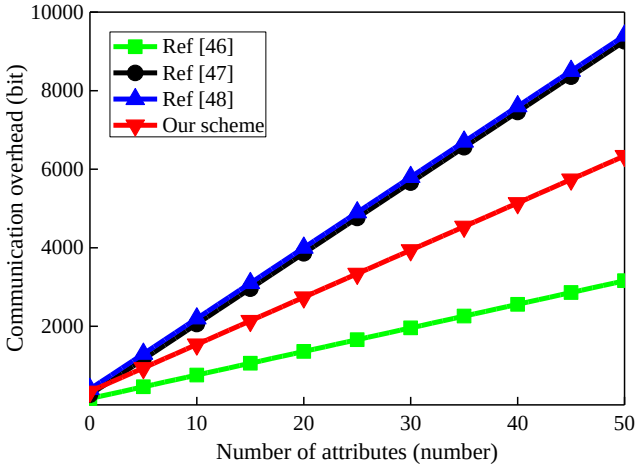

Fig. 9. Communication overhead comparison of the ciphertext

A comparison of the communication cost of the ciphertexts is shown in Fig. 9. As the number of attributes increases, our scheme has obvious advantages over references [47] and [48]. Although the communication cost of our scheme is greater than that of the reference [46], the data identifier and the subitem in the ciphertext prevents the roadside unit from encrypting other data of passengers with the re-encryption key, which achieves control of the ciphertext. Controllability of the ciphertext is not included in the reference [46].

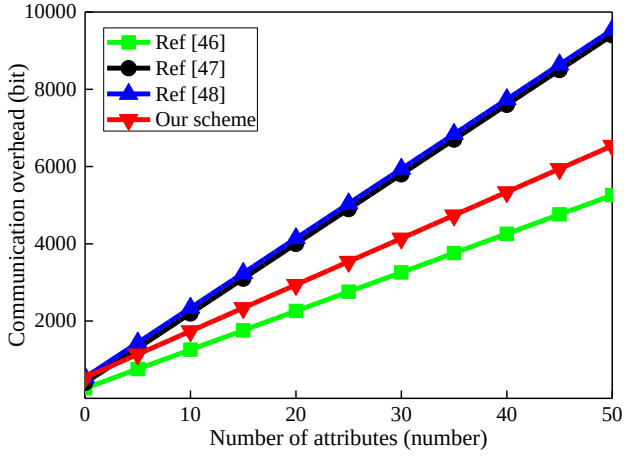

Fig. 10. Communication overhead comparison of the re-encrypted ciphertext

A comparison of the communication cost of the re-encrypted ciphertext is shown in Fig. 10. As the number of attributes increases, our scheme has obvious advantages over references [47] and [48]. Although our scheme has large communication costs compared with the reference [46], we add C_{rk} to the re-encrypted ciphertext to achieve verification of the re-encrypted ciphertext. There is no verification in the reference [46]. Additionally, the communication overhead of generating re-encrypted ciphertext with 50 attributes is reduced by 12.51%.

(3) Blockchain delay

The consensus mechanism adopted in the paper generates a block every 4 seconds without block confirmation time. Therefore, it takes 4 seconds to generate a valid block. Nevertheless, it takes 2 seconds for the DPoS (Delegated Proof of Stake) consensus to generate a block and another 12 seconds for the confirmation time. The PoW (Proof of Work) consensus generates a block in 10 minutes with an additional 60 minutes for block confirmation time. Compared with DPoS and PoW, our consensus mechanism is faster to generate blocks as well as confirm blocks, so blockchain delay is less.

VII. Conclusion and Future Work

Secure ride-sharing services based on a consortium blockchain proposed in this paper not only ensures the confidentiality, security and privacy of information interaction process, but also changes the centralized structure of existing ride-sharing systems, which prevents single point collapse and information monopoly leading to malicious abuse or illegal selling of user data. The roadside unit uses attribute-based proxy re-encryption algorithm to match the appropriate driver for the passenger based on the driver's attribute set as well as the access structure. Attribute-based proxy re-encryption algorithm we proposed meets requirements of unidirectionality, verifiability and confidentiality. It can prevent roadside unis from colluding with drivers satisfying the access structure, so as to avoid the leakage of passengers' privacy data. The improved DPoS consensus mechanism validates ride-sharing records stored in the block, which ensures data integrity and tamper resistance. The data stored on the blockchain can be used as the basis for arbitration in the event of a dispute. The credibility mechanism we designing improves the credibility of the passengers and drivers, which provides a reliable and secure information interaction environment. Security analysis and performance evaluation indicate that our scheme is more secure and efficient than existing schemes. Therefore, our scheme provides a certain theoretical basis and research value for ride-sharing services, which is beneficial to enhance privacy protection and data security, such that the service quality of ride-sharing can be improved.

For the future work, we will strive to propose an attribute-based proxy re-encryption algorithm with constant computational costs as well as communication overheads. Furthermore, due to the time-consuming of bilinear pairing, we consider to propose an attribute-based proxy re-encryption algorithm without bilinear pairing on the premise of ensuring security. Last, the public blockchain is prevalent on account of complete decentralization, still we need to increase the speed of consensus and prevent hackers from tracking the real identity of users based on transaction information while the public blockchain is applied to ride-sharing services.

References

[1] Y. Wang, J. B. Gu, S. Y. Wang, and J. Wang, "Understanding consumers' willingness to use ride-sharing services: The roles of perceived value and perceived risk," *Transp. Res. Part C Emerg. Technol.*, vol. 15, pp. 504-519, Aug. 2019.
[2] https://rideshareapps.com/2015-rideshare-infographic/.
[3] S. Li, H. Tavafoghi, K. Poolla, and P. Varaiya, "Regulating TNCs: should Uber and Lyft set their own rules?" *Transp. Res. Part B Meth.*, vol. 129, pp. 193-225, Nov. 2019.
[4] https://techcrunch.com/2017/12/20/chinas-didi-chuxing-raises-4b/.
[5] Y. Guo, X. T. Li, and X. H. Zeng, "Platform Competition in the Sharing Economy: Understanding How Ride-Hailing Services Influence New Car Purchases," *J. Manag. Inf. Syst.*, vol. 34, no. 4, pp. 1043-1070, Oct.

2019.

[6] Pedro M. d'Orey and M. Ferreira, "Can ride-sharing become attractive? A case study of taxi-sharing employing a simulation modelling approach," *IET Intell. Transp. Syst.*, vol. 9, no. 2, pp. 210-220, Feb. 2015.

[7] M. Zhu, X. Y. Liu, and X. D. Wang, "An Online Ride-Sharing Path-Planning Strategy for Public Vehicle Systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 2, pp. 616-627, Feb. 2019.

[8] J. B. Ni, K. Zhang, X. D. Lin, H. M. Yang, and X. M. Shen, "AMA: Anonymous Mutual Authentication with Traceability in Carpooling Systems," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kuala Lumpur, Malaysia, 2016, pp. 1-6.

[9] A. B. Sherif, K. Rabieh, M. Mahmoyd, and X. H. Liang, "Privacy-Preserving Ride Sharing Scheme for Autonomous Vehicles in Big Data Era," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 611-618, Apr. 2017.

[10] C. Caballero-Gil, C. Caballero-Gil, J. Molina-Gil, F. Martín-Fernández, and V. Loia, "Trust-Based Cooperative Social System Applied to a Carpooling Platform for Smartphones," *Sensors (Basel)*, vol. 17, no. 2, pp. 245-257, Feb. 2017.

[11] P. Hallgren, C. Orlandi, and A. Sabelfeld, "PrivatePool: Privacy-Preserving Ridesharing," in *Proc. IEEE Symp. Comput. Sec. Found. (CSF)*, Santa Barbara, CA, USA, 2017. pp. 276-291.

[12] https://www.forbes.com/sites/ronhirson/2015/03/23/uber-the-big-data-company/#3f377e0918c7.

[13] Y. C. Luo, X. H. Jia, S. J. Fu, and M. Xu, "pRide: Privacy-preserving Ride-matching over Road Networks for Online Ride Hailing Service," *IEEE Trans. Inf. Foren. Sec.*, vol. 14, no.7, pp. 1791-1802, Jul. 2019.

[14] H. N. Yu, X. H. Jia, H. L. Zhang, X. Z. Yu, and J. G.Shu, "PSRide: Privacy-Preserving Shared Ride Matching for Online Ride Hailing Systems," *IEEE Trans. Depend. Secure Comput.*, to be published. DOI: 10.1109/TDSC.2019.2931295.

[15] F. W. Wang, H. Zhu, X. M. Liu, R. X. Lu, F. H. Li, H. Li, and S. N. Zhang, "Efficient and Privacy-preserving Dynamic Spatial Query Scheme for Ride-hailing Services," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11084-11097, Nov. 2018.

[16] .H. N. Yu, J. G. Shu, Xiaohua Jia, H. L. Zhang, and X. Z. Yu, "IpRide: Lightweight and Privacy-Preserving Ride Matching over Road Networks in Online Ride Hailing Systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 10418-10428, Nov. 2019.

[17] A. Pham, I. Dacosta, B. Jacot-Guillarmod, K. Huguenin, T. Hajar, F. Tramèr, V. Gligor, and J. Hubaux, "PrivateRide: A Privacy-Enhanced Ride-Hailing Service," in *Proc. 17th Privacy Enhancing Technol. Symp.*, 2017, pp. 38-56.

[18] Y. Y. He, J. B. Ni, X. Y. Wang, B. Niu, F. H. Li, and X. M. Shen, "Privacy-Preserving Partner Selection for Ride-Sharing Services," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 5994-6005. Jul. 2018.

[19] M. Li, L. H. Zhu, and X. D. 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.

[20] H. J. Zhang, E. D. Deng, H. J. Zhu, and Z. F. Cao, "Smart contract for secure billing in ride-hailing service via blockchain," *Peer Peer Netw. Appl.*, vol. 12, no. 5, pp. 1346-1357, Sep. 2019.

[21] W. C. Zhao, Y. J. Qin, D. Yang, L. J. Zhang, and W. T. Zhu, "Social Group Architecture Based Distributed Ride-Sharing Service in VANET," *Int. J. Distrib. Sens. Netw.*, vol. 10, no. 3, pp. 1-8, 2014.

[22] M. Nabil, A. Sherif, M. Mahmoud, A. Alsharif, and M. Abdallah, "Efficient and Privacy-Preserving Ridesharing Organization for Transferable and Non-Transferable Services," *IEEE Trans. Depend. Secure Comput.*, to be published. DOI: 10.1109/TDSC.2019.2920647.

[23] M. M. Li and L. C. Wang, "Privacy Preservation of Location Information Based on MinHash Algorithm in Online Ride-Hailing Services," in *Proc. 6th Int. Conf. Adv. Cloud and Big Data (CBD)*, Lanzhou, China, 2018, pp. 257-262.

[24] V. A. Memos, K. E. Psannis, Y. Ishibashi, B. Kim, and B. B. Gupta, "An Efficient Algorithm for Media-based Surveillance System(EAMSuS) in IoT Smart City Framework," *Future Gener. Comput. Syst.*, vol. 83, pp. 619-628, Jun. 2018.

[25] K. E. Psannis, C. Stergious, and B. B. Gupta, "Advanced Media-based Smart Big Data on Intelligent Cloud Systems," *IEEE Trans. Sustainable Comput.*, vol. 4, no. 1, pp. 77-87, Mar. 2019.

[26] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," https://bitcoin.org/bitcoin.pdf.

[27] C. Stergious, K. E. Psannis, B. Kim, and B. Gupta, "Secure integration of IoT and Cloud Computing," *Future Gener. Comput. Syst.*, vol. 78, no. 3, pp. 964-975, Jan. 2018.

[28] C. Stergiou and K. E. Psannis, "Recent Advances Delivered by Mobile Cloud Computing and Internet of Things for Big Data Applications: A Survey," *Int. J. Netw. Manag.*, vol. 21, no. 3, pp. 1-12, 2017.

[29] C. Stergiou, K. E. Psannis, A. P. Plageras, Y. Ishibashi, and B. Kim, "Algorithms for Efficient Digital Media Transmission over IoT and Cloud Networking," *J. Multimed. Inf. Syst.*, vol. 5, no. 1, pp. 27-34, Mar. 2018.

[30] M. Belotti, N. Božić, G. Pujolle, and S. Secci, "A Vademecum on Blockchain Technologies: When, Which and How," *IEEE Commun. Surv. Tut.*, vol. 21, no. 4, pp. 3796-3838, Jul. 2019.

[31] W. L.Yang, E. Aghasian, S. Garg, D. Herbert, L. Disiuta, and B. Kang, "A Survey on Blockchain-Based Internet of Things: Performance Analysis and Optimal Communication Node Deployment," *IEEE Access*, vol. 6, no. 3, pp. 5791-5802, Jun. 2019.

[32] Y. Y. Zhang, S. Kasahara, Y. L. Shen, X. H. Jiang, and J. X. Wan, "Smart Contract-Based Access Control for the Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1594-1605, Jun. 2018.

[33] J. L. Pan, J. Y. Wang, A. Hester, I. Alqerm, Y. N. Liu, and Y. Zhao, "EdgeChain: An Edge-IoT Framework and Prototype Based on Blockchain and Smart Contracts," *IEEE Internet Things J.*, vol. 6, no.3, pp. 4719-4732, Jun. 2019.

[34] X. H. Zhang and D. Wang, "Adaptive Traffic Signal Control Mechanism for Intelligent Transportation Based on a Consortium Blockchain," *IEEE Access*, vol. 7, pp. 97281-97295, Jul. 2019.

[35] D. Wang and X. H. Zhang, "Secure Data Sharing and Customized Services for Intelligent Transportation Based on a Consortium Blockchain," *IEEE Access*, vol. 8, pp. 56045-56059, Mar. 2020.

[36] K. K. Gai, Y. L. Wu, L. H. Zhu, L. Xu, and Y. Zhang, "Permissioned Blockchain and Edge Computing Empowered Privacy-Preserving Smart Grid Networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7992-8004, Oct. 2019.

[37] S. Wang, J. Wang, X. Wang, T. Y. Qiu, Y. Yuan, L. W. Ouyang, Y. Y. Guo, and F. Y. Wang, "Blockchain-Powered Parallel Heaithcare Systems Based on the ACP Approach," *IEEE Trans. Comput. Soc. Syst.*, vol. 5, no. 4, pp. 942-950, Dec. 2018.

[38] M. M. Cui, D. Z. Han, and J. Wang, "An Efficient and Safe Road Condition Monitoring Authentication Scheme Based on Fog Computing," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 9076-9084, Oct. 2019.

[39] C. Tartary, S. Zhou, D. Lin, H. Wang, and J. Pieprzyk. "Analysis of bilinear pairing-based accumulator for identity escrowing," *IET Inf. Secur.*, vol. 2, no. 4, pp. 99-107, Dec. 2008.

[40] I. Damgard and R. Thorbek, "Linear Integer Secret Sharing and Distributed Exponentiation," in *Proc. Int. Workshop on Public Key Cryptography*, Berlin, Germany, 2006, pp. 75-90.

[41] J. Liu, M. Xian, H. M. Rong, and H. Rong, "Optimization method for attribute-based cryptographic access control in mobile cloud computing," *Journal on Communications*, vol. 39, no. 7, pp. 39-49, Jul. 2018.

[42] L. Touati and Y. Challal, "Collaborative KP-ABE for cloud-based Internet of Things applications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kuala Lumpur, Malaysia, 2016, pp. 13-20.

[43] C. S. Feng, W. P. Luo, Z. G. Qin, D. Yuan, and L. P. Zou, "Attribute-based proxy re-encryption scheme with multiple features," *Journal on Communications*, vol. 40, no. 6, pp. 177-189, Jun. 2019.

[44] M. Blaze, G. Bleumer, and M. Strauss, "Divertible Protocols and Atomic Proxy Cryptography," in *Proc. Int. Conf. Theory and Appl. of Cryptogr. Techn.*, Berlin, Germany, 1998, pp. 127-144.

[45] D. Larimer, "Delegated Proof-of-State white paper," https://bitfarm.io/.

[46] X. H. Liang, Z. F. Cao, H. Lin, and J. Shao, "Attribute Based Proxy Re-encryption with Delegating Capabilities," in *Proc. 4th Int. Symp. Inf., Comput., Commun. Secur.*, 2009, pp. 276-286.

[47] Y. H. Zhang, J. Li, X. F. Chen, and H Li, "Anonymous attribute-based proxy re-encryption for access control in cloud computing," *Secur. Commun. Netw.*, vol. 9, no. 14, pp. 2397-2411, 2016.

[48] H. J. Yin and L. Y. Zhang, "Security Analysis and Improvement of An Anonymous Attribute-Based Proxy Re-encryption," in *Proc. Int. Conf. Secur. Priv., and Anonymity in Comput., Commun., and Storage*, 2017, pp. 344-352.