# Secure Electronic Ticketing System based on Consortium Blockchain

**Xuelian Li[1], Jie Niu[1*], Juntao Gao[2] and Yue Han[1]**
[1] School of Mathematics and Statistics, Xidian University
Xi'an, Shaanxi, 710126, China
[e-mail: xuelian202@163.com, Niu_Jie1754@163.com, Han_Yue0526@163.com]
[2] School of Telecommunications engineering, Xidian University
Xi'an, Shaanxi, 710126, China
[e-mail: jtgao@mail.xidian.edu.cn]
*Corresponding author: Jie Niu

*Received December 10, 2018; revised February 2, 2019; accepted April 17, 2019; published October 31, 2019*

## Abstract

In electronic ticketing system, the malicious behavior of scalpers damages the customer's interest and disturbs the normal order of market. In order to solve the problem of scalpers, we took two steps. Firstly, we established the electronic ticketing system based on the consortium blockchain (CB-ETS). By establishing CB-ETS, we can make the ticketing market develop better in a controlled environment and be managed by the members in the consortium blockchain. Secondly, we put forward a kind of taxation mechanism for suppressing scalpers based on CB-ETS. Together with the regulatory mechanism, our scheme can effectively reduce the scalpers' profits and further inhibit scalpers. Through the above two steps, the scheme can effectively resist the malicious behavior of scalpers. Among them, in the process of transferring tickets, we optimized the transfer mechanism to achieve a win-win situation. Finally, we analyzed the security and efficiency of our scheme. Our scheme realizes the anonymity through the mixed currency protocol based on ring signature and guarantees the unforgeability of tickets by multi-signature in the process of modifying the invalidity of tickets. It also could resist to Dos attacks and Double-Spending attacks. The efficiency analysis shows that our scheme is significantly superior to relevant works.

***Keywords:*** Consortium blockchain; Electronic ticketing system; Scalpers; Taxation mechanism; Regulatory mechanisms

This work was supported in part by the National Key Research and Development Program of China (No, 2016YFB0800601) and the Natural Science Foundation of China (No. 61303217,61502372). We would like to express our gratitude to Prof. Kyungbaek Kim and reviewers who checked our manuscript.

**http://doi.org/10.3837/tiis.2019.10.022** **ISSN : 1976-7277**

## 1. Introduction

We know that 5G is poised to deliver on the promises of our connected world. CommScope anticipates that one goal of 5G will be to deliver 1,000 times more bandwidth than 4G in any given area, and that the location density of 5G cell sites will be five times that of 4G. It means 10 Gbps throughput, denser networks, super-low latency and unlimited potential [1-3]. Moreover, e-commerce is one of the best ways to connect and develop with the Internet, such as the Electronic Ticketing System (ETS). Therefore, 5G's deployment will provide ETS with more communication possibilities. The establishment will greatly reduced the time cost and required cost of customer reservation in ETS. It is a new growth point of economy in the 21st century. It breaks through the time and space restrictions through the Internet and realizes the functions of convenient and fast reservation booking and ticket management [4-5].

ETS has been closely integrated with daily life, for our life has brought great convenience. However, in order to take full advantage of the electronic ticketing system, we must address the challenges of the electronic ticketing system. There are two main challenges: the first is the challenge posed by the paperless characteristic of ETS, including how to achieve the anonymity and unforgeability of tickets and how to resist attacks such as DoS attack and Double-Spending attack. Since the ETS binds users' identity information to tickets to create unique tickets, we need to consider users' anonymity. At the same time, the ticket must satisfy the unforgeability. The most basic rights of the user can be guaranteed only if the unforgeability is satisfied. We also need to consider how to resist DoS attack and Double-Spending attack. If malicious merchants sell invalid tickets, it will harm the interest of users. In [13], the author added an attribute (serial number) to the electronic tickets in order to avoid the ticket being sold twice by the ETS. However, the authors used group signature, which makes the scheme inefficient.

Another challenge concerns the malicious behavior of scalpers. They take advantage of the convenience of ETS to make huge profits by transferring tickets at high prices. This kind of behavior breaks down the fairness and equality between consumers and merchants in the transaction and greatly damages the interests of consumers. It affects consumers' happiness index and market price, leading to market volatility. To solve the problem, the author hoped to ask intervention of law to protect consumer rights in South Korea[14]. As a result, appropriate mechanisms are expected to deal with these challenges, namely to meet basic security features in electronic ticketing systems and to be able to resist certain malicious attacks.

In recent years, the blockchain technology with decentralized characteristic, security and trust was introduced in ETS. The decentralized and trusted characteristic of blockchain ensures that there is no need to introduce a third party into ETS. The system can directly realize the peer-to-peer online communication between users and merchants without considering whether the two parties are trustworthy. The security characteristic address the challenges associated with ETS, because blockchain can provide characteristics such as inalterable data and public verification. The blockchain database can be managed autonomously through the use of point-to-point networks and distributed timestamp servers. Each block contains a timestamp linked to the previous block which makes the data cannot be untamable. Once recorded, the data in a block will be irreversible. Blockchain is designed as a safeguard, such as a highly fault-tolerant distributed computing system. Blockchain makes hybrid consistency possible. This makes the blockchain suitable for recording events, medical records and other activities that need to include data. What's more, the blockchain is very

helpful to enhance the efficiency in the industrial chain. This increases the overall efficiency of ETS. Therefore, it is necessary to establish an ETS scheme based on blockchain.

There are some reasearches for ETS on blockchain [15-18]. In [15], the author studied the ETS in a specific scenario, that is, they introduced the blockchain system to prevent “ticket theft from posted images” from leaking key information about tickets. The authors continued with Nakamoto's vision by creating a set of commercial-grade services supporting a wide variety of business use cases in [17]. In [18], the authors address the challenge of creating an electronic ticketing system for transportation systems that can partially or completely run on the cloud. However, there remains some shortcomings. For example, the unforgeability and immutable characteristic of tickets are not implemented. What’s worse, the scalpers' problem has not been solved. And the tickets can only be traded on a one-to-one basis with the purchased merchant, not across merchants.

Faced with the above problems, we will solve them through the consortium blockchain on the basis of previous frameworks. Blockchain can be divided into three types: public blockchain, private blockchain and consortium blockchain. Consortium blockchain’s main goal is to reduce cost and enhance efficiency. Its main technical features are the high performance, mass data, with strong identity license and security privacy. In general, the consensus nodes of consortium blockchain can be authenticated, and have high governance structure of the protocol or business rules. If an abnormity occurs in consortium blockchain, the regulatory mechanism can be enabled to track and punish or take further governance measures to reduce losses. For example, HyperLedger is the most active and recognized open source blockchain code projects [34]. Therefore, we developed a secure ETS based on consortium blockchain.

## 2. Related Work

There are lots of relevant works on ETS. In [6], the author studied the ticket issuing system, ticket checking system, retrieving system and automatic examination machine. In [7], the author realized an electric operation ticket expert system based on back propagation network and solved knowledge acquisition bottleneck in traditional expert system. An electronic ticket issuing system and an electronic ticket issuing method for simplifying authentication procedures by the use of biological information such as a fingerprint and voiceprint were proposed in [8]. In [9], the author proposed a portable ticket issuing system with a portable electric ticket issuing machine. In [10], the author studied a system which is disclosed for issuing airline tickets without the intervention of any ticket agent. The author developed a validation system that is capable of issuing the ticket in electronic form, in paper form, as a smart card, or as a season pass in [11]. In [12], a system for monitoring two or more persons in an area with an entry point having an access allowed indication or an access denied indication. However, for basic ETS, there remains lots of problems. For example, the scalper problem still remains open, which greatly damages the interest of consumers. This behavior breaks down the fairness and equality between consumers and merchants in transactions, and the value of goods and services that consumers receive is not equal to the monetary value they pay. What’s more, it affects consumers’ happiness index and the market price to cause market volatility. In [13], the author added an attribute (serial number) to the electronic tickets in order to avoid the ticket being sold twice by the ETS. In [14], the author hoped to ask the intervention of law to protect consumer rights in South Korea.

In recent years, the blockchain technology with decentralized nature, security and trust was introduced in ETS. In [15], the author studied the ETS in a specific scenario, that is, they introduced the blockchain system to prevent "ticket theft from posted images" from leaking key information about tickets. The ticketing system was realized based on a smart contract. But the organizer still has the right to modify the ticket status. In [16], the author designed a scheme for ETS based on the blockchain, and realized the electronic ticketing system with blockchain-BTS by using Ethereum and smart contracts. And in [17-18], the authors also study the electronic ticketing system with blockchain and the cloud server. In [19], the author made improvements to the existing blockchain protocol. To this end, they presented the pruneable sharding-based blockchain protocol by utilizing the sharding technique and PBFT(Practical Byzantine Fault Tolerance) algorithm in the improved Rollerchain, which has high efficiency, slow cubical dilatation, small capacity expansion and high scalability. This improvement can make blockchain better serve us. What's more, in [20-33], the authors studied different contents according to different attributes of the electronic ticketing system. Each work has made a great contribution to the improvement and development of the electronic ticketing system. For example, the paper applies the artificial neural network theory into traditional expert system development, presents realizes an electric operation ticket expert system based on back propagation network and solves knowledge acquisition bottleneck in traditional expert system in [22]. In [28], the author studies a kind of automated ticket sales and dispensing system. And the trusted NFC ticketing is studied in [32]. But users' privacy issues remain open. Therefore, we developed a secure ETS based on consortium blockchain (called CB-ETS). Among multiple authorization nodes, we used Delegated Proof of Stake (called DPoS) as the consensus mechanism. 101 authorization nodes were selected before the system establishment by a vote in the community to jointly manage the consortium blockchain. What's more, in order to protect users' anonymity effectively, we adopt the mixed currency protocol based on ring signature provided in [35].

The main contributions of this paper are as follows.

1. We established an ETS based on consortium blockchain with the DPoS consensus mechanism. It can effectively avoid the problem of resource consumption. Authorization nodes of DPoS maintain the consortium blockchain in a cooperative rather than competitive relationship, which improves the efficiency of the CB-ETS.

2. We embed a taxation mechanism into our ETS based on consortium blockchain. Combined with a regulation mechanism of transfer of tickets, if scalpers want to transfer a ticket in a high price, they will have to pay a high tax to authorization nodes in CB-ETS. This will effectively limit the malicious behaviors of scalpers. What's more, we optimize the transfer mechanism to achieve a win-win situation.

3. Our scheme guaranteed the users' consumption rights. In order to achieve the goal, we introduced the multi-signature in tickets' consumption, which we can avoid the tickets being cancelled unilaterally by the merchants. In addition, we optimize the transfer mechanism to achieve a win-win situation. Finally, we proved CB-ETS is secure by analyzing the ability to resist various attacks and it is efficient by implementing.

This paper is organized as follows. A brief background is introduced in Section 2. In Section 3, we establish the CB-ETS system based on the consortium blockchain, and then elaborate the behavior of scalpers and the corresponding solution. In section 4, we optimize the transfer mechanism to achieve a win-win situation. We analyze the security and performance of the scheme in Section 5. Finally, we make a summary in Section 6.

## 3. CB-ETS Model

In this section, we firstly introduce some notations that will be used in this paper, and then we give the security model and complexity assumption. Finally, we give the system model.

### 3.1 Symbols

The following notations shown in Table 1 will be used in this paper:

**Table 1.** The symbols

| Symbols | Explanations |
|---|---|
| $U_i$ | Users of Consortium blockchain |
| *Merchant* | Merchant of selling tickets in Consortium blockchain |
| $pk_i, sk_i$ | Public keys and secret keys of users |
| $pk_M, sk_M$ | Public keys and secret keys of Merchants |
| $cert_i$ | Certificate of users |
| $WID_i$ | Wallet address of users with *ID* |
| $m$ | The information of tickets |
| *Hash* | The Hash value of $m$ |
| $\sigma_i$ | The signature of users in buying process |
| $w_i$ | The signature of users in consuming process |
| $t_1, t_2, t_3$ | Timestamp |
| $Price, Price'$ | The original price and transferring price of tickets |
| $(-1,0,1,2)$ | Ticket age |
| $T^*$ | The transferring tickets |

This scheme mainly involves three parties: *users*, *nodes* and *Merchants*.

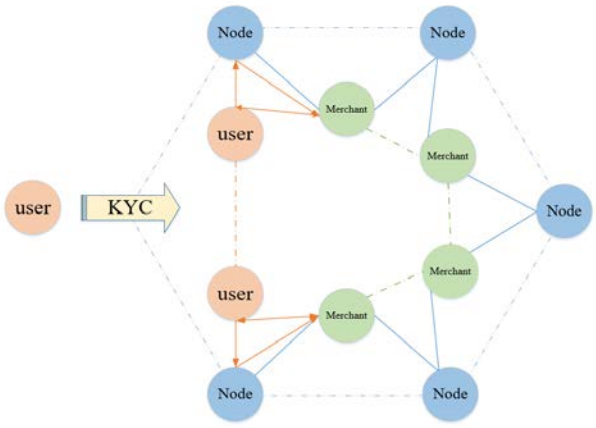


**Fig. 1.** Structure of our proposed consortium blockchain

*Users*: The ticket buyer. There are two main types of transactions for users: the transaction with *Merchants* and the transfer transaction on the user chain.

*Nodes*: Each organization has one or more pre-selected *Nodes* running. *Nodes* of the scheme are similar to the "miners" in the bitcoin system, which will wrap the transaction data in the transaction process into the consortium blockchain. *Nodes* collect all kinds of ticket requests and matches them with the relevant *Merchants*.

*Merchants*： Ticket seller and ticket counter. The ticket seller is responsible for generating new notes and trading with users and the ticket counter is responsible for verifying the validity of the tickets and the destruction of the tickets.

In this paper, DPoS is adopted as the consensus mechanism. Users and merchants conduct transactions and then send the transaction records to nodes, which collect transaction records and write to the block. The overall frame diagram of this paper is shown in **Fig. 1**.

### 3.2 Complexity assumption and Security Model

*Complexity assumption.* The Elliptic Curve Discrete Logarithm Problem (ECDLP). Given an elliptic curve $E$ defined over a finite field $F_q$, a point $P \in E(F_q)$ of order $n$, and a point $Q=lP$ where $0 \le l \le n-1$,determine $l$.

If an attacker can calculate A's private key from A's public key and public parameters, the attacker can forge A's signature at will. In our scheme, we show that ECDSA provides secure. We use ECDSA to sign. Based on the security model, if there exists a PPT algorithm that can attack the signature security with advantage at least $Adv(\mathcal{A})$, then there exists another PPT algorithm that can break ECDLP assumption. We aim to achieve the following goals:

*The Unforgeability and Anonymity of Tickets.* Since the privacy of users is involved in the ticket, data security is crucial. This paper mainly analyzes the anonymity of the ticket. Typically, the anonymity of the data is ensured by encryption and signature. The unforgerability of tickets is also the guarantee of users' rights, so the unforgerability of tickets is realized to ensure the uniqueness and authenticity, so that each process can be verified.

*Double Spending Attack.* Double spending is a kind of attack on blockchain. This paper establishes electronic ticketing system based on blockchain, so solving the problem is also one of our goals. This paper focuses on the payment problem in the purchase and transfer process of the tickets. When *Eve* pays, she may pay two or more tickets with the same amount of money in the process of completing the transaction but not the block being confirmed.

*Sale of Invalid Tickets.* When the user purchases a ticket, the users' interest may be harmed by the Merchant selling the invalid ticket or selling the duplicate ticket. We cite the blockchain system. It is an open and transparent distributed ledger. The transaction process of any ticket will be publicly recorded on the blockchain for public confirmation. If the ticket is invalid, the user can also apply to access the consortium blockchain for authentication.

*Modify the Invalidity of the tickets.* When a user purchases or consumes a ticket, the merchant needs to change the status of the ticket and sign it to make it invalidate. In literature [15], by this way, merchants' unlimited rights can modify the status of tickets to make them invalid arbitrarily. We will also focus on the security goal.

*Resistant to DoS Attacks.* The user may not respond to an address request during initialization, causing the entire process to fail. This is a Dos attack. There is another Dos attack in which an attacker prevents nodes from collecting and packaging transaction records onto blockchain, but this attack must satisfy 51% of the attacks.

*Resistant to Scalpers Attacks.* The scalper problem greatly damages the interest of consumers. This behavior breaks down the fairness and equality between consumers and merchants in transactions, so solving the scalpers problem is also our security goal.

### 3.3 CB-ETS system

*A. System initialization*

In CB-ETS, the scheme uses *ECDSA* and public key cryptography to initialize the system.

With strict *KYC* mechanism, each user is legally registered. A user $U_i$ who holds true identity *Idi* joins this system, and gets secret keys and certificate ($pk_i$, $sk_i$, $cert_i$), where $cert_i$ can only identify user with the binding registration information of the $U_i$. According to the *ECDSA* algorithm, $U_i$ selects an elliptic curve $Ep(a,b)$ and base point $G$. Then $U_i$ selects private key $sk_i$ and calculates public key $pk_i$ with base point $G$:

$$\begin{cases} sk_i = k(k < n) \\ pk_i = kG \end{cases} \tag{1}$$

$U_i$ sends his wallet address $WID_i$ to a third party, and the third party generates {$pk_i$, $sk_i$, $cert_i$, $WID_i$}. When $U_i$ runs System initialization, the wallet address is used by the nearest nodes account pool. After that, $U_i$ checks the integrity of the wallet and downloads relevant data about the wallet through the memory pool. The memory pool stores all transaction records in consortium blockchain. Moreover, *Merchant* gets his own secret keys $(sk_M, pk_M) = (k', k'G)$.

*B. Buying*

*Users* send the ticket request to *Nodes*, and *Nodes* broadcast the message to *local Merchants*. Then the *Merchant* provides timely feedback. After *Nodes* receive feedback, they match the purchase request made by *Users* with *Merchants*.

The *Users* and *Merchants* match successfully, and the transaction is conducted.

1) $U_i$ runs *UserSign* algorithm, and inputs his secret keys ($sk_i$, $pk_i$) and relevant parameters, then algorithm will output signature $\sigma_i$; The specific steps are as follows.

   A random integer $d$ ($d<n$ and $n$ is the order of $G$), is generated and computes $R$ and $r$.

$$\begin{cases} R(x, y) = dG \\ r = x \bmod n \end{cases} \tag{2}$$

   The ticket information $m$ and the coordinate values of point $R$ ($x$, $y$) are taken as parameters, and the *Hash* value and $s$ are calculated by using *SHA*1 as follows:

$$\begin{cases} Hash = SHA1(m, x, y) \\ s = (Hash + rk)d^{-1} \bmod n \end{cases} \tag{3}$$

   Where the signature is $\sigma_i = (s,d)$, and $x$ and $Hash(m)$ should be converted to integers.

2) $U_i$ sends $cert_i$, $pk_i$, $\sigma_i$, $m$ to *Merchant* to generate the purchase request

$$req_1 = \{cert_i, pk_i, \sigma_i\} \tag{4}$$

3) *Merchant* verifies the information received. If $cert_i$ exits or fails to verify, the request is refused. Of course, if $cert_i$ does not exit and verify successfully, and the specific verification steps are as follows.

   Calculate：

$$Hash = SHA1(m, x, y) \tag{5}$$

$$\begin{cases} u = s^{-1}Hash(m) \bmod n \\ v = s^{-1}r \bmod n \\ (x', y') = uG + vpk_i = uG + v(kG) \\ r' = x' \bmod n \end{cases} \tag{6}$$

   Where $x$ and $Hash(m)$ should be converted to integers.

   Verify：

$$r \overset{?}{=} r' \tag{7}$$

If the equation (7) holds, the signature is accepted; Otherwise the signature is invalid.

Correctness.

$$
\begin{aligned}
&\because pk_i = kG, s = (Hash(m) + rk)d^{-1} \bmod n \\
&\quad dG = (x, y), u = s^{-1}Hash(m) \bmod n \\
&\quad v = s^{-1}r \bmod n, (x', y') = uG + vpk_i \\
&\therefore d \equiv (Hash(m) + rk)s^{-1} \equiv (s^{-1}Hash(m) + s^{-1}rk) \\
&\quad \equiv (u + vk) \bmod n \\
&\quad \therefore (x, y) = dG = uG + vkG = (x', y') \\
&\quad r' = x' \bmod n = x \bmod n = r \\
&\quad \therefore r' = r
\end{aligned} \tag{8}
$$

Then *Merchant* accepts the request and stores the information ($cert_i$, $pk_M$ ,$pk_i$, 1, 0) in the local accounts pool; (1 is on behalf of the valid ticket, and 0 is on behalf of the state of tickets for the new generation, which has no ticket transferring. Also, -1 is on behalf of the state of tickets for the waiting for ticket transferring. 2 is on behalf of the state of the non-assignable tickets which have been transferred once.)

**Table 2.** The Buying Algorithm

| **The Algorithm 1** |
|---|
| 1. $\sigma_i \leftarrow UserSign(cert_i, sk_i, 1^k)$; |
| 2. *Merchant*$\leftarrow$*Request* 1($cert_i$, $pk_i$, $\sigma_i$)$\leftarrow U_i$; |
| 3. $b \leftarrow$ *Merchant* checks $cert_i$ and runs *Verify*((*enroll*, $cert_i$, $pk_M$, $pk_i$), $\sigma_i$, $pk_i$)=1; |
| 4. if $b$=1 or ((*enroll*, $cert_i$, $pk_M$, $pk_i$), $\sigma_i$, $pk_i$)=0, then abort;<br>if $b$=0 and ((*enroll*, $cert_i$ ,$pk_M$, $pk_i$), $\sigma_i$, $pk_i$)=1,then accept and storage ($cert_i$ $pk_M$ ,$pk_i$,1, 0); |
| 5. $\sigma_1 \leftarrow MerchantSign$((*enroll*, $cert_i$ , $pk_m$ ,$pk_i$ ) ,$t_1$ ,$sk_M$); |
| 6. *blockchain*$\leftarrow$($\sigma_1$ , *request* 1). |

4) *Merchant* uses the private key $sk_M$ to sign for (enroll, $cert_i$, $pk_m$, $pk_i$, $t_1$) by running *MerchantSign* like *UserSign*， and then sends ($\sigma_1$ , *request*1) to the *blockchain*, where $t_1$ refers to the time of purchasing tickets. The specific steps are as follows. Recording to the functions (2-3), *Merchant* runs the Algorithm and signs to $m_1$ :

$$
\begin{cases}
R(x, y) = d_1 G \\
r_1 = \overline{x} \bmod n
\end{cases} \tag{9}
$$

$$
\begin{cases}
Hash_1 = SHA1(m_1, x, y) \\
s_1 \equiv d_1^{-1}(Hash_1 + kr_1) (\bmod n)
\end{cases} \tag{10}
$$

Where $m_1$={$enroll, cert_i, pk_m, pk_i, t_1$} , $x$ and $Hash(m_1)$ should be converted to integers. The signature is $\sigma_1$=($s_1, r_1$) . $U_i$ verifies the signature signed by Merchant like the functions (5-8). The specific algorithm is shown in **Table 2**.

### *C. Paying*

After the transaction, $U_i$ pays the ticket coins through a given *Merchant's* wallet address. The *Merchant* confirms whether the block containing the transaction is existing and effective or not. If not, *Merchant* stops the transaction. If yes, $U_i$ generates transaction records, and *Merchant* verifies the record and signs on it. Finally, they upload it to *Nodes* for reviewing. In this phase, the mixed currency protocol based on ring signature is added to ensure the anonymity of the user. We modified the mixed currency protocol [35] in the bitcoin system, in order to be applicable to CB-ETS. The specific procedure is as follows.

In the *requesting* phase, a user desiring to mix his ticket coins sends an initial request message to hybrid server. The request message comprises the public key of the user $pk_i$ and the

transactions he wants to mix. After receiving the request from a user, the hybrid server sends back a certain amount of public keys $\{pk_1, pk_2, \cdots, pk_n\}$ collected from users. The amount of public keys $n$ should be adjusted in consideration of the server performance.

In the *generation* phase, the user receives public keys of other users and generates ticket coin addresses $\{MID_1, MID_2, \cdots, MID_m\}$for getting back his own ticket coins after mixing. To obtain $addr_m$, a public key is hashed with *SHA*-256 algorithm first and *RIPEMD*-160 algorithm. After concatenating a check sum and a version number with the hash value, it is encoded through a special *base58* to generate a valid address $addr_m$. Then he signs all generated addresses through the ring signature and sends them back to hybrid server one by one. Upon receiving the response from the user, the hybrid server generates a mix transaction containing all the input and output addresses, and sends it to corresponding customers respectively.

In the *final* confirmation phase, the user will check whether the mixing transaction contains all his input transactions and output addresses or not. If all the information included in the mixing transaction is corrected, the user will sign the mixing transaction and broadcast it in the blockchain as normal transactions.

An initial request message $m_r$ consists of user's public key $pk_i$ and transactions going to be mixed. The response message from hybrid server contains all public keys of customers, who want to mix their own transactions. Each address message $m_{addrn}$ includes only one ticket coin address and amount of ticket coin transfers to this address. At last, the user checks whether the mixing transaction $m_{mix}$ containing all the amount of ticket coins corresponding with his addresses is correct or not. If yes, the mixing transaction succeed. If not, the user can stop and reinitiate a request to hybrid server.

*D. Consuming*

When $U_i$ consumes this ticket, he still trades with *Merchant*. What's more, *Merchant* and $U_i$ will perform a 2-out-of-2 multi-signature for the ticket, i.e. double signature. They jointly decide the validity of the ticket. The specific steps are as follows:

1) $U_i$ runs the *UserSign* and signs the ticket: $\omega_i \leftarrow UserSign(Ticket, t_2)$;

   Firstly, $U_i$ selects secret integer $d_2$ and calculates $Hash(m_2)$ and $d_2G = (x_i, y_i)$.

   Then calculates：

$$a_i = x_i \bmod n \tag{11}$$

$$\begin{cases} d_2^{-1} \bmod n \\ b_i = d_2^{-1}(m_2 + ka_i) \bmod n \end{cases} \tag{12}$$

$$s_i = b_i^{-1} \bmod n \tag{13}$$

   The signature of $m_2$ is $\omega_i = Sign(m_2) = (a_i, s_i)$. Where $a_i$ and $b_i$ cannot be 0, otherwise the calculation will be recalculated.

2) $U_i$ sends $pk_i$, $\sigma_i$, $Hash(m_2)$ and $\omega_i$ to *Merchant*;

3) *Merchant* verifies the validity of $cert_i$ and runs successively $Verify(invalidate, \omega_i, pk_i)$、$Verify(invalidate, \sigma_i, pk_i)$ and $Verify((invalidate, pk_i, ticket\ age), \sigma_i, pk_M)$. If verifications fail, then refuses. If verifications are successful, then *Merchant* modifies the ticket status to make it invalid. Of course, the modified ticket must be signed by $U_i$, otherwise it is deemed incorrect, where $t_2$ is the current time to add to the ticket information.

Firstly, *Merchant* verifies whether $a_i$ and $s_i$ are all integers less than $n$ and not equal to 0. If not, reject. If yes, calculate as follows:

$$\begin{cases} u' = Hash(m_3) \cdot s_i \bmod n \\ v' = a_i s_i \bmod n \end{cases} \tag{14}$$

$$P=u'G+v'(kG) \tag{15}$$

$$z=x_p \bmod n \tag{16}$$

Where $x_p$ is the $x$-coordinate of the point $P$ and $m_3=\{m_2,t_2\}$.

Then verify：

$$z \overset{?}{=} a_i \tag{17}$$

Correctness.

$$\begin{aligned}
&\because b_i=d_2^{-1}(m_2+ka_i) \bmod n,\ s_i=b_i^{-1} \bmod n \\
&\therefore d_2=s_i(m_2+ka_i) \bmod n \\
&\quad =(s_i m_2 \bmod n+s_i k a_i \bmod n) \bmod n \\
&\because v'=a_i s_i \bmod n, v'+tn=a_i s_i \\
&\therefore ka_i s_i \bmod n=k(v'+tn) \bmod n=kv' \bmod n \\
&\because u'=m_2 s_i \bmod n \\
&\therefore d'=(u'+kv' \bmod n) \bmod n=(u'+kv') \bmod n \\
&\therefore P=u'G+v'(kG)=(u'+kv')G=d_2G=x_i \\
&\therefore z=a_i
\end{aligned} \tag{18}$$

4) *Merchant* verifies successfully and runs *MerchantSign* to sign the information.

Firstly, Merchant selects the secret integer $d_3$ and calculates the signature $\sigma_2=\{a_M, s_M\}$.

$$d_3G=(x_M,y_M) \tag{19}$$

$$a_M=x_M \bmod n \tag{20}$$

$$\begin{cases} d_3 \bmod n \\ b_M=d_3^{-1}(m_3+k'a_M) \bmod n \end{cases} \tag{21}$$

$$s_M=s_i+b_M^{-1} \bmod n \tag{22}$$

Then *Merchant* sends ($\sigma_2$, *request* 2) to *blockchain* together.

The specific algorithm is shown in **Table 3**.

**Table 3.** The Consuming Algorithm

| **The Algorithm 2** |
|---|
| 1. $\omega_i$←*UserSign*(*m'*); |
| 2. *Request* 2($pk_i$, $\sigma_i$, *Hash*(*m'*), $\omega_i$)←$U_i$; |
| 3. *Merchant*←*request* 2; |
| 4. *b*←*Merchant* checks $cert_i$ and run *Verify*(*invalidate*, $\omega_i$, $pk_i$), *Verify*(*invalidate*, $\sigma_i$, $pk_i$) and *Verify*((*invalidate* , $cert_i$ , $pk_i$ , *ticket age*), $\sigma_i$, $pk_M$)=1; |
| 5. if *b*=0 or ((*invalidate*, $cert_i$ , $pk_i$ , *ticket age*), $\sigma_i$, $pk_M$)=0, then refuse; if *b*=1 and ((*invalidate*, $cert_i$ , $pk_i$ , *ticket age*), $\sigma_i$, $pk_M$)=1,then accept and change ($cert_i$, $pk_M$, $pk_i$, 1, *ticket age*, $t_0$) into ($cert_i$, $pk_M$, $pk_i$, 0, $\omega_i$, *ticket age*, $t_1$, $t_2$); |
| 6. $\sigma_2$← *MerchantSign*((*invalidate*, $cert_i$, $pk_i$, 0, *ticket age*, $\omega_i$, $t_1$, $t_2$), $sk_M$); |
| 7. *blockchain*←($\sigma_2$ , *request* 2). |

*E. Building blocks*

Before running the system, 101 super nodes are elected by voting in the community in the consortium blockchain, which are called *Nodes*. In this paper, DPoS is adopted as the consensus algorithm of CB-ETS. When BM made the first version of DPOS, he targeted 101 producers, all of which were elected by voting. Bitshare2.0 adjusted the number of 101 to be user-defined, so that when people vote, they can freely adjust the number of votes. In order to

achieve a more "decentralized" goal, the more nodes there are, the better it is to run the system. So we select 101 *Nodes* which has been running for years to prove that it works. *Nodes* are selected to be the *Leader* of the next moment in a random manner, then they collect and write transaction records to the blockchain.(The *Nodes* are reshuffled each time to prevent the *Nodes* from colluding with each other because of some connection). When 101 *Nodes* end a round, then re-vote 101 nodes and continue the above process. *Nodes* collect all local transaction records within a certain period of time and then encrypt and sign these records to ensure authenticity and accuracy. These transaction records are constructed into blocks that contain a hash value pointing to the previous block in the blockchain.

### *F. Carrying out consensus process*

As the *Leader* in the consensus process, he will broadcast the data and time stamps to the blockchain for verification and check of the other 100 *Nodes*. These *Nodes* check the block data and verify the results with the *Leader*'s signature. After receiving the results, each *Node* compares the own results with the results of the other *Nodes*, then sends the results (including their own results, comparison results, the own records of the results) back to the *Leader*. The *Leader* counts the replies received. If all *Nodes* agree on the block data, the *Leader* sends the current record of the block data reviewed and the corresponding signature to all the authorized *Nodes*. After that, the block is stored in the consortium blockchain in chronological order. If all the authorized *Nodes* have different data on the block, the *Leader* will analyze all results received. If all results are correct, *Leader* could send the data again to confirm whether *Nodes* go wrong or not. In addition, according to the results and corresponding signature, the malicious *Nodes* can be supervised. If Nodes have malicious behavior, the system could discover timely and take further measures to recovery system.

After the successful broadcast of the block, we can read the detailed block content. The data structure is shown in **Fig. 2**. It consists of block header, payload, signature of the contributor, and timestamp. Block header concludes three components: Block ID, block size, and hash value of previous block. Payload has two parts: pseudo identity of buyer and encrypted ticket hash. Specifically, the block stores the pseudo identity of the data sender, which is derived from the true identity. Contributor signature helps to track the generator of the block. Timestamp shows the generation time of the block.

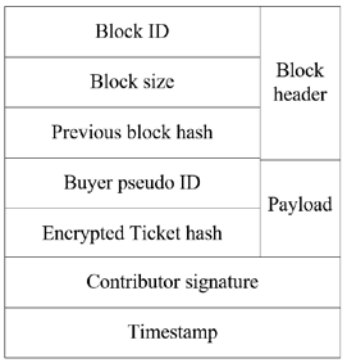


**Fig. 2.** Structure of Blockchain

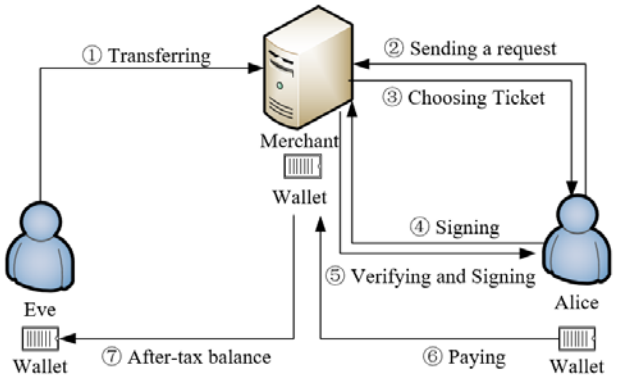


**Fig. 3.** Structure of our proposed transferring of tickets

### *G. Transactions on the user chain—The transf.erring of tickets*

This section will focus on the problem of scalpers. According to the characteristics of the scalpers, we introduce a new concept of "tax", and design a taxation mechanism. It makes scalpers to seek few interest to force them to give up transferring the ticket with a higher price. The frame diagram of the transferring of tickets is shown in **Fig. 3**.

1) *Eve* sends a transferring request *request* 3 to *Merchant*;

$$request\ 3 = (T^*, Price', MID_{Eve}) \tag{23}$$

where $T^*$ refers to the ticket and *Price'* is the price of *Eve*'s transferring ticket.

2) *Merchant* verifies the validity of $T^*$. If it is not valid, reject directly. If valid, accept.

*Merchant* verifies the validity of $cert_i$ and runs successively *Verify*(*invalidate*, $\omega_i$, $pk_i$), *Verify*(*invalidate*, $\sigma_i$, $pk_i$) and *Verify*((*invalidate*, $pk_i$, *ticket age*), $\sigma_i$, $pk_M$). If verifications fail, then refuses. If verifications are successful, then *Merchant* modifies the ticket status to make it invalid, that is *Merchant* changes ($cert_i$, $pk_M$, $pk_i$, 1, *ticket age*, $t_1$) into ($cert_i$, $pk_M$, $pk_i$, -1, $\omega_i$, *ticket age*, $t_1$, $t_3$). Of course, the modified ticket must be signed by $U_i$, otherwise it is deemed incorrect, where $t_3$ is the current time. See the formulas (11-21) for the specific operations.

3) *Merchant* changes the state of $T^*$ from 1 to -1 and stores, then broadcasts ($T^*$, *Price'*);

4) *Alice* sends the purchase request to *Nodes*, while *Nodes* does not receive any response after broadcasting to the local *Merchants*. Therefore, *Nodes* should respond to *Alice* without a ticket in time.

5) *Alice* receives the ticket transferring message and trades with *Merchant* directly;

6) According to (*T*, *Price*), *Alice* selectes the transferring tickets. If $T^*$ is selected, the tickets will be purchased by *Alice*.

7) *Merchant* sends $T^*$ to *Alice*, and *Alice* signs it and returns to *Merchant*. *Merchant* verifies the signature, and changes ($cert_i$, $pk_M$, $pk_i$, -1, *ticket age*, $t_1$) into ($cert_i$, $pk_M$, $pk_i$, 2, $\omega_i$, *ticket age*, $t_1$, $t_4$) and signs (the ticket can only be transferred once) if the verification is passed. Then *Merchant* covers the time stamp on the ticket.

**Table 4.** The Transferring Algorithm

| **The Algorithm 3** |
|---|
| 1. *Merchant*←*request* 3 = ($T^*$,*Price'*,$MID_{Eve}$) ←*Eve*; |
| 2. *b'*← *Merchant* checks $cert_{eve}$ and run *Verify*(*invalidate*, $\sigma_{eve}$, $pk_{eve}$), *Verify* (*invalidate*, $\omega_{Eve}$, $pk_{Eve}$) and *Verify*((*invalidate*, $cert_{eve}$, $pk_{eve}$ , *ticket age*), $\sigma_{eve}$, $pk_M$)=1; |
| 3. if *b'*=0 or ((*invalidate*, $cert_{eve}$ , $pk_{eve}$ , *ticket age*), $\omega_{Alice}$ , $\sigma_{eve}$, $pk_M$)=0, then refuse;<br>if *b'*=1 and ((*invalidate*, $cert_{eve}$ , $pk_{eve}$ , *ticket age*), $\omega_{Alice}$ , $\sigma_{eve}$, $pk_M$)=1,then accept and change ($cert_{eve}$, $pk_M$, $pk_{eve}$, 0, *ticket age*, $t_1$) into ($cert_{eve}$, $pk_M$, $pk_{eve}$, -1, *ticket age*+1, $\omega_{Alice}$, $t_1$, $t_3$); |
| 4. *blockchain*←(*T*, Price'*) ←*Merchant*; |
| 5. *Alice*←*local nodes*(NO!←broadcast 1(*local nodes*(*Request*(*Alice*)))); |
| 6. *Alice*←broadcast (*T*, Price'*) ←*Merchant*; |
| 7. *Merchant* ← ($\alpha$, $\omega_{Alice}$ ←*UserSign*$_{Alice}$(*T*, Price'*, $t_3$))←*Alice*; |
| 8. *b'*← *Merchant* checks $cert_{eve}$ and run *Verify*(*invalidate*, $\sigma_{eve}$, $pk_{eve}$), *Verify*(*invalidate*, $\omega_{Alice}$, $pk_{Alice}$) and *Verify*((*invalidate*, $cert_{eve}$, $pk_{eve}$ , *ticket age*), $\sigma_{eve}$, $pk_M$)=1; |
| 9. if *b'*=0 or ((*invalidate*, $cert_{eve}$ , $pk_{eve}$ , *ticket age*), $\omega_{Alice}$ , $\sigma_{eve}$, $pk_M$)=0, then refuse;<br>if *b'*=1 and ((*invalidate*, $cert_{eve}$ , $pk_{eve}$ , *ticket age*), $\omega_{Alice}$ , $\sigma_{eve}$, $pk_M$)=1,then accept and change ($cert_{eve}$, $pk_M$, $pk_{eve}$, -1, *ticket age*, $t_1$) into ($cert_{eve}$, $pk_M$, $pk_{eve}$, 2, *ticket age*+1, $\omega_{Alice}$, $t_1$, $t_3$); |
| 10. $\sigma_3$← *MerchantSign*((*invalidate*, $cert_{eve}$, $pk_{eve}$, *ticket age*+1, $\omega_{Alice}$, $t_1$, $t_3$), $sk_M$); |

7.1) If the ticket is valid, *Alice* signs on the ticket and sends the related information to *local Merchant*. The *Merchant* verifies $cert_{eve}$ and runs *Verify*(*invalidate*, $s_{eve}$, $pk_{eve}$) , *Verify*(*invalidate*, $\omega_{Alice}$, $pk_{Alice}$) and *Verify*((*invalidate*, $pk_{eve}$, *ticket age*), $s_{eve}$, $pk_M$) successively. If the verifications are invalid, then aborts. If the verifications succeed, then *Merchant* modifies the ticket information, that is *Merchant* changes ($cert_{eve}$, $pk_M$, $pk_{eve}$, -1,

*ticket age*, $t_1$) into ($cert_{eve}$, $pk_M$, $pk_{eve}$, 2, *ticket age*+1, $UserSign_{sk(Alice)} \rightarrow s_{Alice}$, $t_1$, $t_3$), where the ticket age plus 1 represents the transferring of the ticket one time. In addition, the modified ticket information should add *Alice's signature*, meaning the ticket owned by *Alice* now, otherwise considered the behavior is invalid. $t_3$ is the current time.

7.2) *Merchant* runs *MerchantSign*， and signs onto the information, then sends ($\sigma_3$ , *request* 4) to *blockchain* together. See the formulas (11-21) for the specific operations.

The specific algorithm is shown in **Table 4**.

8) *Alice* pays ticket coins to *Merchant*. Then Merchant broadcasts to the blockchain.

9) *Merchant* collects taxes on the fare spread as a transaction fee deduction, which transfers the remainder *Actual Transaction Price* (*ATP*) to $MID_{Eve}$. *Merchant* calculates the total tax, according to "*Full rate progressive tax rate*". (This scheme simply modifies the individual income tax of China. The rates are shown in **Table 5**.) Of course, there is no difference and tax in the ordinary transferring of tickets. The tax formula is calculated as follows.

$$\begin{cases} MID_{Eve} = Price' - Tax \\ Tax = (Price' - Price) \cdot TaxRate \end{cases} \tag{24}$$

**Table 5.** The difference tax rate

| Tax payable/coins | Tax rate |
|---|---|
| Difference ≤3 | 53% |
| 3 < Difference ≤9 | 60% |
| 9 < Difference ≤18 | 70% |
| 18 < Difference ≤ 70 | 75% |
| 70 < Difference ≤ 110 | 80% |
| 110 < Difference ≤ 160 | 85% |
| 160 < Difference | 95% |

10) The reputation mechanism will be added in this paper, in which *Eve*'s reputation will be determined by the number of transferring times and the price difference of each transferring ticket, so his certain rights will be restricted accordingly. Concrete action: the reputation mechanism starts from the user registration, and the initial value is zero. A reputation score of 5 points is awarded for each successful purchase of a ticket. If *Eve* wants to transfer the ticket, the reputation score shall be deducted according to the number and price difference of *Eve*'s transferring of the ticket. If there is a transfer difference, the reputation score is deducted according to the multiple of the transfer difference. That is, if *Eve* transfers once, her reputation score ($\Delta Price$) will be deducted; if she transfers $x$ times, the corresponding ($\Delta Price \cdot x$) score will be deducted. If the reputation score is less than 500 points, the tickets may not be transferred. If the reputation score is more than 1500 points, the transaction fee for the transferring is deducted at 80% as an incentive.

## 4. Optimization of Transferring Mechanism

In this section, we optimize the transfer mechanism to maximize the benefits of both parties and achieve a win-win situation. The transferee is *Eve* and the buyer is *Alice*.

Let us denote a set of buyers of tickets as $BT_i^n (i \in B, B = \{0,1,...,I\})$ and the transferees of tickets as $TT_j^n (j \in T, T = \{0,1,...,J\})$. $b_i^{n,\max}$ is the maximum of buying tickets, and $t_j^{n,\max}$ is also the maximum of transferring tickets. The value of $b_i^{n,\max}$ and $t_j^{n,\max}$ have to do with the credit value of every user. Therefore, the satisfaction function of $BT_i^n$ and the cost function of $TT_j^n$ :

$$A_i(BT_i^n) = w_i \ln(b_i^{n,\max} - \eta \sum_{j=1}^{J} b_{ij}^n + 1) \tag{25}$$

$$E_j(TT_j^n) = l_1 \sum_{i=1}^{I} t_{ji}^n + l_{2j} \ln(\eta \sum_{i=1}^{I} t_{ji}^n - t_j^{n,\max} + 1) \tag{26}$$

Where $w_i$ is the willingness of buying tickets for $BT_i^n$ and $\eta$ is the average of transferring tickets. $l_1$ and $l_2$ are all cost factors. For $t_{ji}^n > t_j^{n,\max}$, $l_2$ is effective. This ensures that even if there are extra tickets to be transferred, the tickets are valid, but the profits of the transferee are limited for $TT_j^n$. Accordingly, his credit rating will fall. When transferring, Merchant acts as a third-party platform for both parties to achieve win-win. That is, $BT_i^n$ can maximize interest and $TT_j^n$ can minimize costs. Therefore, Merchant build the following target planning:

$$\max_{B^n, T^n} \sum_{i=1}^{I} A_i(BT_i^n) - \sum_{j=1}^{J} E_j(TT_j^n) \tag{27}$$

$$\begin{cases} 0 \le \eta \sum_{i=1}^{I} b_{ij}^n \le b_i^{n,\max} \\ 0 \le \eta \sum_{j=1}^{J} t_{ji}^n \le t_j^{n,\max} \quad (\forall i \in B, \ \ \forall j \in T) \\ b_{ij}^n, \ \ t_{ji}^n \ge 0 \end{cases} \tag{28}$$

The function (27) is strictly concave with compact and convex constraints, so there exists a unique optimal solution using *Karush-Kuhn-Tucker* (*KKT*) conditions. We carry out relaxation of constraints yielding the following *Lagrangian* $L_1$:

$$\begin{aligned} L_1(B^n, T^n, \alpha, \beta, \gamma, \lambda, \mu, \varepsilon) = & \sum_{i=1}^{I} A_i(BT_i^n) - \sum_{j=1}^{J} E_j(TT_j^n) + \sum_{i=1}^{I} \alpha_i(-\eta \sum_{i=1}^{I} b_{ij}^n) \\ & + \sum_{i=1}^{I} \beta_i(\eta \sum_{i=1}^{I} b_{ij}^n - b_i^{n,\max}) + \sum_{j=1}^{J} \gamma_j(-\eta \sum_{j=1}^{J} t_{ji}^n) + \sum_{j=1}^{J} \lambda_j(\eta \sum_{j=1}^{J} t_{ji}^n - t_j^{n,\max}) \\ & + \sum_{i=1}^{I} \sum_{j=1}^{J} \mu_{ij} b_{ij}^n + \sum_{i=1}^{I} \sum_{j=1}^{J} \varepsilon_{ij} t_{ji}^n \end{aligned} \tag{29}$$

Here, $\alpha_i, \beta_i, \gamma_j, \lambda_j, \mu_{ij}$ and $\varepsilon_{ij}$ are Lagrange multipliers for the constraints. The corresponding sets are $\alpha, \beta, \gamma, \lambda, \mu$ and $\varepsilon$. From the stationary conditions, the optimal solution meets:

$$\nabla_{b_{ij}^n} L_1(B^n, T^n, \alpha, \beta, \gamma, \lambda, \mu, \varepsilon) = -\frac{\eta w_i}{b_i^{n,\max} - \eta \sum_{j=1}^{J} b_{ij}^n + 1} - \eta \alpha_i + \eta \beta_i + \mu_{ij} = 0 \tag{30}$$

$$\nabla_{t_{ji}^n} L_1(B^n, T^n, \alpha, \beta, \gamma, \lambda, \mu, \varepsilon) = -l_1 - \frac{l_{2j} \eta}{\eta \sum_{i=1}^{I} t_{ji}^n - t_j^{n,\max} + 1} - \eta \gamma_j + \eta \lambda_j + \varepsilon_{ij} = 0 \tag{31}$$

What's more, before the transferring tickets, the buyer *Alice* sets the price first, in which the price vector is $BT_i^n = \{a_{ij}^n \mid j \in Z\}, BT^n = \{BT_i^n \mid i \in C\}$. Then her optimal pricing is as follows.

$$\max_{BT_i^n} [A_i(BT_i^n) - pay_i(A_i^n)] \tag{32}$$

*Eve* sets the price and price vector is $TT_j^n = \{e_{ji}^n \mid i \in C\}, TT^n = \{TT_j^n \mid j \in Z\}$. The optimal pricing:

$$\max_{TT_j^n} [E_j(TT_j^n) - tax_j(E_j^n)] \tag{33}$$

*Merchant* targets the pricing as follows to calculate the amount of transferring tickets:

$$\max_{BT^n,\ TT^n} \sum_{i=1}^{I}\sum_{j=1}^{J}[a_{ij}^n \ln b_{ij}^n - e_{ji}^n t_{ji}^n] \tag{34}$$

where the equality (34) has the same constraints (28) as the equality (27).

The equality (34) is strictly concave with compact and convex constraints, so there exists a unique optimal solution using *Karush-Kuhn-Tucker* (*KKT*) conditions. We carry out relaxation of constraints yielding the following *Lagrangian* $L_2$:

$$\begin{aligned} L_2(B^n,T^n,\alpha,\beta,\gamma,\lambda,\mu,\varepsilon) &= \sum_{i=1}^{I}\sum_{j=1}^{J}[a_{ij}^n \ln b_{ij}^n - e_{ji}^n t_{ji}^n] + \sum_{i=1}^{I}\alpha_i(-\eta\sum_{i=1}^{I} b_{ij}^n) \\ &+ \sum_{i=1}^{I}\beta_i(\eta\sum_{i=1}^{I} b_{ij}^n - b_i^{n,\max}) + \sum_{j=1}^{J}\gamma_j(-\eta\sum_{j=1}^{J} t_{ji}^n) + \sum_{j=1}^{J}\lambda_j(\eta\sum_{j=1}^{J} t_{ji}^n - t_j^{n,\max}) \\ &+ \sum_{i=1}^{I}\sum_{j=1}^{J}\mu_{ij} b_{ij}^n + \sum_{i=1}^{I}\sum_{j=1}^{J}\varepsilon_{ij} t_{ji}^n \end{aligned} \tag{35}$$

Here, $\alpha_i,\beta_i,\gamma_j,\lambda_j,\mu_{ij}$ and $\varepsilon_{ij}$ are Lagrange multipliers for the constraints. The corresponding sets are $\alpha,\beta,\gamma,\lambda,\mu$ and $\varepsilon$. From the stationary conditions, the optimal solution of *SW* meets following conditions:

$$\nabla_{b_{ij}^n} L_2(B^n,T^n,\alpha,\beta,\gamma,\lambda,\mu,\varepsilon) = \frac{a_{ij}^n}{b_{ij}^n} - \eta\alpha_i + \eta\beta_i + \mu_{ij} = 0 \tag{36}$$

$$\nabla_{t_{ji}^n} L_2(B^n,T^n,\alpha,\beta,\gamma,\lambda,\mu,\varepsilon) = -e_{ji}^n - \eta\gamma_j + \eta\lambda_j + \varepsilon_{ij} = 0 \tag{37}$$

Therefore, the optimal solution of simultaneous equations (30)(31)(36)(37) is as follows:

$$a_{ij}^n = \frac{\eta w_i b_{ij}^n}{b_i^{n,\max} - \eta\sum_{j=1}^{J} b_{ij}^n + 1} \tag{38}$$

$$e_{ji}^n = l_1 + \frac{l_{2j}\eta}{\eta\sum_{i=1}^{I} t_{ji}^n - t_j^{\max} + 1} \tag{39}$$

When all three parties solve their optimization problems, *Merchant* makes pricing according to the market and obtains the following pricing rules:

$$pay_i(A_i^n) = \sum_{j=1}^{J} a_{ij}^n \tag{40}$$

$$tax_j(E_j^n) = \sum_{i=1}^{I}\frac{(e_{ji}^n - price)^2}{4} = \sum\frac{(price' - price)^2}{4} \tag{41}$$

Correctness. From the equation (32)， we have known the optimal buying price satisfies the following condition $\frac{\partial A_i(BT_i^n)}{\partial a_{ij}^n} - \frac{\partial pay_i(A_i^n)}{\partial a_{ij}^n} = 0$.

According to the equation (40)，we obtain $\frac{\partial A_i(BT_i^n)}{\partial a_{ij}^n} = \frac{\partial A_i(BT_i^n)}{\partial b_{ij}^n}\cdot\frac{\partial b_{ij}^n}{\partial a_{ij}^n} = \frac{\partial pay_i(A_i^n)}{\partial a_{ij}^n} = 1$.

Hence， $a_{ij}^n = \frac{\partial A_i(BT_i^n)}{\partial b_{ij}^n}\cdot b_{ij}^n = \frac{\eta w_i b_{ij}^n}{b_i^{n,\max} - \eta\sum_{j=1}^{J} b_{ij}^n + 1}$， we can see it is the same with the equation (38) and it is correct. According to the equation (41)， we obtain

$$\frac{\partial E_j(TT_j^n)}{\partial e_{ji}^n}=\frac{\partial E_j(TT_j^n)}{\partial t_{ji}^n}\cdot\frac{\partial t_{ji}^n}{\partial e_{ji}^n}=-\frac{\partial tax_j(E_j^n)}{\partial e_{ji}^n}=\frac{e_{ji}^n-price}{2}$$

Hence，

$$\frac{\partial t_{ji}^n}{\partial e_{ji}^n}=\frac{\partial tax(E_j^n)}{-\partial e_{ji}^n\cdot\frac{\partial E_j(TT_j^n)}{\partial t_{ji}^n}}=\frac{\partial tax(E_j^n)}{\partial e_{ji}^n\cdot l_1+\frac{l_{2j}\eta}{\eta\sum_{i=1}^{I}t_{ji}^n-t_j^{\max}+1}}=\frac{e_{ji}^n-price}{2(l_1+\frac{l_{2j}\eta}{\eta\sum_{i=1}^{I}t_{ji}^n-t_j^{\max}+1})}$$

Based on the above equation， we consider that there exist linear correlations between $t_{ji}^n$ and $e_{ji}^n$. Therefore， $e_{ji}^n=l_1+\frac{l_{2j}\eta}{\eta\sum_{i=1}^{I}t_{ji}^n-t_j^{\max}+1}$， we can see it is the same with (39)and it is correct.

Finally we simulate the optimization and get optimal bid prices with an example in section 5.5.

## 5. Security and performance Analysis

In this section, we give detailed security and performance analyses of the whole scheme. This section is divided into four parts. We introduce the security model in our scheme firstly. Secondly, based on the model, some important properties are also discussed. And then the performance evaluation is presented. Finally, we give the efficiency analysis of our scheme.

### 5.1 Security model

Our scheme is based on the elliptic curve cryptosystem, so it is difficult for an attacker to solve the signer's private key through the signer's public key and master domain parameters, which is equal to the difficulty in solving the elliptic curve discrete logarithm problem. What's more, it is also not possible for an attacker to get the private key through the signer's signature $\sigma_i$ because equation $d=(w_i-\sigma_i)(m+sk_ia_i)\bmod n$ has two unknowns parameters: $d$ and $sk_i=k$.

Theorem 1. No adversary can forge a signature for any message with greater than negligible probability, even if that adversary has seen signatures for polynomially many messages of its choice. Formally, for all PPT adversaries $\mathcal{A}$ with access to the signing oracle $Sign_{sk}(\cdot)$, where $Q$ is the set of queries $\mathcal{A}$ asks the oracle:

$$\Pr_{pk,sk}[Verify_{pk}(m,\sigma)=1\wedge m\notin Q:(m,\sigma)\leftarrow\mathcal{A}^{Sign_{sk}(\cdot)}(pk)]<negl(k)$$

### 5.2 Security analysis

*a. The Unforgeability and Anonymity of Tickets*

The tickets, whether are purchased, transferred or consumed, all require the signature of users and merchants. Consider the following attack scenario: suppose adversary $\mathcal{A}$ tries to get away with paying by forging the Merchant's signature. $\mathcal{A}$ refers to the adversary. $E_{(k_{\mathcal{A}}^{-1})}$ refers to $\mathcal{A}$ signs the tickets with his private key and $E_{k_M}$ refers to *Merchant* signs the tickets with his own private key. *Ticket* refers to some specific information of the ticket. $\mathcal{A}$ refers to the information related to $\mathcal{A}$ on the ticket. timestamp refers to $\mathcal{A}'$s timestamp when forging the signature and in the original plan, it refers to the purchase timestamp of the ticket.

$$\mathcal{A}\rightarrow Merchant:E_{(k_{\mathcal{A}}^{-1})}(E_{k_M}(Ticket,\mathcal{A},timestamp))$$

If $\mathcal{A}$ wants to attack successfully, he must successfully forge *Merchant's* signature to pass *Merchant's* verification at the time of consumption. Then he can get *Merchant's* double signature and successfully broadcast the consumption process on the blockchain. As can be obtained from Theorem 1, the signature is non-forgerable, so the forged signature cannot pass verification and the attack fails. If adversary wants to forge the signature, it is not feasible in computing because the private key is classified. *Merchants* can verify users' timestamp and signature, and attackers cannot deny the signature. In turn, merchants or attackers cannot tamper the users' signature and make any changes to the status of the tickets, otherwise the verification will fail. In addition, timestamp could prevent the repeated use of the tickets. The scheme deals with the anonymity of tickets with the mixed currency protocol. Due to the return address is through an effective ring signature, hybrid server can only ensure members from a legal address. But it cannot distinguish users' addresses. External attackers can only intercept individual transactions and addresses. They can't relate the address to the transaction. It effectively guarantees the users' anonymity, even if the hybrid server is compromised. In transactions, users and merchants make blind transactions through a hybrid server. The user acts as an input to the hybrid server and merchant acts as the output of the hybrid server. Hybrid servers cannot track users' transactions, and ring signatures ensure that the relationship between their input and output addresses is not visible. Where signature is signed on the ring by one of users and no information about who signed it is disclosed. Therefore, the mixed currency protocol based on ring signature can ensure that users are anonymous to the hybrid server, and the outside of the protocol. Although the server of consortium blockchain knows the real information of the user, but it does not know the real transaction of the user. Moreover, the consortium blockchain server will not disclose privacy of the user, so blockchain external will not get the user's privacy information. The anonymity of the user can be guaranteed.

*b. Double Spending Attack*

In this paper, the DPoS consensus mechanism is used to establish CB-ETS, and it effectively solves the double spending problem. DPoS leverages the power of stakeholder approval voting to resolve consensus issues in a fair and democratic way. A double spending can occur anytime in a blockchain reorganization excludes a transaction previously included. This means that the witnesses had a communication breakdown caused by disruptions in the infrastructure of the Internet. With DPoS, the probability of a communication breakdown enabling a double spend attack is very low. Because the nodes' relationship of DPoS is cooperation rather than competition. Nodes package the blocks in an orderly manner, there's no branching basically. What's more, the users' identity information is contained in tickets, which can also be on the track, so it's impossible to make double spending. In addition, the improved algorithm of DPoS was proposed in March 2018, which combines BFT with DPoS and got a kind of hybrid consensus mechanism. It makes time for a block reduce from 3s to 500ms, so it will better prevent the attack.

*c. Sale of Invalid Tickets*

Each ticket has its own identity information and is stored in the blockchain. Consider the following attack scenario: suppose the *Merchant* is malicious, and the invalid ticket is sold to the buyer Alice. $E_{k_A}$ refers to Alice signs the tickets with his private key and $E_{k_M}$ refers to *Merchant* signs the tickets with his own private key. *Ticket* refers to some specific information of the ticket. *Ticket state* refers to the state of the ticket. *Alice* refers to the information related to Alice on the ticket. timestamp refers to Alice's timestamp when buying the ticket.

$$Merchant \rightarrow Alice : E_{(k_M)}(E_{k_A}(Ticket, Ticket\ state, Alice, timestamp))$$

The tickets, whether are purchased, transferred or consumed, all require the users' signature and *Merchants'* signature. (It needs the signatures of users and *Merchants* while

purchasing and consuming. Also, when transferring the ticket, it needs the signatures of transferring of the users and *Merchants*.) After the relevant operation is completed, *Merchants* will verify and broadcast to the blockchain. If the ticket is invalid, such an attack cannot happen, because the state of invalid tickets have changed and is recorded on the blockchain.

*d. Modify the Invalidity of the tickets*

The scheme uses multi-signature in the process of modifying the invalidity of the tickets, which uses the signatures of *Merchants* and users to codetermine the validity of the tickets. *Merchants* can't repudiate the tickets unilaterally. Even if *Merchants* know the user's public key $pk_i$ and certificate $cert_i$ , they also could not forge the signature of users. Therefore, the user's rights are guaranteed.

*e. Resistant to DoS Attacks*

The user may not respond to an address request during initialization, causing the entire process to fail. This is a Dos attack. In our scheme, the mixed currency protocol [4] is adopted, in which user only need to establish a connection to the mixed server. Any user who does not comply with the rules of the protocol can be detected and excluded. Therefore, our scheme can effectively prevent the Dos attack. There is another Dos attack in which an attacker prevents nodes from collecting and packaging transaction records onto blockchain, but this attack must satisfy 51% of the attacks. Our scheme can completely resist this kind of Dos attack.

*f. Resistant to Scalpers Attacks.*

In order to prevent scalpers from disturbing the market order, this paper adds merchants as the third party in the process of transferring. In this way, the scalpers can be avoided to complete the transaction of the high-priced transferring ticket under the blockchain after direct contact with the users. In addition, we have adopted tax mechanism and other effective anti-scalpers mechanisms on the blockchain to protect the interest of users.

### 5.3 Performance evaluation

This section mainly analyzes the performance of our scheme and compares it with related methods[13,15]. And the comparison results are shown in **Table 6**. We conducted security analysis of our scheme with the scheme of traditional electronic ticketing system [13] and the ticketing system of specific scenarios with blockchain [15]. As can be seen from **Table 6**, our scheme is not only anonymous, transferable, but also able to resist Double-spending attacks and Dos attacks. In addition, our scheme is also robust against specific attacks in [15].

**Table 6.** Comparison of security

| Scheme | Anonymity | Transferability | Double-Spending | Dos | Sale of Invalid Tickets | Invalidity of the tickets |
|---|---|---|---|---|---|---|
| [13] | Yes | Yes | Yes | No | No | No |
| [15] | No | Yes | Yes | Yes | Yes | No |
| Ours | Yes | Yes | Yes | Yes | Yes | Yes |

### 5.4 Cost-Effectiveness analysis

a. Theoretical analysis

In this section, we analyze the efficiency of the proposed scheme. We compared our scheme with the traditional electronic ticketing system in [13].There are two schemes are presented in [13]: electronic ticketing system based on RSA signature and group signature. Although ETS based on group signature achieves strong anonymity, the efficiency of group signature and zero-knowledge proof is low. Therefore, we only compared the efficiency analysis with the

scheme based on RSA signature. We will analyze the efficiency from two aspects: CoC (Cost of Communication) and TL (Traffic Load). In the CoC comparison, we mainly calculated and explained the interaction times of the whole protocol. As can be seen from Table 7, our scheme is significantly superior to the scheme [13]. Our scheme is also superior to the scheme [13] in the ticket transferring process. When TL is compared, the interaction amount in the whole protocol is mainly calculated and explained. As can be seen from Table 7, our scheme is better than the scheme [13], and has almost twice the advantage in ticket transferring process. The scheme comparison diagram can also be clearly seen from **Fig. 4** and **Fig. 5**.

**Table 7.** Comparison of CoC and TL

| **phase** | **Ours** | | | **[13]** | | |
|---|---|---|---|---|---|---|
| | Interacting parties | CoC | TL | Interacting parties | CoC | TL |
| System Initialization | User<br>A third party<br>Nodes | 3 | 9 | User<br>RA<br>CA | 4 | 9 |
| Buying | User<br>Nodes<br>Merchant | 3 | 14 | User<br>Issuer | 7 | 21 |
| Consuming | User<br>Merchant<br>Nodes | 2 | 16 | No | No | No |
| Transferring | Users<br>Merchant | 5+3n | 11+8n | Users<br>Prover | 6n | 5+15n |

b. Experimental analysis

**Fig. 4** and **Fig. 5** shows the main communication cost and traffic load of the proposed scheme and previous study in [13]. As shown in **Fig. 4** and **Fig. 5**, we calculate the total computational cost for the user at each stage.

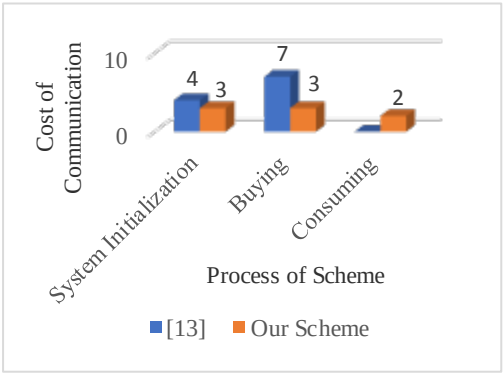


**Fig. 4.** Comparison of CoC

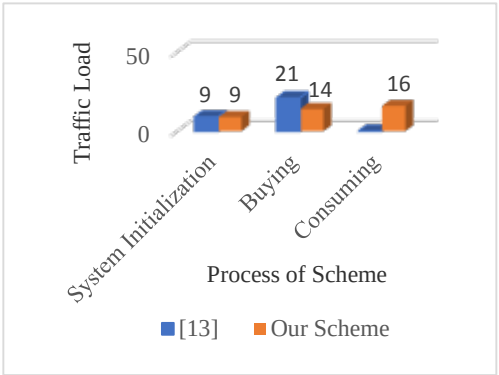


**Fig. 5.** Comparison of TL

We use the Java security API to implement these cryptographic operations. The implement platform is a laptop with an Intel Core i3-6100, 3.70GHz processor, 8GB memory, and Windows 7 Ultimate, 64-bit 6.1.7601, Service Pack 1 operating system. We use Eclipse 4.8 and MIRACL library to implement.

**Table 8**. Computational cost of cryptographic algorithms

| Operations | Time/ms |
|---|---|
| RSA keys generation $T_{RSA-kg}$ | 4850.712 |
| RSA encryption/verification $T_{RSA-e/v}$ | 4.012 |
| RSA decryption/signing $T_{RSA-d/s}$ | 172.371 |
| ECDSA keys generation $T_{ECDSA-kg}$ | 323.3808 |
| ECDSA encryption/verification $T_{ECDSA-e}$ | 636.286 |
| ECDSA decryption/signing $T_{ECDSA-d}$ | 214.902 |

**Table 8** shows the computation time of some cryptographic algorithms under the same security level. The notations $T_{RSA-kg}$, $T_{RSA-e/v}$, $T_{RSA-d/s}$, $T_{ECDSA-e}$, $T_{ECDSA-kg}$ and $T_{ECDSA-d}$ represent one RSA keys generation [45], one RSA encryption/verification with a 1024 bits modulus [44], one RSA decryption/signing with a 1024 bits modulus [44], one ECDSA keys generation with a 160 bits modulus [45], one ECDSA encryption with a 160 bits modulus [45], and one ECDSA decryption with a 160 bits modulus [45] respectively.

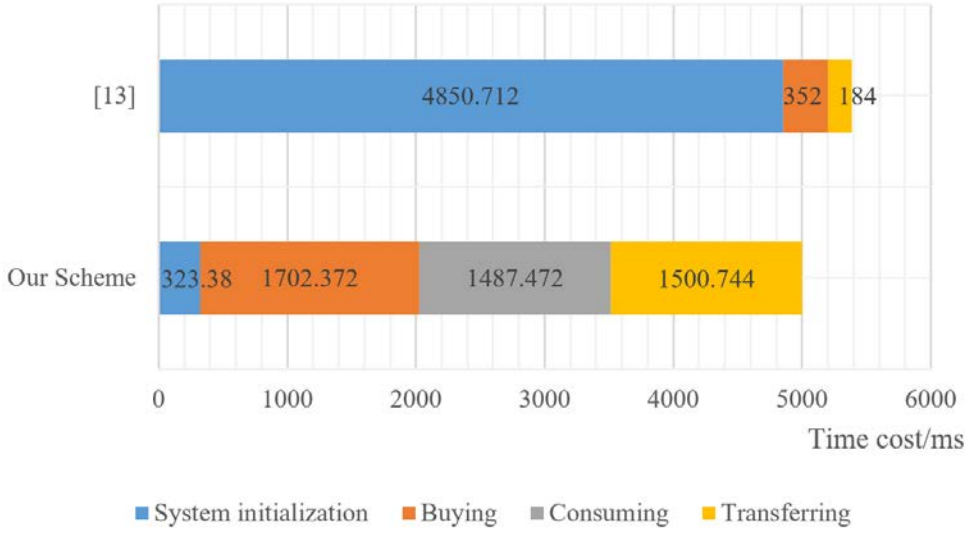


**Fig. 6.** Comparison of time cost

According to the parameters in Table 8, we simulated each process of the scheme. Fig. 6 shows the main computational cost of the proposed scheme and previous studies in [13]. As shown in Fig. 6, we calculate the total computational cost at each process. We can get that although our scheme is relatively time-consuming in the process of buying, consuming and transferring tickets, the time cost of our scheme in the process of system generation is far less than the time cost of literature [13]. Since the author did not discuss the process of consuming tickets in literature [13], we set the time cost of consuming tickets to 0. After running the whole electronic ticketing system, our scheme took 5013.968ms, while the final time of literature [13] was 5386.712ms. What's more, the time of consumption process is included in the calculation of the total time of our scheme, which is 1487.472ms. Therefore, if the time of consumption process is not taken into account, our scheme only needs to spend 3513.2248ms, which is far less than the literature [13]. So our scheme achieves the goal of efficiency.

### 5.5 Algorithm Implementation of Optimization

**Table 9.** The Optimization Algorithm

| The Algorithm 4 |
|---|
| 1. *Input:* $\varepsilon$, $\eta$; |
| 2. *Initialization:* $A^{n(0)}$, $E^{n(0)}$, $t \leftarrow 0$, $flag \leftarrow 1$; |
| 3. Based on $A^{n(t)}$ and $E^{n(t)}$, the auctioneer gets $BT^n$ and $TT^n$ , and then broadcasts the optimized results to buyers and transferees, respectively. |
| 4. Based on $BT^n$ and $TT^n$, the buyers compute their optimal bid prices $A_i^n$ through solving Problem EA, and submit them to the merchant. |
| 5. Based on $BT^n$ and $TT^n$, the transferees compute their optimal bid prices $E_j^n$ through solving Problem EE, and submit them to the merchant. |
| 6. $t \leftarrow t + 1$. |
| 7. If RA< $\varepsilon$ and RE < $\varepsilon$ , then merchant stops the iteration. |

According to our scheme, the buyer and the transferee will submit their own bid price vectors and merchant will achieve a win-win situation through these bid prices. Then merchant broadcasts the information. Finally, users compute and obtain optimal bid prices for the next iteration. Therefore, we will give the iteration algorithm in order to get the optimal prices. What's more, the stopping criterion for iteration are as follows.

$$RA = \left|a_{ij}^{n(t)} - a_{ij}^{n(t-1)}\right| \Big/ a_{ij}^{n(t)}, RE = \left|e_{ji}^{n(t)} - e_{ji}^{n(t-1)}\right| \Big/ e_{ji}^{n(t)} \tag{42}$$

According to the optimization algorithm, let $l_1 = 0.01, l_2 = 0.015, \eta = 0.8, \varepsilon = 0.001$ . And we will choose 35 buyers and 45 transferees as an example. **Fig. 7** shows that the result of the emulation of win-win point according to the Optimization Algorithm. We can get that the win-win point rapidly converges close to the optimal one after 16 iterations. What's more, the optimal one can be obtained when the transaction volume reaches 24.

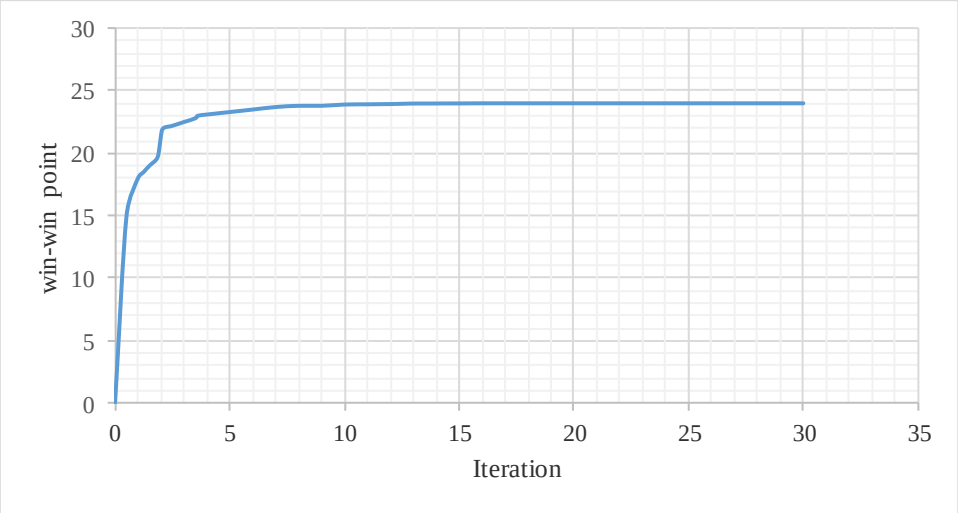


**Fig. 7.** Emulation of win-win point

## 6. Conclusion

In order to solve the problem of scalpers, some proposals have been put forward for ETS, such as asking the government to take legal measures against it. However, the proposal does

not regulate the scalpers yet. In this paper, we establish CB-ETS with DPoS. We propose a kind of taxation mechanism for scalpers. Together with the regulatory mechanism, the scalpers have been effectively suppressed. It makes scalpers have few profits but pay a heavy price when they choose to violate the benefits of other users. What's more, the reward mechanism is used to encourage supervision among users and jointly maintain the security of CB-ETS. Dual control effectively solves the problem of scalpers. Moreover, the security analysis illustrates that the system is robust. The blockchain system and digital signature used effectively preserves some properties of the tickets. In addition, our scheme is compared with literatures [13] and [15]. Our scheme is better than [13] in Coc (Cost of Communication) and TL (Traffic Load), and has almost twice advantages in ticket transfer process. Moreover, we implement the CB-ETS, which shows our scheme is more efficient compared to [13]. We also find the win-win point through iterative algorithm in the example. Finally, we show that our scheme is not only anonymous, transferable, but also robust against Double-spending attacks, Dos attacks and specific attacks in [15].

## Acknowledgement

This work was supported in part by the National Key Research and Development Program of China (No, 2016YFB0800601) and the Natural Science Foundation of China (No. 61303217, 61502372). We would like to express our gratitude to Prof. Kyungbaek Kim and reviewers who checked our manuscript.

## References

[1] Chih-Lin I, Han S and Xu Z, et al, "5G: rethink mobile communications for 2020+," *Philosophical Transactions of the Royal Society A, Mathematical, Physical and Engineering Sciences*, 374(2062), 20140432, 2016. Article (CrossRef Link).
[2] Compose, "5G shaping the always on networks of tomorrow," 2018. https://www.commscope.com/5g/wp-5G-shaping-the-always-on-networks/
[3] GSM Association, "Mobile's Green Manifesto 2012,". http://www.gsma.com/mee
[4] Fast Track Research Institute, "Report on Online Ticket Market Analysis in the First Quarter of 2017," in *Proc. of Internet World*, 25 – 27, 2017. Article (CrossRef Link).
[5] Shu Kai, "Significance and Implementation of Electronic Ticketing System," *Wireless Music Education Frontier*, vol. 9, p. 128, 2015. Article (CrossRef Link).
[6] Ziegeldorf J H, Matzutt R and Henze M, et al, "Secure and anonymous decentralized Bitcoin mixing," *Future Generation Computer Systems*, vol. 80, pp. 448-466, 2018. Article (CrossRef Link).
[7] Yli-Huumo J, Ko D and Choi S, et al, "Where Is Current Research on Blockchain Technology?-A Systematic Review," *Plos One*, 11(10), e0163477, 2016.
[8] Bershad L, "Ticket Scalping Legislation-A New Jersey Case Study," 1985. https://heinonlinebackup.com/hol-cgi-bin/get_pdf.cgi?handle=hein.journals/sethlegj9&section=9.
[9] Li M, Xu A and Xue N, et al, "Enable Bitcoin Transcation in Public Transport Ticketing System," in *Proc. of Conference on Emerging Topics in Interactive Systems*, 2016. Article (CrossRef Link).
[10] Lamont C, "Automated ticketing system and method for law enforcement," US20040094621, 2004. https://patents.glgoo.top/patent/US20040094621A1/en
[11] Xu W, "Movie ticket vending system and hand-held electronic device and method thereof," US9124782B2, 2015. https://patents.glgoo.top/patent/US9124782B2/en
[12] Jin X and Guan X, et al, "Research and Development of an Intelligent Software for Dispatching Order Operating Ticket System," *Journal of Shanghai Institute & Electric Power*, 2001. Article (CrossRef Link).

[13] M. Magdalena Payeras-Capellà and Macià Mut-Puigserver et al, "Design and Performance Evaluation of Two Approaches to Obtain Anonymity in Transferable Electronic Ticketing Schemes," *Mobile Networks & Applications*, 22(6), 1137-1156, 2017. Article (CrossRef Link).
[14] Kuserk G J and Locke P R, "Scalper behavior in futures markets: An empirical examination," *Journal of Futures Markets*, 13(4), 409-431, 1993. Article (CrossRef Link).
[15] Tackmann B, "Secure Event Tickets on a Blockchain," in *Proc. of Garcia-Alfaro J, Navarro-Arribas G, Hartenstein H, Herrera-Joancomartí J (eds) Data Privacy Management, Cryptocurrencies and Blockchain Technology. DPM 2017, CBT 2017. Lecture Notes in Computer Science*, vol. 10436, pp. 437-444, 2017. Article (CrossRef Link).
[16] R. M. Florian Mathieu, "Blocktix: Decentralized Event Hosting and Ticket Distribution," 2017.
[17] Sidhu J, "Syscoin: A Peer-to-Peer Electronic Cash System with Blockchain-Based Services for E-Business," in *Proc. of International Conference on Computer Communication & Networks*, 2017. Article (CrossRef Link).
[18] Araujo F, Curado M and Furtado P, "Taking an electronic ticketing system to the cloud: Design and discussion," in *Proc. of Workshop on Scalable Cloud Data Management*, 2014. Article (CrossRef Link).
[19] Feng X, Ma J and Miao Y, et al, "Pruneable sharding-based blockchain protocol," *Peer-to-Peer Networking and Applications*, vol. 12, no. 4, pp. 934-950, 2019. Article (CrossRef Link).
[20] Mimata Y, "Ticket issuing system, ticket checking system, check system, retrieving system and automatic examination machine," US, US6070146 A, 2000. https://patents.glgoo.top/patent/US6070146A/en
[21] Parker N, "MOBILE TICKETING: Springer Berlin Heidelberg," WO/2014/055772, 2014.
[22] Shi Z, Changhe L I and Feng Y, et al, "Electric Operation Ticket Expert System Based on BP," *Computer Engineering*, 2003.
[23] Kohta Y, "Electronic ticket issuing system and electronic ticket issuing method," US, US7004388, 2006. https://patents.glgoo.top/patent/US7004388B2/en
[24] Meade R, Schofield P and Wright M C, "Ticket issuing systems," US, US4845650, 1989. https://patents.glgoo.top/patent/US4845650A/en
[25] Yuris N, Cekander E J and Kazaoka M, et al, "Self-service passenger ticketing system," US, US4247759 A, 1981. https://patents.glgoo.top/patent/US4247759A/en
[26] Lewis W C, "Electronic ticketing and validation system and method," US20030105641, 2003. https://patents.glgoo.top/patent/US20030105641A1/en
[27] Boyd L F, "Method of operating a ticketing system," US, US7520427, 2009. https://patents.glgoo.top/patent/US20030164400A1/en
[28] Wilder W B, "Automated ticket sales and dispensing system," US, US5408417 A, 1995. https://patents.glgoo.top/patent/US5408417A/en.
[29] Yanai H, "ELECTRONIC TICKET SYSTEM," EP, EP1267289, 2002. https://patents.glgoo.top/patent/US20030154169A1/en
[30] Mimata Y, "Ticket issuing system, ticket checking system, check system, retrieving system and automatic examination machine," US, US6070146 A, 2000. https://patents.glgoo.top/patent/US6070146A/en
[31] Bergdale M, Grasser M and Ihm N, et al, "METHOD AND SYSTEM FOR ELECTRONIC TICKET VALIDATION USING PROXIMITY DETECTION," US20150142483, 2015. https://patents.glgoo.top/patent/US20150084741A1/en
[32] Robinton M and Wähl F, "TRUSTED NFC TICKETING,"US20170017947, 2017. https://patents.glgoo.top/patent/US20170017947A1/en
[33] Bergdale M, Ihm N and Valyer G, et al, "SYSTEMS AND METHODS FOR ELECTRONIC TICKET VALIDATION USING PROXIMITY DETECTION FOR TWO OR MORE TICKETS," US 20150294515 A1, 2015. .https://patents.glgoo.top/patent/US20150294515A1/en
[34] Androulaki E, Barger A and Bortnikov V, et al, "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains," in *Proc. of the Thirteenth EuroSys Conference*, 2018. Article (CrossRef Link).

[35] Liu Y and Li R, et al, "Enhancing Anonymity of Bitcoin Based on Ring Signature Algorithm," in *Proc. of International Conference on Computational Intelligence and Security. IEEE*, 317-321, 2017. Article (CrossRef Link).
[36] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," pp. 1-9, 2008. Article (CrossRef Link).
[37] Li W, Andreina S, Bohli JM and Karame G, "Securing Proof-of-Stake Blockchain Protocols," in *Proc. of Garcia-Alfaro J., Navarro-Arribas G., Hartenstein H., Herrera-Joancomartí J. (eds) Data Privacy Management, Cryptocurrencies and Blockchain Technology. DPM 2017, CBT 2017. Lecture Notes in Computer Science*, vol. 10436, pp. 297-315, 2017. Article (CrossRef Link).
[38] Bartoletti M, Lande S and Podda A S, "A Proof-of-Stake Protocol for Consensus on Bitcoin Subchains," in *Proc. of International Conference on Financial Cryptography and Data Security. Springer, Cham*, 568-584, 2017. Article (CrossRef Link).
[39] S. Daniel Larimer, "DPOS Consensus Algorithm - The Missing White Paper," 2014.
[40] S. Miguel Castro and Barbara, "Practical Byzantine Fault Tolerance,".
[41] Kraft D, "Difficulty control for blockchain-based consensus systems," *Peer-to-Peer Networking and Applications*, 9(2), 397-413, 2016. Article (CrossRef Link).
[42] Nofer M, Gomber P and Hinz O, et al, "Blockchain," *Business & Information Systems Engineering*, 59(3), 183-187, 2017. Article (CrossRef Link).
[43] Fanning K and Centers D P, "Blockchain and Its Coming Impact on Financial Services," *Journal of Corporate Accounting & Finance*, 27(5),53-57, 2016. Article (CrossRef Link).
[44] Rivest R, Shamir A and Adleman L, "A method for obtaining digital signature and public key cryptosystems," *Commun*, vol.21, pp.120–126, 1978. Article (CrossRef Link).
[45] Cuifang Xing, Ying Li and Haibing Zhao, "A technical solution for mobile Web service security," *Computer technology and development*, 4, 122-125, 2013.

**Xuelian Li** received the Ph.D degree in cryptography in 2010. She is now an associate professor in School of Mathematics and Staticstics, Xidian University. Her research interests focus on information security and blockchain.(Email: xlli@mail.xidian.edu.cn, xuelian202@163.com)

**Jie Niu** was born in Shuozhou, Shanxi Province, China in 1995. She is currently pursuing the M.S.degree with Xidian University, Xi'an, China. Her research interests include Blockchain, Internet of Things, and searchable encryption.(Email: Niu_Jie1754@163.com)

**Juntao Gao** received the Ph.D degree in cryptography in 2006. He is now an associate professor in School of Telecommunication and Engineering, Xidian University. His research interests focus on pseudorandom sequences and blockchain. (Email: jtgao@mail.xidian.edu.cn)

**Yue Han** was born in Yuncheng, Shanxi Province, China in 1994. She is currently pursuing the M.S. degree with Xidian University, Xi'an, China. Her research interests include Internet of Vehicles. (Email: Han_Yue0526@163.com)