Hindawi
Security and Communication Networks
Volume 2022, Article ID 8296608, 13 pages
https://doi.org/10.1155/2022/8296608

Research Article

BBARHS: Blockchain-Based Anonymous Ride-Hailing Scheme for Autonomous Taxi Network

Kun Wang[1], **Mingzhe Liu**[1], **Jianping Wang**[2], **Min Wu**[3] **and Feixiang Zhao**[1]

[1]*State Key Laboratory of Geohazard Prevention and Geoenvironment Protection Technology, Chengdu, Sichuan, China*
[2]*Petrochina Southwest Oil and Gas Field Company Northeast Sichuan Gas District, Chengdu, Sichuan, China*
[3]*Beijing Institute of Computer Technology and Applications, Beijing, China*

Correspondence should be addressed to Mingzhe Liu; liumz@cdut.edu.cn

Received 1 December 2021; Revised 9 May 2022; Accepted 27 May 2022; Published 9 August 2022

Academic Editor: Zhili Zhou

In the past few years, ride-hailing platforms such as Uber, Waymo, and Baidu have built their own autonomous taxi system. Unlike public transit services, ride-hailing platforms raise severe privacy issues. To provide excellent autonomous taxi service, some significant security and privacy problems must be addressed. In this study, we present the security and privacy threats and first proposed blockchain-based anonymous ride-hailing scheme (BBARHS) for autonomous taxi network. We give the formal system model and security model of BBARHS. Then, we outline the concrete BBARHS scheme by making use of Monero and some efficient crypto tools. Through security analysis and performance analysis, the designed scheme is provably secure and efficient. The analysis results also show the designed BBARHS scheme is practical for autonomous taxi network.

1. Introduction

With the development of information technology and AI, autonomous vehicles (AVs) come true. One of the most discussed potential use cases of AVs is RHS (ride-hailing service). AVs and public transit would cut traffic by 90%. It could be 10 times cheaper to take E-AVs taxi than to own a car by 2030 [1]. Waymo, a company owned by Google, officially obtained the first commercial automatic driving taxi service license and took the lead in launching relevant services in Phoenix, the United States [2]. Autonomous Lexus has been tested in California, Michigan, and Japan and preparing for real-world use during the 2020 Tokyo Olympics. Chinese ride-hailing giant Didi expects at least 10% of vehicles to be highly autonomous by 2025, and fully autonomous by 2030 [3]. But in fact, under the existing technical conditions, there are still many challenges and difficulties for the autonomous taxi to achieve commercial scale on the actual road. Without the passengers' knowledge, Uber collected information including passengers' names, e-mails, boarding locations, spending amounts, etc. Autonomous taxi network consists of riders, RHS, and AVs. An autonomous taxi system needs to meet various indicators, for example, performance, safety, and stability [4, 5].

Over the past decade, blockchain as the backbone of bitcoin has experienced a rapid development [6]. Many researchers explore blockchain application scenarios [7–9]. A blockchain is essentially a distributed database of records, or public ledger of all transactions or digital events that have been executed and shared among participating parties. Each transaction in the public ledger is verified by the consensus of a majority of the participants in the system. Once entered, information can never be erased. The blockchain contains a certain and verifiable record of every single transaction ever made [10]. The basic structure of blockchain is shown in Figure 1.

The blockchain uses digital signatures to determine the identity of the sender of information. A digital signature is a digital string that can only be generated by the sender of the message, and this digital string is also an effective proof of the authenticity of the message sent by the sender. It is a kind of ordinary physical signature similar to that written on paper, but it is realized by the technology in the field of public key encryption, which is used to authenticate digital

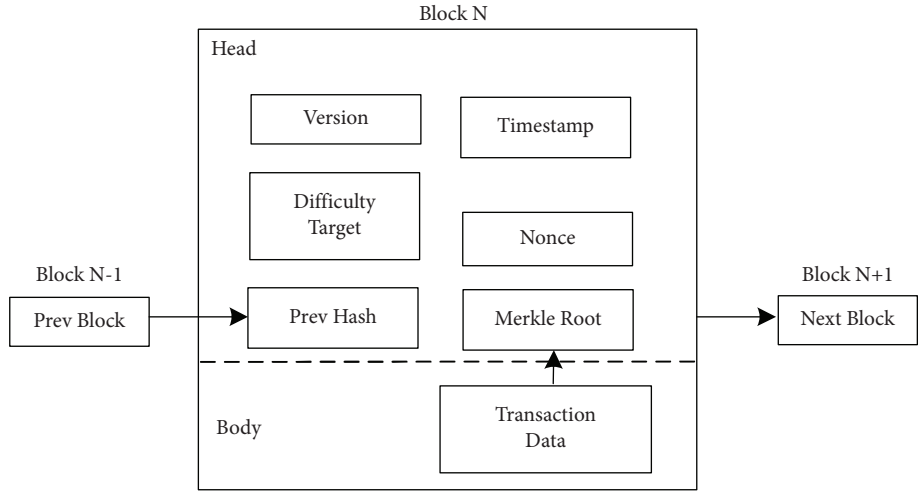

Figure 1: The basic structure of blockchain.

information. Bitcoin uses an elliptic curve digital signature algorithm (ECDSA) to generate public and private keys for accounts and to verify transactions and blocks. The ECDSA is an analog of the digital signature algorithm (DSA) using elliptic curve cryptography (ECC).

The cryptographic hash function is a type of hash function. An arbitrary amount of data input of this hash function is usually called a message, and its fixed-size output result is often called a hash value. The secure hash algorithm (SHA) is a series of cryptographic hash functions designed by the National Security Agency (NSA) and published by the National Institute of Standards and Technology (NIST), including SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 variants. Bitcoin uses SHA-256. That is, no matter how many bits the original data have; as long as the hash operation is passed, the length of the result is fixed as 256 bits.

Representative projects of public chains such as bitcoin and ethereum use POW and POS consensus algorithms. Miners use SHA-256 to calculate the hash value that meets the difficulty target value (starting with N zeros). Representative projects of the alliance chain such as hyperledger fabric use PBFT as the consensus calculation method. Nodes use digital signatures to directly exchange information to reach a consensus. Blockchain guarantees the authenticity, privacy, and security of information through cryptographic tools and consensus algorithms.

Riders need to exchange information between multiple participants, and privacy protection and the authenticity of information are extremely important. This article aims to use the characteristics of blockchain technology to solve these problems in the field of autonomous taxi.

1.1. Motivation. Security and privacy problems are the two major problems that need to be solved urgently for autonomous taxi network. In order to be able to provide services, the system needs to manage the location information of autonomous taxi, rider standards, service scenarios, how much fuel there is, and so on. After that, an autonomous taxi is allocated based on the service information requested by the rider. Finally, the rider needs to pay for the service. In these processes, how to protect the privacy of rider and how to ensure the safety of autonomous taxi are very important. If privacy of rider and autonomous taxi is leaked or autonomous taxi has service standards mismatch or safety issues, then it can cause reputation and economic loss and increase the difficulty for the autonomous taxi to be accepted by the market. For example, several employees of Didi used their authority to check user travel records and make illegal profits [11]. Thus, blockchain-based anonymous ride-hailing services are of great significance to the development of RHS.

In real life, there may be problems such as overloading of autonomous taxi to get more fees and mismatches in many service standards. This will cause safety issues and disputes between the rider and the platform. How to achieve transparent and credible management is a problem that the platform has been solving. In addition, autonomous taxi platforms may collect riders' data. How to ensure the privacy of riders while providing services is also worthy of attention.

In order to resolve the privacy issues between the rider and autonomous taxi, as well as disputes over autonomous service standards, we propose a blockchain-based anonymous ride-hailing scheme (BBARHS) for autonomous taxi network.

1.2. Related Works. Autonomous vehicles are one of the most anticipated technological developments of our time, and they have potential wide-ranging social influence [12]. However, there are serious privacy issues when users leak location information to the taxi server.

In 2019, Yu et al. proposed a lightweight and privacy-preserving ride-matching scheme, called lpRide, to address the issue of protecting rider's location privacy during ride matching [13]. In 2017, Zhang et al. used a third-party database and data interactive review platform to protect personal privacy. Online taxi service software platforms and other profit-making organizations can rent data interactive review platforms to announce to the public their improvements

in personal privacy protection. However, this article does not provide a specific implementation and simulation of the system [11]. In 2018, Khazbak et al. proposed an enhanced solution that relies on enhanced driving matching and temporary stealth algorithms. The enhanced solution provides riders with personalized location privacy while limiting the loss of matching accuracy [14]. In 2017, Pham et al. proposed Private Ride, a practical solution that uses well-established privacy and cryptographic tools to enhance location privacy for the participants, while preserving the convenience and functionality offered by the current system [15]. In 2017, Pham et al. proposed ORide (Oblivious Ride), a privacy protection RHS based on somewhat-homomorphic encryption and with optimized functions such as ciphertext packaging and conversion processing, to address privacy issues in ride-hailing service [16].

Many researchers have proposed different privacy-enhancing solutions for taxi service. However, according to our literature review, little work exists in the area of privacy and security for autonomous taxi network. Thus, we propose a blockchain-based anonymous ride-hailing service scheme in autonomous taxi network.

Based on the summary of previous research, this study studied privacy and safety issues in autonomous taxi network. Our contributions are listed as follows:

(1) For the first time, we propose anonymous ride-hailing service for autonomous taxi network by making use of blockchain. We give a formal system model and security model. Our system can protect rider's privacy and platform's privacy simultaneously.

(2) The unlinkability between the payer address and the payee address is satisfied. Nontraceability is also satisfied. When the payment is executed, no one can find the relationship between them. No one can find the address of the payer.

(3) We first propose the blockchain-based anonymous ride-hailing service scheme (BBARHS). The proposed BBARHS scheme can be proved to be secure in the random oracle model. The analysis and the simulation show that the scheme is effective and practical for secure ride-hailing service of autonomous taxi network.

1.3. Organization. The rest is organized as follows: Section 2gives some cryptographic background, which includes PKCs in PKI, blockchain, etc. The system model and the security model of the BBARHS scheme are introduced in Section 3. Our scheme is presented in Section 4. The analysis and the simulation are in Section 5. Finally, this study is concluded in Section 6.

2. Background

This section will briefly explain the cryptographic background involved in the application of this article, the application of PKI and its PKCs and PKCs on the blockchain, ECC encryption algorithms and bilinear pairs, ring signatures, group signatures, and Monero.

2.1. PKCs on PKI and PKCs on Blockchain. PKCs are a set of public key cryptography standards developed by RSA data security company and its partners, which promote secure transaction and data transmission on the network, such as e-commerce and confidential mail [17]. It includes a series of related protocols, such as certificate application, certificate renewal, certificate revocation form release, certificate content extension, digital signature, and digital envelope format. PKC is a cryptosystem with a pair of keys, a widely spread public key and a private key known only by itself. The essence of blockchain is a decentralized database. It has no trusted third party. Therefore, there is no organization like CA (certificate authority) and PKI involved in the blockchain [18]. In the PKCs of blockchain, we use public key encryption to create a key pair to control the acquisition of assets. The key pair includes a private key and a unique public key derived from it. The public key corresponds to the address on the blockchain, in which the assets are stored, while the private key is used to sign transactions of the assets [19].

2.2. ECC and Bilinear Pairings. ECC (elliptic curve cryptosystem) is a public key cryptosystem with shorter key length than RSA. It is realized by special multiplication of a specific point on an elliptic curve. It takes advantage of the fact that the inverse operation of this multiplication operation is very difficult to achieve a good effect of encryption.

Elliptic Curve on the Finite Field $\mathbb{F}_q$. For fixed a and b, all points (x, y) satisfying the shape of the equation $y^2 \equiv x^3 + ax + b \pmod{p}$, $(a, b, x, y \in \mathbb{F}_q, 4a^3 + 27b^2 \pmod{p} \neq 0)$ are set, plus a zero point and an infinite point 0, where a, b, x, and y are all take a value on the finite field $\mathbb{F}_q$, that is, take a value on $0, 1, 2 \ldots p - 1$. P is a prime number.

The discrete logarithm problem in the elliptic curve group refers to the problem of knowing P and Q in the group to solve the equation $kP = Q$ in the value of k. It is easy to find Q from k and P, but it is difficult to find k from P and Q. This is the discrete logarithm problem on the elliptic curve, which can be applied to public key cryptosystems.

In addition, the elliptic curve can be used to realize the Diffie–Hellman key exchange. By selecting a base field $\mathbb{F}_q$ and two parameters a, b, the points on the elliptic curve and the infinity points form the Abel group $\mathbb{E}_p(a, b)$.

Taking a generator $G = (x_1, y_1)$ of $E_p(a, b)$, the order of G is prime number n. $\mathbb{E}_p(a, b)$, G, and n are public parameters. User A and B start to exchange keys. A randomly selects an integer $k_A < n$, keeps k_A, calculates $P_A = k_A G$ to generate E, and sends a point on (a, b) to B. B similarly selects the secret k_B and calculates P_B and sends it to A. A and B generate the shared secret key of both parties by $K = k_A P_B$ and $K = k_B P_A$ respectively, in fact $K = k_A, P_B = k_A, (k_B G) = k_B, (k_A G) = k_B P_A$. If the attacker wants to obtain K, then he must find k_A from P_A and G, or find k_B from P_B and G; that is, the discrete logarithm on the elliptic curve is required. Thus, it is not feasible.

In 1983, Menezes et al. defined a special elliptic curve on a finite field, that is, a supersingular curve, and pointed out that these elliptic curves use the elliptic curve discrete

logarithm problem to construct a more standard discrete logarithm of the cryptosystem [20]. The problem of using smaller finite fields does not exist. In the supersingular curve, there is an effective algorithm that maps two points on the curve (on a finite field) to an element in the base field. In the supersingular curve, the typical representatives are the Weil pair and Tate pair. These two supersingular elliptic curve pair transformations can be used to construct a bilinear pair. The basic concept of bilinear pairing is as follows:

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be the additive group and multiplicative group of order q respectively and P is the generator of $\mathbb{G}_1$. Suppose that in the groups $\mathbb{G}_1$, $\mathbb{G}_2$, the discrete logarithm problem is difficult to solve. The bilinear mapping pair can be defined as e: $\mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ and meet the following characteristics:

(1) *Bilinear*. For all P, $Q \in \mathbb{G}_1$ and $a, b \in \mathbb{F}_p^*$, $e(aP, bQ) = e(abP, Q) = e(P, abQ) = e(P, Q)^{ab}$.

(2) *Nondegenerate*. There is one $P \in \mathbb{G}_1$, which satisfies $e(P, P) \neq 1$.

(3) *Computability*. For $P, Q \in \mathbb{G}_1$, there is an effective algorithm to calculate $e(P, Q)$.

A bilinear map e can be constructed using the modified Weil [21] or Tate pairings [22] on supersingular elliptic curve $\mathbb{G}_1$. A group $\mathbb{G}_1$ with such a map e is called a bilinear group, on which the CDHP is assumed hard, while the DDHP (decisional Diffie–Hellman problem) is easy [23]. Namely, given unknown $a, b, c \in \mathbb{F}_p^*$ and $P, aP, bP, cP \in \mathbb{G}_1$, it is known that there exists an efficient algorithm to determine whether $ab = c \bmod p$ by verifying $e(aP, bP) \stackrel{?}{=} e(P, cP)$ in polynomial time (DDHP), while there exist no efficient algorithms to compute $abP \in \mathbb{G}_1$ with non-negligible probability within polynomial time (CDHP).

2.3. Ring Signature, Aggregate Signature, and Monero Blockchain. Ring signature is a kind of digital signature scheme, which was first proposed by Kim [24]. Ring signature has only ring members, no manager and no cooperation between ring members. Ring signature refers to hiding the public key with private key among n public keys, which supports hiding transaction sender (address/public key) on blockchain. Suppose there are n users, and each user has a public key and its corresponding private key. Ring signature is a signature scheme, which can realize the signer's unconditional anonymity, which is mainly composed of the following algorithms:

(1) *Generate Gen*. It is a probabilistic polynomial-time (PPT) algorithm. The input is security parameter K, and the output is public key and private key. It is assumed that Gen generates a public key and private key for each user, and the public and private keys of different users may come from different public key systems, such as RSA and DL.

(2) *Sign*. It is a PPT algorithm. After inputting the message m and the public key of n ring members $L = y_1, y_2, \ldots, y_n$ and the private key information of one of the members, a signature R is generated for the message m. The parameters are in a ring shape according to certain rules.

(3) *Verify*. It is a deterministic algorithm. After inputting (m, R), if R is the ring signature of M, then it will output "true"; otherwise, it will be "false."

Aggregate signature is a variant signature scheme used to aggregate any number of signatures into a single signature [25]. It can merge the public key and signature of each participant in a multisignature transaction into one public key and signature. The whole merging process is invisible, and the information before merging cannot be deduced from the combined public key and signature; only one verification is needed during verification. At present, the Schnorr signature algorithm is usually used to implement signature aggregation [26].

Monero is a cryptocurrency for the connected world [27]. It is fast, private, and secure. Monero has the following three characteristics:

(1) *Anonymity*. Monero achieves anonymity by using ring confidential transactions (a combination of ring signature and anonymous transactions) and anonymous addresses. In addition, Kovri is used to confuse point-to-point communication. Due to the use of ring signature, at least six bait coins are added to each transaction, and each currency seems to be the actual amount spent in the transaction, making the actual source and target almost impossible to trace.

(2) *Scalability*. Due to the use of ring signature, each transaction is attached with additional data, which greatly increases the size of the blockchain. At the time of writing, the Monero blockchain is about 48 GB in size and will continue to grow with wider adoption, placing a burden on scalability. We estimate that the average transaction size in the Monero network is about 14 KB, almost 25 times the size of bitcoin. Simply put, when Monero reaches bitcoin's current volume of transactions, its blockchain will be about 5 TB—almost unbearable for ordinary PCs, let alone on small devices. It is worth noting that the Monero team is currently implementing a bulletproof protocol that can increase scalability by up to 80 (still about five times that of bitcoin).

(3) *Auditing*. Monero provides the viewkey function to allow a third party to audit users' transactions. However, it only allows you to view the input transactions, not the output transactions, making it less friendly to auditors. In addition, there seems to be no way to prove that the incoming transaction list is complete.

3. System Model and Security Model

In this section, we describe the system model and security model for the BBARHS scheme. It contains the participating entities and the security requirements of the scheme.

3.1. System Model. In the BBARHS scheme for autonomous taxi network, there are four participating entities, namely, RHS, LAV (local autonomous vehicles), rider, and the blockchain. The detailed description is as follows:

(1) *RHS*. It is a ride-hiding service platform. It authorizes the legal riders and receives payment from riders. Considering management and performance bottlenecks, RHS does not provide direct services and LAVs added to autonomous taxi network.

(2) *LAV*. It is a local AVs provider. LAV provides ride-hiding service directly to local riders. When the ride-hiding service is completed, it sends the certificate to RHS and the receipt to the rider, respectively. Receipt and certificate are used to prove its service.

(3) *Rider*. Before requesting service, the rider needs to obtain RHS certification. Rider accepts LAV's service and makes the payment through the blockchain.

(4) *Blockchain*. Rider and RHS have digital currency on the blockchain. Riders pay for service through the blockchain and RHS receives the digital coins sent by the rider.

3.2. Security Model. In the BARHS for autonomous taxi network, we consider four privacy issues. The privacy issues include the identity of the rider, payer address of the rider, payee address of RHS, and the unlinkability between payer address and payee address. In order to protect the above privacy, our BBARHS scheme must satisfy the following goals:

(1) *Mutual Identification among Rider, RHS, and LAV*. When the rider needs ride-hailing service, he must get the authorization from RHS. By verifying the validity of the authorization from RHS, LAV chooses whether to provide service. RHS and LAV interact to get the detailed information of rider.

(2) *Anonymity for Rider*. In the process of receiving LAV's service, LAV cannot identify the identity of rider. In the process of authorization and payment, RHS cannot identify the identity of rider.

(3) *Anonymity for RHS*. Although RHS receives payment from the rider, RHS cannot identify rider's address on the blockchain.

(4) Unlinkability between payer address (for the rider) and payee address (for RHS) on blockchain.

This study uses many notations. These notations and the corresponding descriptions are listed in Table 1.

According to the above security requirements, we give the formal security definitions as follows:

Definition 1. Unforgeability: RHS's authentication protocol satisfies the unforgeability if the probability that A wins the following game is negligible where A is PPT (probabilistic polynomial time) adversary.

(1) *Setup*. The system parameters are created and RHS's private/public key pair is generated. At the same time, rider's private/public key pair is also generated. For RHS, its private/public key pair is generated in PKI (public key infrastructure). For the rider, its private/public key pairs are generated by the rider itself where there is no trusted third party. System parameters, RHS's public key, and rider's public key are sent to A. We denote the public parameters as params.

(2) *Interaction between A and the Challenger C*. In the interaction, A adaptively queries C and gets C's responses. The queries and responses are listed as follows:

(a) *Hash Query*. A sends the hash queries to C. C creates the hash function value and sends it to A (random Oracle model). C accesses the hash function and responds A with the real hash value (standard model).

(b) *Authentication Query*. A makes the authentication query on the different public key with the corresponding message, which are denoted as Cont_i. C creates the authentication $\sigma_{\widehat{i}}$ and sends it to A. In the process, we denote the query set as $\left\{\mathrm{Cont}_{\widehat{i}} | \widehat{i} \in \mathbb{I}\right\}$ and the response set as $\left\{\sigma_{\widehat{i}} | \widehat{i} \in \mathbb{I}\right\}$.

(3) *Forgery*. A can forge a valid authentication on a new public key with the corresponding message, which is denote as Cont. The forged public key and metadata are different from the queried public keys with the corresponding messages, that is, $\mathrm{Cont} \notin \left\{\mathrm{Cont}_{\widehat{i}} | \widehat{i} \in \mathbb{I}\right\}$.

We say that A wins the above game between A and C if $\Pr[\mathrm{Verify}(\mathrm{Cont}, \mathrm{params}) = \text{“success”} | \mathrm{Cont} \notin \left\{\mathrm{Cont}_{\widehat{i}} | \widehat{i} \in \mathbb{I}\right\}] \geq (1/p(\mathrm{k}))$ is a polynomial of the security parameter k. In other words, we say that A wins if A's success probability is non-negligible.

Note 1. The above definition gives the formal definition of unforgeability for RHS authentication. For LAV authentication and ring signature for transaction, the formal definitions of their unforgeability are similar to the above definition. Due to the page limits, we omit the corresponding definitions.

Definition 2. Anonymity for the Rider: in the BBARHS scheme, RHS is unconditionally anonymous. In other words,

Table 1: Notations and descriptions.

Notations	Descriptions
E	Elliptic curve over $\mathbb{F}_q$ for Monero
$\mathbb{F}_q, \mathbb{F}_p$	The finite field
$\mathbb{F}_q^*, \mathbb{F}_p^*$	The corresponding multiplicative groups of $\mathbb{F}_q, \mathbb{F}_p$
G	Base point of E
$\widehat{l}$	Prime order of G
H_1, H_2	Cryptographic hash functions
(a, b)	RHS's random private key
(A, B)	RHS's random public key
(x, y)	Rider's private key on the blockchain
(X, Y)	Rider's public key, which is also the rider's address on the blockchain
$(\mathbb{G}_1, \mathbb{G}_2)$	Bilinear group pair
p	The prime order of $\mathbb{G}_1$ and $\mathbb{G}_2$
e	Bilinear pairing
P A	Generator of $\mathbb{G}_1$
(z, Z)	RHS's private key/public key pair over $\mathbb{G}_1$ in PKI
(l_j, L_j)	LAV_j's private key/public key pair over $\mathbb{G}_1$ in PKI
H	Full-domain cryptographic hash function
$\mathbb{I}$	The set of rider's index
$\mathbb{J}$	The set of LAV's index
$m_{i,j}$	The message that describes the service between rider_i and LAV_j
Tab	The table to record which riders are valid or invalid
$(\text{Cont}_i, \sigma_i)$	Authorization for Rider_i from RHS
$(m_{i,j}, A_i, B_i\widehat{\sigma}_{i,j})$	The receipt for Rider_i from LAV_j and the certificate for RHS from LAV_j
$\widehat{P}_i$	One-time public key for RHS, which is also RHS's address on the blockchain
bal_i	The payment balance from rider_i
bal_c	The remaining change for rider_i
P_i	The public key of the user U_i, which is also U_i's address on the blockchain
$\lvert S\rvert$	The cardinality of the set S

the adversary cannot identify the rider's real identity even if the adversary's computing power is infinite.

Definition 3. Anonymity for RHS: when the rider has n_1 output addresses, if all the output addresses do not belong to the adversary, then the probability that RHS's address can be identified is not more than $1/n_1$.

Definition 4. Untraceability: for each incoming transaction, all possible senders are equiprobable.

Definition 5. Unlinkability: for any two outgoing transactions, it is impossible to prove they were sent to the same person.

4. An Efficient BBARHS Scheme

According to the system model and security model proposed in the previous section, we design a BBARHS scheme for autonomous taxi network. To satisfy the security model and transparent management, the proposed scheme takes use of some cryptographic techniques, which include PKI, digital signature, elliptic curve cryptography, and Monero. The proposed scheme consists of seven procedures: (i) initialization, (ii) contract-based authorization, (iii) anonymous service provision, (iv) pay for services, (v) verification and gain, (vi) solving the dispute, and (vii) rider revocation. The detailed procedures are given below.

In order to show the intuition of the scheme, the structure of our scheme is shown in Figure 2. There are four entities, that is, blockchain, RHS, LAV, and rider. We give the description of their interactions. (1) Riders and RHS generate the private key and public key respectively. The public key is the address on blockchain. RHS and LAVs also generate private key and public key pairs in PKI. (2) By making use of rider's public key, the rider registers himself at RHS. (3) RHS has a table to record which riders are valid or invalid. RHS authorizes LAV to provide service for valid riders. (4) LAV provides service for the rider and sends the receipt to the rider. At the same time, LAV sends certificate to RHS. (5) Rider pays for the service through blockchain.

4.1. Initialization. In our proposed system, we use two types of system parameters. One type of parameter is used for blockchain. Other types of parameters are used for registration, receipt, certificate, authorization, and revocation. The detailed generated procedures are given below. Figure 3 shows the process of contract-base authorization, anonymous service and receipt, pay for the service, and verification and gain.

4.2. Contract-Based Authorization. In order to get the service from LAV, the ith rider, that is, rider_i, needs RHS's agreement. First, RHS needs to get rider_i's status information. Based on this information, the agreed contract Cont_i is created. Cont_i contains rider's status information, pay for

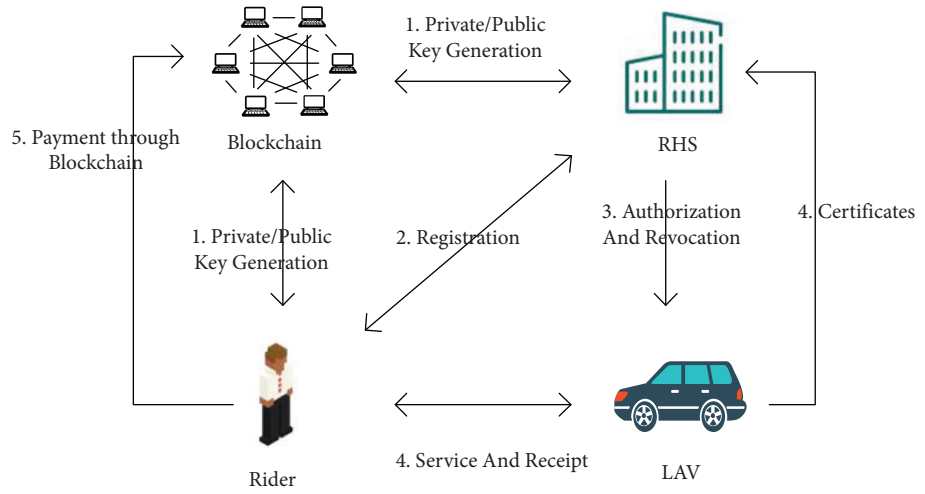

Figure 2: Architecture of BBARHS scheme.

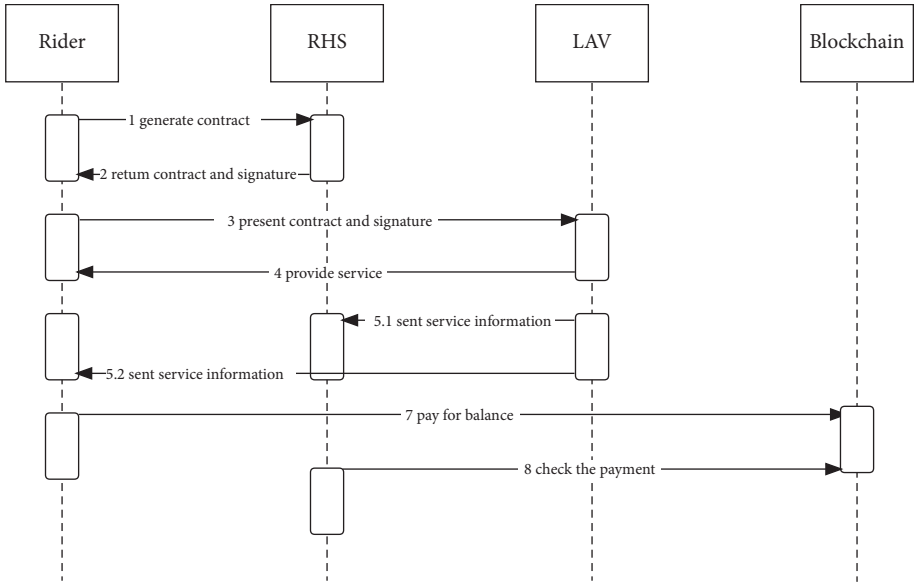

Figure 3: Sequence diagrams of BBARHS scheme.

the service, service standards, etc. In order to protect rider's privacy, Cont_i does not contains rider's identity information. X_i is contained into Cont_i to get the authorization from RHS. At the same time, RHS records a table, which lists the valid rider. The authorization procedures are as follows:

(1) RHS generates the signature σ_i for Cont_i below: $\sigma_i = zH(\text{Cont}_i)$.

(2) RHS adds the address (X, Y) into the table Tab.

(3) RHS sends $(\text{Cont}_i, \sigma_i)$ to rider.

(4) RHS sends the updated table Tab to all the LAVs.

4.3. Anonymous Service and Receipt. When rider_i accesses the anonymous taxi network and requests the service from RHS, it presents $(\text{Cont}_i, \sigma_i)$ to the LAV_j in the local area, where $j \in \mathbb{J}$. LAV_j's private/public key pair is (l_j, L_j), where $L_j = l_j P$. $(\text{Cont}_i, \sigma_i)$ can be verified as follows:

(1) At some moment, LAV_j receives a lot of pairs $(\text{Cont}_i, \sigma_i)$, where $i \in \mathbb{I}$.

(2) For every $i \in \mathbb{I}$, LAV_j extracts rider_i's public keys (X, Y) from Cont_i. If (X, Y) belongs to Tab, then LAV_j provide service for rider_i; otherwise, LAV_j rejects rider_i's request.

(3) LAV_j picks the random numbers $\alpha_i \in \mathbb{F}_p$, $i \in \mathbb{I}$ and verifies whether the following formula holds:

$$e\left(\sum_{i\in\mathbb{I}} \alpha_i \sigma_i, P\right) = e\left(\sum_{i\in\mathbb{I}} \alpha_i H(\text{Cont}_i), Z\right). \quad (1)$$

If the formula does not hold, then LAV_j finds out the invalid pairs and rejects them, and then go to the following procedure.

(4) When rider_i receives service from LAV_j, LAV_j generates the receipt as follows:

(a) Denote LAV_j's service information as the message $m_{i,j}$. LAV_j computes $\overline{\sigma}_{i,j} = l_j H(m_{i,j}, X, Y)$.

(b) LAV_j sends $(m_{i,j}, X, Y, \overline{\sigma}_{i,j})$ to RHS and rider_i, respectively.

For rider_i, $(m_{i,j}, X, Y, \overline{\sigma}_{i,j})$ is the receipt for its service. For RHS, $(m_{i,j}, X, Y, \overline{\sigma}_{i,j})$ is the certificate of the service for rider_i.

4.4. Pay for the Service. Suppose that there are many LAVs. We denote them as $\mathrm{LAV}_j, j \in \mathbb{J}$. For LAV_j, the corresponding private/public key pair is (l_j, L_j), where $L_j = l_j P$. RHS receives the certificate $(m_{i,j}, A, B, \overline{\sigma}_{i,j})$ from LAG_j, where $i \in \mathbb{I}, j \in \mathbb{J}$. RHS picks the random numbers $\beta_{i,j} \in \mathbb{F}_p, i \in \mathbb{I}, j \in \mathbb{J}$ and verifies them by checking whether the following formula holds:

$$e\left(\sum_{j\in\mathbb{J},i\in\mathbb{I}} \beta_{i,j}\overline{\sigma}_{i,j}, P\right) = \prod_{j\in\mathbb{J},i\in\mathbb{I}} e\left(\beta_{i,j}H\left(m_{i,j}, A, B\right), L_j\right). \quad (2)$$

By checking the correctness of the above formula, RHS can determine whether the service is complete.

In order to pay for the service, rider_i performs the following procedures:

(1) For the rider_i where $i \in \mathbb{I}$, rider_i unpacks the received messages and gets RHS's address (A, B). Rider_i generates a random number $r \in \mathbb{F}_l^*$, and then it gets a one-time public key $\widehat{P}_i = H_1(rA)G + B$ for RHS.

(2) For the messages $m_{i,j}$, where $i \in \mathbb{I}, j \in \mathbb{J}$, rider_i can unpack it and get the balance $\mathrm{bal}_{i,j}$ to be paid. Suppose that rider_i has the balance bal on the blockchain. Concretely, for RHS whose one-time public key (account address) is $\widehat{P}_i$, the output corresponds to the rewarding balance bal_i. Besides them, the additional output corresponds to the change $\mathrm{bal}_c = \mathrm{bal} - \mathrm{bal}_i$. In order to simply the symbols, we denote the outputs and some metadata as the message m. For example, m contains R and $(\widehat{P}_i, \mathrm{bal}_i)$ for RHS, where $R = rG$.

(3) Rider_i calculates $A = H_2(\mathrm{Sign}_t(m))$ where $\mathrm{Sign}_t(m)$ is the signature on the message m by making use of rider_i's private key t.

(4) Rider_i selects a random subset $S\imath$ of the other users' public key P_i, $S\imath$ has the cardinality n, and his own private/public key pair is (x_s, P_s), where $0 \le s \le n$. It also computes the image $I = x_s H_2(Ps)$. It picks the random numbers $q_i | i = 0, 1, \ldots, n$ and $w_i | i = 0, 1, \ldots, n, i \ne s$ from $\mathbb{F}_l^*$. Then, it computes the following points:

$$L_i = \begin{cases} q_i G & \text{if } i = s \\ q_i G + w_i P_i & \text{if } i \ne s, \end{cases}$$
$$R_i = \begin{cases} q_i H_2(P_i) & \text{if } i = s \\ q_i H_2(P_i) + w_i I & \text{if } i \ne s. \end{cases} \quad (3)$$

Then, rider_i computes

$$c = H_1(m, A, L_0, \ldots, L_n, R_0, \ldots, R_n). \quad (4)$$

The following values

$$c_i = \begin{cases} w_i & \text{if } i \ne s \\ c - \sum_{i\ne s} c_i & \text{if } i = s, \end{cases}$$
$$r_i = \begin{cases} q_i & \text{if } i \ne s. \\ q_s - c_s x_s & \text{if } i = s. \end{cases} \quad (5)$$

The resulting signature is as follows:

$$\sigma = (I, A, c_0, \ldots, c_n, r_0, \ldots, r_n). \quad (6)$$

When rider_i finished the above procedures, it sends the balance bal_i to $\widehat{P}_i$ and bal_c to $\widehat{P}_c$ with the signature σ, where $\widehat{P}_c$ is selected by rider_i. $\widehat{P}_c$ is used to store the change bal_c.

Note 2. To avoid double-spending, the private key can be used only one time on the Monero. Thus, the change bal_c must be moved to new address $\widehat{P}_c$, which is chosen by rider_i on the Monero.

4.5. Verification and Gain. When RHS receives the signature σ and the message m, RHS performs the following procedures to check the validity of the signature σ:

(1) RHS computes $F2$ in Table 2.

(2) RHS checks whether $F3$ in Table 2 holds. If the formula does not hold, the signature is rejected.

(3) RHS checks whether I has been used in past signatures. If it appeared in the past signatures, then the signature is rejected; otherwise, RHS accepts σ.

In other words, the coin bal_i is moved to $\widehat{P}_i$ and bal_c is moved to $\widehat{P}_c$ from the address P_s.

RHS checks rider's payment (m, σ). From m, RHS extracts R and $(\widehat{P}_i, \mathrm{bal}_i)$ where $i \in \mathbb{I}$. RHS computes $P_i' = H_1(aR)G + B_i$. If $P_i' \in \widehat{P}_i, i \in \mathbb{I}$, then there exists $\widehat{i} \in \mathbb{I}$, which satisfies $P_i' = P_{\widehat{i}}$. In order to gain the reward l_i, RHS computes $k_i = H_1(a_i R) + b_i$, which satisfies $P_{\widehat{i}} = k_i G$. Thus, RHS gains the reward bal_i. Since RHS knows the private key of the address $P_{\widehat{i}}$, RHS gains the reward bal_i.

Notes. In the initialization procedure, RHS can generate a lots of account address, which are used for receiving coins from different transactions.

4.6. Solve the Dispute. When RHS cannot find its bal_i, it sends its certification to rider. Rider checks RHS's certification, and if it is valid, then rider tells RHS the transaction

Table 2: Formulas in Section 4.

$F1$: $L_i' = r_i G + c_i P_i, R_i' = r_i H_2(P_i) + c_i I$
$F2$: $e(\sum_{j\in\mathbb{J},i\in\mathbb{I}} \beta_{i,j}\overline{\sigma}_{i,j}, P) = \prod_{j\in\mathbb{J},i\in\mathbb{I}} e(\beta_{i,j} H(m_{i,j}, A, B), L_j)$
$F3$: $\sum_{i=0}^{n} c_i = H_1(m, A, L_0', \ldots, L_n') \bmod \widehat{l}$

information with the ring signature. If RHS thinks rider is not the real signer, then rider performs the following procedure to prove he is the real signer:

(1) Rider shows RHS the preimage $\mathrm{Sign}_t(m)$ of the hash function H on the image A.

(2) RHS verifies whether $\mathrm{Sign}_t(m)$ is the preimage of A and whether $\mathrm{Sign}_t(m)$ is a valid signature signed by rider's public key T. If it is valid, then RHS admits that rider is the real signer; otherwise, RHS denies that rider is the real signer.

4.7. Rider Revocation. Rider revocation schemes from two cases are as follows:

(1) Case 1. When RHS wants to reject rider's service, RHS updates the table Tab. RHS adds the revocation information to the table Tab and sends it to LAVs.

(2) Case 2. When the rider would like to be revoked, the rider can inform the revocation information to RHS. RHS updates the table Tab and sends it to LAVs.

5. Security and Performance Analysis

The security and performance of our scheme are given in this section. According to security analysis, performance analysis, and the simulation result, our BBARS scheme is secure and practical.

5.1. Security Analysis

Theorem 1. *Authorization Correctness: if RHS and LAV are honest and follow the proposed BBARS scheme, then rider's authorization from RHS can pass LAV's verification.*

Proof. According to the generation procedures of rider_i's authorization, we get

(1) Correctness for rider_i's authorization:

$$\begin{aligned} e\left(\sum_{i\in\mathbb{I}} \alpha_i\sigma_i, P\right) &= \prod_{i\in\mathbb{I}} e(\sigma_i, P)^{\alpha_i}, \\ &= \prod_{i\in\mathbb{I}} e(zH(\mathrm{Cont}_i), P)^{\alpha_i}, \\ &= \prod_{i\in\mathbb{I}} e(\alpha_i H(\mathrm{Cont}_i), zP), \\ &= e\left(\sum_{i\in\mathbb{I}} \alpha_i H(\mathrm{Cont}_i), Z\right). \end{aligned} \tag{7}$$

(2) Correctness for LAV_j's certificates:

$$\begin{aligned} e\left(\sum_{j\in J, i\in I} \beta_{i,j}\overline{\sigma}_{i,j}, P\right) &= \prod_{j\in\mathbb{J},i\in\mathbb{I}} e\left(\beta_{i,j}\overline{\sigma}_{i,j}, P\right), \\ &= \prod_{j\in\mathbb{J},i\in\mathbb{I}} e\left(\beta_{i,j} l_j H\left(m_{i,j}, X_i, Y_i\right), P\right), \\ &= \prod_{j\in\mathbb{J},i\in\mathbb{I}} e\left(\beta_{i,j} H\left(m_{i,j}, X_i, Y_i\right), l_j P\right), \\ &= \prod_{j\in\mathbb{J},i\in\mathbb{I}} e\left(\beta_{i,j} H\left(m_{i,j}, X_i, Y_i\right), L_j\right). \end{aligned} \tag{8}$$

□

Theorem 2. *Verification Correctness: if the rider and RHS are honest and follow the proposed scheme, then rider's signature can pass RHS's verification.*

Proof. From the generation process of rider's signature, we get

(1) When $i \neq s$, we get

$$L_i' = r_i G + c_i P_i = q_i G + \omega_i P_i = L_i. \tag{9}$$

$$R_i' = r_i H_2(P_i) + c_i I = q_i H_2(P_i) + \omega_i I = R_i. \tag{10}$$

(2) When $i = s$, we get

$$L_s' = r_s G + c_s P_s = (q_s - c_s x_s)G + c_s P_s = q_s G = L_s. \tag{11}$$

$$\begin{aligned} R_s' &= r_s H_2(P_s) + c_s I, \\ &= (q_s - c_s x_s) H_2(P_s) + c_s I, \\ &= q_s H_2(P_s), \\ &= R_s. \end{aligned} \tag{12}$$

Based on equations (1) and (2), we get

$$\begin{aligned} \sum_{i=0}^{n} c_i = c &= H_1(m, A, L_0, L_1, \ldots, L_n, R_0, R_1, \ldots, R_n) \\ &= H_1(m, A, L_0', L_1', \ldots, L_n', R_0', R_1', \ldots, R_n') \bmod \widehat{l}. \end{aligned} \tag{13}$$

□

Theorem 3. *Unforgeability: In our BBARHS scheme, rider_i's authorization from RHS, the receipt and certificate from LAV, and the ring signature from rider satisfy the unforgeability.*

Proof. In our BBARHS scheme, we take use of BLS short signature scheme in PKI. BLS short signature scheme is provably secure, and the detailed proof process has been given in the reference. For the ring signature from rider, we take use of Saberhagen's transaction scheme. The detailed proof process is similar to Saberhagen's proof process, which can get in the reference. [28] Due to page limitations, we omit the proof process. □

Theorem 4. *Anonymity for Rider: in our BBARHS scheme, the rider is unconditionally anonymous. We prove this theorem from the following two parts:*

(1) In the phase of contract-based authorization, rider's contract $Cont_i$ does not include its identity information. The chosen public keys (X_i, Y_i) have nothing to do with $rider_i$'s identity. In the phase anonymous service and receipt, $rider_i$ submits $(Cont_i, \sigma_i)$ to LAV_j. The message does not include $rider_i$'s identity. Thus, our BBARHS scheme does not need $rider_i$'s identity. The scheme satisfies the anonymity for rider.

(2) Our phase pay for the service takes use of the ring signature to realize the anonymity of rider. From the final signature $\sigma = (I, c_0, \ldots, c_n, r_0, \ldots, c_n)$, we know that $I = x_s H_2(P_s)$. On the other hand, both $c_s = H_1(m, A, L_1, \ldots, L_n, R_1, \ldots, R_n) - \sum_{i \neq s} c_i$. Thus, if rider generates a transaction on behalf of a ring of n addresses, then rider's anonymity can be satisfied.

Theorem 5. *Anonymity for RHS: our BBARHS scheme satisfies anonymity for RHS.*

Proof. When there are n_1 output addresses and all the output addresses do not belong to the adversary, all the one-time public key $\widehat{P}_i = H_1(rA_i)G + B_i$ is random due to the property of the hash function H_1. Thus, all the n_1 output addresses are random for the adversary.

In the following part, we show why our BBARHS scheme can satisfy the unlinkability and solve the dispute:

(1) From the phase pay for the service, the outgoing transaction addresses are one-time public key $\widehat{P}_i = H_1(rA)G + B$ for RHS, where $r \in \mathbb{F}_{\widehat{q}}^*$. Thus, for any two outgoing transactions, it is impossible to prove they are sent to the same person.

(2) From the phase pay for the service, we know that $A = H_2(\text{sign}_t(m))$. Based on the security properties of hash function H_2, only rider know A's preimage $\text{sign}_t(m)$. Thus, by showing $\text{Sign}_t(m)$, rider can prove he is the real signer. If RHS still does not believe rider is the real signer because A's preimage may be come from others, then RHS verifies $\text{sign}_t(m)$ by making use of rider's public key T.

(3) If rider and RHS are honest and follow the above process, then they can resolve dispute. □

5.2. Performance Analysis. Our BBARHS scheme must be efficient in terms of computation cost and communication cost. In our BBARHS scheme, two different PKCs are used: PKCs in PKI and PKCs on the blockchain. For PKCs on the blockchain, on the elliptic curve E, the point addition cost is denoted as C_{Eadd} and the scalar multiplication cost is denoted as C_{Emul}. On the bilinear group pair $(\mathbb{G}_1, \mathbb{G}_2)$, the bilinear pairing cost is denoted as C_{Bpair}, the scalar multiplication cost is denoted as C_{Bmul}, and the point addition cost is denoted as C_{Badd}. Comparing to the above computation cost, the other computation cost is smaller. For the above denotations, E_{mul} is the abbreviation of scalar multiplication and E_{add} is the abbreviation of point addition on E. On the other hand, B_{pair} is the abbreviation of bilinear pairing, B_{mul} is the abbreviation of scalar multiplication, and B_{add} is the abbreviation of point addition on $(\mathbb{G}_1, \mathbb{G}_2)$.

5.3. Theoretical Performance Analysis. For the performance of our scheme, the theoretical analysis is given in Table 3. In Table 3, $*$ denotes the entity does not take part in the procedure. "Small" denotes the computation cost is less than the five operations C_{Emul}, C_{Eadd}, C_{Bpair}, C_{Bmul}, and C_{Badd}. n denotes the ring size of the ring signature. The procedure contract-based authorization is performed between the rider and RHS. Rider's computation cost is small and RHS's computation cost is $|\mathbb{I}|C_{\text{Bmul}}$, where $|\mathbb{I}|$ is the number of riders. The procedure anonymous service and receipt is performed between rider and LAV. Rider's computation cost is small and LAV's computation cost is $2C_{\text{Bpair}} + 3|\mathbb{I}|C_{\text{Bmul}} + 2(|\mathbb{I}| - 1)C_{\text{Badd}}$. The procedure pay for the service is performed by rider and RHS. Rider's computation cost is $(4N + 4n - 2)C_{\text{Emul}} + (N + 2n - 2)C_{\text{Eadd}}$, where N is the number of services to be paid. N denotes the number of transactions needed to pay. RHS's computation cost is $(N + 1)C_{\text{Bpair}} + 2NC_{\text{Bmul}} + (N - 1)C_{\text{Badd}}$. The procedure verification and gain is performed by RHS and blockchain. RHS's computation cost is $3NC_{\text{Emul}} + NC_{\text{Eadd}}$ and blockchain's computation cost is $4nC_{\text{Emul}} + 2nC_{\text{Eadd}}$.

According to the above two PKCs, we analyze our BBARHS scheme's communication cost, as listed in Table 4. The PKC on the blockchain is the elliptic curve over the finite field $\mathbb{F}_q$, where $|q| = 256$ bits. In the bilinear group $(\mathbb{G}_1, \mathbb{G}_2)$, $\mathbb{G}_1$ is the supersingular elliptic curve over the finite field $\mathbb{F}_{\widehat{q}}$, where $|\widehat{q}| = 512$ bits. In the procedure contract-based optimization, rider sends $|\text{Cont}_i|$ to RHS. RHS sends $C_{\text{Rider}_i} = |\text{Cont}_i| + |\sigma_i| = 1024 + |\text{Cont}_i|$ bits to Rider_i and sends $C_{\text{LAV}} = |\text{Tab}|$ bits to LAV. σ_i's size $|\sigma_i|$ is a constant 1024. $|\text{Cont}_i|$ is the size of the contract. At the same time, the communication cost of LAV only comes from the size $|\text{Tab}|$. Thus, C_{LAV} has the linear relation with $|\mathbb{I}|$. In the procedure anonymous service and receipt, LAV_j sends $(m_{i,j}, X_i, Y_i, \overline{\sigma}_{i,j})$ to RHS and rider_i, respectively. The corresponding communication cost is $C_{\text{LAV}_j} = 2(|m_{i,j}| + |X_i| + |Y_i| + |\sigma_{i,j}^-|) = 2|m_{i,j}| + 6144$ bits. For the communication cost C_{LAV_j}, $|X_i|$, $|Y_i|$, and $\overline{\sigma}_{i,j}$ are the same constant, which are 1024 bits. On the other hand, $|m_{i,j}|$ has the linear relation with C_{LAV_j}. In the procedure pay for the service, the final signature size is $C_{\text{rider}} = |\sigma| = |I| + |A| + \sum_{i=0}^{n} |c_i| + \sum_{i=0}^{n} |r_i| = 1024 + 506n$ bits for per service. For the communication cost C_{rider}, the sizes of I, A, c_i, and r_i are different constants. They are 512 bits, 512 bits, 253 bits, and 253 bits, respectively. At last, C_{Payment} has the linear relation with n, which is the number of the ring users.

5.4. Implementation. In order to demonstrate our BBARHS scheme's effectiveness, we implemented it by making use of the GMP (GMP-5.0.5), Miracl, and PBC (pbc-0.5.13) libraries. In our simulation, both RHS and LAV ran on a computer, which has the following features:

Table 3: Computation cost of the different entities.

	Rider	LAV	RHS	Blockchain
Contract-based authorization	Small	*	$\lvert\mathbb{I}\rvert C_{\mathrm{Bmul}}$	*
Anonymous service and receipt	Small	$2C_{\mathrm{Bpair}} + 3\lvert\mathbb{I}\rvert C_{\mathrm{Bmul}} + 2(\lvert\mathbb{I}\rvert - 1)C_{\mathrm{Badd}}$	*	*
Pay for the service	$(4N + 4n - 2)C_{\mathrm{Emul}} + (N + 2n - 2)C_{\mathrm{Eadd}}$	*	$(N + 1)C_{\mathrm{Bpair}} + 2NC_{\mathrm{Bmul}} + (N - 1)C_{\mathrm{Badd}}$	*
Verification	*	*	$3NC_{\mathrm{Emul}} + NC_{\mathrm{Eadd}}$	$4nC_{\mathrm{Emul}} + 2nC_{\mathrm{Eadd}}$

Table 4: Communication cost of the different entities (bits).

Procedure	Rider	LAV	RHS
Contract-based authorization	$2\|\text{Cont}_i\| + 1024$	$\|\text{Tab}\|$	$2\|\text{Cont}_i\| + \|\text{Tab}\| + 1024$
Anonymous service and receipt	$\|m_{i,j}\| + 3072$	$2\|m_{i,j}\| + 6144$	$\|m_{i,j}\| + 3072$
Pay for the service	$1024 + 506n$	*	*

*denotes the entity that does not take part in the procedure.

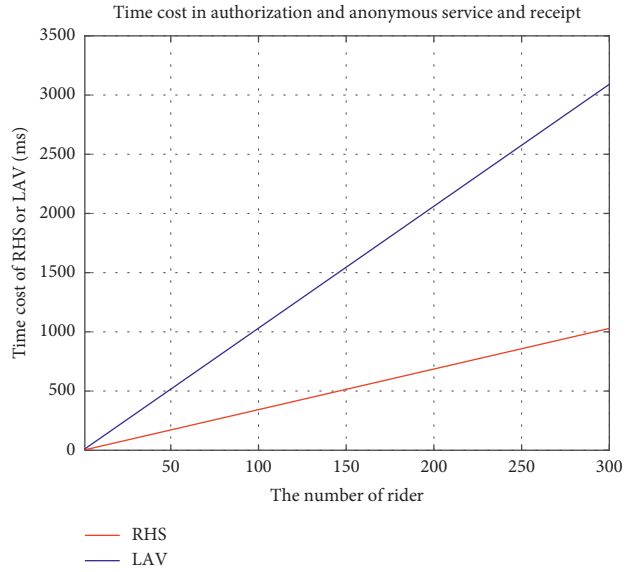

Figure 4: RHS's time cost in contract-based authorization and LAV's time cost in anonymous service and receipt.

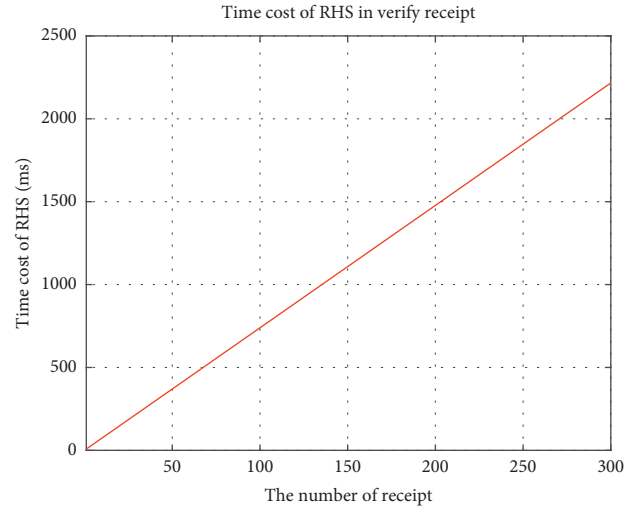

Figure 5: RHS's time cost in verifying receipt.

(i) CPU: Intel Core i5-8500 @ 3.0 GHz

(ii) Physical memory: 8 Gb

(iii) OS: Ubuntu 18.04

Rider ran on a laptop, which has the following features:

(i) CPU: Intel Core i5-8500 @ 3.0 GHz

(ii) Physical memory: 8 Gb

(iii) OS: Ubuntu 18.04

Besides the above simulation environment, we take use of the Monero blockchain. For the PKCs in PKI, we take use of the bilinear group $(\mathbb{G}_1, \mathbb{G}_2)$, where $\mathbb{G}_1$ is defined on the finite field $F_{\widehat{q}}$ with $|\widehat{q}| = 512$ bits. At the same time, $\mathbb{G}_1$ is a supersingular elliptic curve with 160 bits group order. Figure 4 depicts the time cost of RHS and LAV in the procedures contract-based authorization and anonymous service and receipt, respectively. In the figure, X-axis represents the number of riders. The Y-axis represents RHS and LAV's computing time in ms (i.e., milliseconds). Because RHS authorizes rider independently, CAG's time cost increases along with the increasing of rider number. On the other hand, LAV's time cost increases fastly along with the increasing of rider number, which we can get the corroboration from Table 3. Figure 5 depicts the time cost of RHS in the procedure pay for the service. The Y-axis represents RHS's computing time in ms (i.e., milliseconds). RHS's time cost increases fastly along with the increasing of receipt number. TPS refers to the number of orders processed per second. In the procedure contract-based authorization, the average consumption time of RHS is about 10 ms and TPS is 100. RHS's time cost in verifying receipt is about 8 ms and TPS is 125. LAG's average consumption time in anonymous service and receipt is about 3 ms and TPS is 333. According to Uber's published travel data for the third quarter of 2021, the number of orders processed per second was 206. The impact of performance is limited. The payment from the rider in the procedure pay for the service and the verification in the procedure verification are implemented by making use of Monero blockchain, which is effective and secure since 2014. Due to the page limits, we omit the corresponding simulation.

6. Conclusion

In this study, we studied the privacy protection and anonymous ride-hailing service for autonomous taxi networks. By taking use of blockchain, we present the security and privacy threats and first proposed blockchain-based anonymous ride-hailing scheme for autonomous taxi network. We give the formal system model and security model. Based on the aggregated signature, ring signature, and Monero, we design the first BBARHS scheme. The analysis and implementation show that our BBARHS scheme is provably secure and practical.

In the future, we will further study the system model and security model for different application scenarios, for

example, how to solve the privacy threats in multiperson carpooling.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (nos: U19A2086 and SKLGP2019Z014).

References

[1] A. Peters, "It could be 10 times cheaper to take electric robotaxis than to own a car by 2030," *Ver\xF6ffentlichung*, 2017.

[2] H. Lipson and M. KurmanMIT Press, Cambridge, MA, USA, 2016.

[3] H. Kersten, T. Möller, A. Padhi, and A. Tschiesner, *How Mobility Players Can Compete as the Automotive Revolution Accelerates*, McKinsey&Company, GA, USA, 2017.

[4] S. Solano, Á. Segura, G. León, J. M. Gutiérrez, and T. Burnouf, "Low ph formulation of whole igg antivenom: impact on quality, safety, neutralizing potency and viral inactivation," *Biologicals*, vol. 40, no. 2, pp. 129–133, 2012.

[5] G. Avoine, L. Calderoni, J. Delvaux, D. Maio, and P. Palmieri, "Passengers information in public transport and privacy: Can anonymous tickets prevent tracking?" *International Journal of Information Management*, vol. 34, no. 5, pp. 682–688, 2014.

[6] M. Swan O'Reilly Media, Inc, 2015.

[7] F. ZHao, M. Liu, K. Wang, and H. Zhang, "Color image encryption via Hénon-zigzag map and chaotic restricted Boltzmann machine over Blockchain," *Optics & Laser Technology*, vol. 135, Article ID 106610, 2021.

[8] M. R. Nosouhi, S. Yu, K. Sood et al., "Ucoin: An Efficient Privacy Preserving Scheme for Cryptocurrencies," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, 2021.

[9] M. Liu, X. Jiang, F. Zhao, X. Feng, and R. Wang, "A fast adaptive blockchain consensus algorithm via WLAN mesh network," *Journal of Internet Technology*, vol. 21, pp. 523–533, 2020.

[10] S. Underwood, *Blockchain Beyond Bitcoin*, Communications of the ACM, 2016.

[11] N. Zhang, S. Zhong, and L. Tian, "Using blockchain to protect personal privacy in the scenario of online taxi-hailing," *International Journal of Computers, Communications & Control*, vol. 12, no. 6, 886 pages, 2017.

[12] D. Bissell, T. Birtchnell, A. Elliott, and E. L. Hsu, "Autonomous automobilities: The social impacts of driverless vehicles," *Current Sociology*, vol. 68, no. 1, pp. 116–134, 2020.

[13] H. Yu, J. Shu, X. Jia, H. Zhang, and X. Yu, "Lpride: Lightweight and privacy-preserving ride matching over road networks in online ride hailing systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 10418–10428, 2019.

[14] Y. Khazbak, J. Fan, S. Zhu, and G. Cao, "Preserving location privacy in ride-hailing service," in *Proceedings of the IEEE Conference on Communications and Network Security (CNS)*, pp. 1–9, IEEE, Beijing, China, August 2018.

[15] A. Pham, I. Dacosta, B. Jacot-Guillarmod et al., "PrivateRide: a privacy-enhanced ride-hailing service," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 2, pp. 38–56, 2017.

[16] A. Pham, I. Dacosta, G. Endignoux, R T P Juan, K. . Huguenin, and J.-P. Hubaux, "Oride: a privacy-preserving yet accountable ride-hailing service," in *26th{USENIX}Security Symposium*vol. 17, pp. 1235–1252, USENIX}Security, 2017.

[17] Y. Wang, *Public Key Cryptography Standards: PKCS*, 2012.

[18] J. S. Vaeth and C. S. Walton, "Virtual certificate authority," US Patent 6,035,402, 2000.

[19] M. Nofer, P. Gomber, O. Hinz, and D. Schiereck, "Blockchain," *Business & Information Systems Engineering*, vol. 59, no. 3, pp. 183–187, 2017.

[20] A. Menezes, T. Okamoto, and S. A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE Transactions on Information Theory*, vol. 39, no. 5, pp. 1639–1646, 1993.

[21] M. Franklin, "Identity-based encryption from the weil pairing," in *Proceedings of the Annual International Cryptology Conference*, pp. 213–229, Springer, Berlin, Heidelberg, August 2001.

[22] A. Miyaji, M. Nakabayashi, and S. Takano, "New explicit conditions of elliptic curve traces for fr-reduction," *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, vol. 84, no. 5, pp. 1234–1243, 2001.

[23] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," *Journal of Cryptology*, vol. 17, no. 4, pp. 297–319, 2004.

[24] K. Kim, "Id-based blind signature and ring signature from pairings," in *Proceeings of the International Conference On the Theory And Application Of Cryptology And Information Security*, pp. 533–547, Springer, Berlin, Heidelberg, November 2002.

[25] D. Boneh, G. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proceedings of the International Conference On the Theory And Applications Of Cryptographic Techniques*, pp. 416–432, Springer, Berlin, Heidelberg, May 2003.

[26] G. Maxwell, A. Poelstra, Y. Seurin, and P. Wuille, "Simple schnorr multi-signatures with applications to bitcoin," *Designs, Codes and Cryptography*, vol. 87, no. 9, pp. 2139–2164, 2019.

[27] M. Möser, K. Soska, E. Heilman et al., "An empirical analysis of traceability in the monero blockchain," *Proceedings on Privacy Enhancing Technologies*, vol. 3, pp. 143–163, 2018.

[28] N. Van Saberhagen, "Cryptonote v 2.0," *Semantic Scholar*, vol. 17, no. 10, 2013.